Limitations of the Recall Capabilities in Delay-Based Reservoir Computing Systems

Felix Köster1 · Dominik Ehlert1 · Kathy Lüdge1

Received: 28 February 2020 / Accepted: 14 May 2020 / Published online: 2 October 2020 © The Author(s) 2020

Abstract
We analyse the memory capacity of a delay-based reservoir computer with a Hopf normal form as nonlinearity and numerically compute the linear as well as the higher order recall capabilities. A possible physical realization could be a laser with external cavity, for which the information is fed via electrical injection. A task-independent quantification of the computational capability of the reservoir system is done via a complete orthonormal set of basis functions. Our results suggest that even for constant readout dimension the total memory capacity is dependent on the ratio between the information input period, also called the clock cycle, and the time delay in the system. Optimal performance is found for a time delay about 1.6 times the clock cycle.

Keywords Lasers · Reservoir computing · Nonlinear dynamics

Introduction
Reservoir computing is a machine learning paradigm [1] inspired by the human brain [2], which utilizes the natural computational capabilities of dynamical systems. As a subset of recurrent neural networks it was developed to predict time-dependent tasks with the advantage of a very fast training procedure. Generally the training of recurrent neural networks is connected with high computational cost resulting e.g. from connections that are correlated in time. Therefore, problems like the vanishing gradient in time arise [3]. Reservoir computing avoids this problem by training just a linear output layer, leaving the rest of the system (the reservoir) as it is. Thus, the inherent computing capabilities can be exploited. One can divide a reservoir into three distinct subsystems, the input layer, which corresponds to the projection of the input information into the system, the dynamical system itself that processes the information, and the output layer, which is a linear combination of the system’s states trained to predict an often time-dependent task.

Many different realizations have been presented in the last years, ranging from a bucket of water [4] over field programmable gate arrays (FPGAs) [5] to dissociated neural cell cultures [6], being used for satellite communications [7], real-time audio processing [8, 9], bit-error correction for optical data transmission [10], amplitude of chaotic laser pulse prediction [11] and cross-predicting the dynamics of an injected laser [12]. Especially opto-electronic [13, 14] and optical setups [15–19] were frequently studied because their high speed and low energy consumption make them preferable for hardware realizations.

The interest in reservoir computing was refreshed when Appeltant et al. showed a realization with a single dynamical node under influence of feedback [20], which introduced a time-multiplexed reservoir rather than a spatially extended system. A schematic sketch is shown in Fig. 1. In general the delay architecture slows down the information processing speed but reduces complexity of the hardware. Many neuron based, electromechanical, opto-electronic and photonic realizations [21–26] showed the
capabilities from time series predictions [27, 28] over an equalization task on nonlinearily distorted signals [29] up to fast word recognition [30]. More general analysis showed the general and task-independent computational capabilities of semiconductor lasers [31]. A broad overview is given in [32, 33].

In this paper we perform a numerical analysis of the recall capabilities and the computing performance of a simple nonlinear oscillator, modelled by a Hopf normal form, with delayed feedback. We calculate the total memory capacity as well as the linear and nonlinear contributions using the methodology derived by Dambre et al. in [34].

The paper is structured as follows. First, we shortly explain the concept of time-multiplexed reservoir computing and give a short overview of the method used for calculating the memory capacity. After that we present our results and discuss the impact of the delay time on the performance and the different nonlinear recall contributions.

Methods

Traditionally, reservoir computing was realized by randomly connecting nodes with simple dynamics (for example the tanh-function [1]) to a network, which was then used to process information. The linear estimator of the readouts is then trained to approximate a target, e.g. predict a time series prediction. The input is preprocessed by multiplication with a mask that is then electrically injected. The laser in our case is governed by a Hopf normal form. The output of state reactions is then trained to approximate a target, e.g. predict a time to process information. The linear estimator of the readouts is then trained to approximate a target, e.g. predict a time series prediction.

Methods

Traditionally, reservoir computing was realized by randomly connecting nodes with simple dynamics (for example the tanh-function [1]) to a network, which was then used to process information. The linear estimator of the readouts is then trained to approximate a target, e.g. predict a time to process information. The linear estimator of the readouts is then trained to approximate a target, e.g. predict a time series prediction.

Methods

Traditionally, reservoir computing was realized by randomly connecting nodes with simple dynamics (for example the tanh-function [1]) to a network, which was then used to process information. The linear estimator of the readouts is then trained to approximate a target, e.g. predict a time to process information. The linear estimator of the readouts is then trained to approximate a target, e.g. predict a time series prediction.

Methods

Traditionally, reservoir computing was realized by randomly connecting nodes with simple dynamics (for example the tanh-function [1]) to a network, which was then used to process information. The linear estimator of the readouts is then trained to approximate a target, e.g. predict a time to process information. The linear estimator of the readouts is then trained to approximate a target, e.g. predict a time series prediction.

Methods

Traditionally, reservoir computing was realized by randomly connecting nodes with simple dynamics (for example the tanh-function [1]) to a network, which was then used to process information. The linear estimator of the readouts is then trained to approximate a target, e.g. predict a time to process information. The linear estimator of the readouts is then trained to approximate a target, e.g. predict a time series prediction.

Methods

Traditionally, reservoir computing was realized by randomly connecting nodes with simple dynamics (for example the tanh-function [1]) to a network, which was then used to process information. The linear estimator of the readouts is then trained to approximate a target, e.g. predict a time to process information. The linear estimator of the readouts is then trained to approximate a target, e.g. predict a time series prediction.
complete orthonormal set of basis functions on a sequence of inputs \( u_n = (\ldots, u_{n-2}, u_{n-1}, u_n) \) at time \( n \). In this case the index indicates the input \( \Delta t \) time steps ago. The goal is to investigate how the system transforms the inputs \( u_n \). For this the chosen basis functions \( z(u_n) \), forming a Hilbert space, are constructed and used to describe every possible transformation on the inputs \( u_n \). The system’s capability to approximate these basis functions is evaluated. Consider the following examples: The function \( z(u_n) = u_{n-5} \) is chosen as a task \( o_n \). This is a transformation of the input sequence 5 steps back. The question this task asks is, how well the system can remember the input 5 steps ago. Another case would be \( o_n = z(u_n) = u_{n-5}u_{n-2} \), asking how well it can perform the nonlinear transformation of multiplying the input 5 steps into the past with the input 2 steps into the past. A useful quantity to measure the capability of the system is the capacity defined as

\[
C = 1 - \text{NRMSE}^2.
\]  

A value of 1 corresponds to the system being perfectly capable of approximating the transformation task and 0 corresponds to no capability at all. A simpler method, giving equal results like Eq. (4) developed by Dambre et al. in [34] to calculate \( C \) is given by

\[
C = \frac{\mathbf{o}^T \mathbf{z}(\mathbf{s}) T^{-1} \mathbf{s} \mathbf{o}}{\|\mathbf{o}\|^2 N^2},
\]

where \( T \) indicates the transpose of a matrix and \(-1\) the inverse. We use Eq. (5) to calculate the memory capacity.

In this paper we use finite products of normalized Legendre polynomials \( P_{\Delta t} \) as a full basis of the constructed Hilbert space for each input step combination. \( d \) is the order of the used Legendre polynomial and \( \Delta t \) the \( \Delta t \)th step into the past passed as value to the Legendre polynomial. Multiplying a set of those Legendre polynomials gives the target task \( y|_{\Delta t} \), which yields (see example below for clarification)

\[
y|_{\Delta t} = \Pi \Delta t P_{\Delta t} (u_{-\Delta t}).
\]

This is directly taken from [34]. It is important that the inputs to the system are uniformly distributed random numbers \( u_n \), which are independent and identically drawn in \([-1, 1]\) to match the used normalized Legendre polynomials. To calculate the memory capacity \( MC^d \) for a degree \( d \), a summation over all possible past input sets is done

\[
MC^d = \sum_{\{\Delta t\}} C^d_{\{\Delta t\}},
\]

where \( \{\Delta t\} \) is the set of past input steps, \( C^d_{\{\Delta t\}} \) is the capacity of the system to approximate a specific transformation task \( z|_{\Delta t}(u_n) \) and \( d \) is the degree of all Legendre polynomials combined in the task \( z|_{\Delta t}(u_n) \). In the example from above with \( z|_{-5,-2}(u_n) = u_{n-5}u_{n-2} \), it is \( d = 2 \) and \( \{\Delta t\} = \{-5,-2\} \). For \( d = 1 \) we get the well-known linear memory capacity. To compute the total memory capacity, a summation over all degrees \( d \) is done.

\[
MC = \sum_{d=1}^{D} MC^d
\]

Dambre et al. showed in [34] that the \( MC \) is limited by the readout dimension \( M \), given here by the number of virtual nodes \( N_V \).

The simulation was written in C++ with standard libraries used except for linear algebra calculations, which were calculated via the linear algebra library “Armadillo”. A Runge-Kutta 4th-order method was applied to integrate numerically the delay-differential equation given by Eq. (10) with an integration step \( \Delta t = 0.01 \) in time units of the system. First, the system was simulated without any inputs to let transients decay. Afterwards a buffer time was applied with 100000 inputs, that were excluded from the training process. Then, the training and testing process itself was done with 250000 inputs to have sufficient statistics. The tasks are constructed via Eq. (6) and the corresponding capacities \( C^d_{\{\Delta t\}} \) were calculated via Eq. (5). All possible combinations of the Legendre polynomials up to degree \( D = 10 \) and \( \Delta t = 1000 \) input steps into the past were considered. \( C^d_{\{\Delta t\}} \) below 0.001 were excluded because of finite statistics. To calculate the inverse, the Moore–Penrose pseudoinverse from the C++ linear algebra library “Armadillo” was used.

We characterize the performance of our nonlinear oscillator by evaluating the total memory capacity \( MC \), the contributions \( MC^d \) as well as the NRMSE of the NARMA10 task. The latter is a benchmark test and combines memory and nonlinear transformations. It is given by an iterative formula

\[
A_{n+1} = 0.3A_n + 0.05A_n \left( \sum_{i=0}^{9} A_{n-i} \right) + 1.5u_{n-9}u_n + 0.1.
\]  

Here, \( A_n \) is an iteratively given number and \( u_n \) is an independent and identically drawn uniformly distributed random number in \([0, 0.5]\). The reservoir is driven by the random numbers \( u_n \) and has to be able to predict the value of \( A_{n+1} \), \( \mathbf{o} = \mathbf{A} \). The reservoir we use for our analysis is a Stuart-Landau oscillator, also called Hopf normal form [36], with delayed feedback. This is a generalized model applicable for all systems operated close to a Hopf bifurcation, i.e. close to the onset of intensity oscillations. One example would be a laser operated closely above bifurcation, i.e. close to the onset of intensity oscillations.
by
\[ \dot{Z} = (\lambda + \eta g I + i \omega + \gamma |Z|^2)Z + \kappa e^{i \phi} Z(t - \tau), \] (10)
and was taken from [18]. Here, \( Z \) is a complex dynamical variable (in the case of a laser \( |Z|^2 \) resembles the intensity), \( \lambda \) is a dimensionless pump rate, \( \eta \) the input strength of the information fed into the system via electrical injection, \( g \) is the masking function, \( I \) is the input, \( \omega \) is the frequency with which the dynamical variable \( Z \) rotates in the complex plane without feedback (in case of a laser, this is the frequency of the emitted laser light), \( \gamma \) the nonlinearity in the system, \( \kappa \) is the feedback strength, \( \phi \) the feedback phase and \( \tau \) the delay time. The corresponding parameters used in the simulations are found in Table 1 if not stated otherwise.

**Results**

To get a first impression about how the system can recall inputs from the input sequence \( u_n = (u_{n-2}, u_{n-1}, u_n) \), we show the linear recall capacities \( C^1_{\Delta n} \) in Fig. 2. Here, each set of all inputs \( \{\Delta n\} \) consists of only one input step \( \Delta n \), because \( d = 1 \) for which \( z_{\Delta n}(u_n) \) consists of the Legendre polynomial \( P_1(u_{n-\Delta n}) = u_{n-\Delta n} \). The capacities \( C^1_{\Delta n} \) are plotted over the step \( \Delta n \) to be recalled for 3 different delay times \( \tau \) (blue, orange and green in Fig. 2) while the input period time \( T \) is kept fixed to 80 and the readout dimensions \( N_V \) to 50. These timescale parameters were chosen to fit the characteristic timescale of the system, such that the time between two virtual nodes \( \theta \) is long enough for the system to react, but short enough such that the speeding process is still as high as possible. For input period times \( T = \tau \) (the blue solid line in Fig. 2) a high capacity is achieved for a few recall steps after which the recallability drops steadily down to 0 at about the 15th step \( (\Delta n = 15) \) to recall. This changes when the input period time reaches values of 3 times the delay time \( \tau = 3T \) (the orange solid line in Fig. 2). Here, the linear recallability \( C^1_{\Delta n} \) oscillates between high and low values as a function of \( n \), while its envelope steadily decreases until it reaches 0 at around the 35th \( (\Delta n = 35) \) step to be recalled. Considering that \( \tau = 3T \) is a resonance between the input period time \( T \) and the delay time \( \tau \), one can also take a look at the case for off-resonant setups, which is shown by the green solid line in Fig. 2 with \( \tau \approx 3.06T \). This parameter choice shows a similar behaviour as the \( T = 3\tau \) one but with higher capacities for short recall steps and a faster decay of the recallability at around the 29th \( (\Delta n = 29) \) step.

To get a more complete picture, we evaluated the linear capacity \( C^1_{\Delta n} \) and quadratic capacities \( C^2_{\Delta n} \) of the system and depicted these as a heatmap over the delay time \( \tau \) and the input steps in Fig. 3 for a constant input period time \( T \). The x-axis indicates the \( \Delta n \)th step to be recalled while the delay time \( \tau \) is varied from bottom to top on the y-axis. In Fig. 3a the linear capacities \( C^1_{\Delta n} \) are shown, for which the red horizontal solid lines indicate the scan from Fig. 2. One can see a continuous capacity \( C^1_{\Delta n} \) for \( \tau < 2T \) which forks into rays of certain recallable steps \( \Delta n \) that linearly increase with the delay time \( \tau \). This implies that specific steps \( \Delta n \) can be remembered while others inbetween are forgotten, a crucial limitation to the performance of the system. Generally the number of steps into the past that can be remembered increases with \( \tau \) (at constant \( T \)), while on the other hand also the gaps inbetween the recallable steps increase. Thus, the total memory capacity stays constant. This will be discussed later in Fig. 4. In Fig. 3b the pure quadratic capacity \( C^{2,p}_{\Delta n} \) is plotted within the same parameter space as in Fig. 3a. Pure means that only Legendre polynomials of degree 2, i.e. \( P_2(u_{n-\Delta n}) = \frac{1}{2}(3u^2_{n-\Delta n} - 1) \) were considered, rather than also considering combinations of two Legendre polynomials of degree 1, i.e. \( P_1(u_{n-\Delta n})P_1(u_{n-\Delta n}_2) = u_{n-\Delta n}u_{n-\Delta n}_2 \). In the graph one can see the same behaviour as for the linear capacities \( C^1_{\Delta n} \) (Fig. 3a), but with less rays and thus less steps that can be remembered from the past. This indicates that the dynamical system is not as effective in recalling inputs and additionally transforming them nonlinearly as it is in just recalling them linearly. For the full quadratic nonlinear transformation capacity, all

![Fig. 2](image-url)
Fig. 3  a Linear capacity $C_{1(\Delta n)}$ plotted color-coded over the delay time $\tau$ and the input steps $\Delta n$ to recall. Parameters as given in Table 1. The red horizontal solid lines indicate the scan from Fig. 2.  

b Quadratic pure capacity $C_{2(\Delta n)}$. c Combination of two Legendre polynomials of degree 1 indicating the capability of nonlinear transformations of the form $u_{-\Delta n_1} u_{-\Delta n_2}$. Here $\Delta n_1$ of the first polynomial is plotted while between two $\Delta n_1$-steps $\Delta n_2$ is increased from 0 to 45 steps into the past. Yellow indicates good while blue and black indicates bad recallability. The input period time $T = 80$

Combination of two Legendre polynomials of degree 1 for different input steps into the past have to be considered, i.e.

$$P_1(u_{n-\Delta n_1}) P_1(u_{n-\Delta n_2}) = u_{n-\Delta n_1} u_{n-\Delta n_2}.$$ 

This is shown in Fig. 3c. Again, the capacities $C_{2(\Delta n)}$ are depicted as a heatmap and the delay time $\tau$ is varied along the y-axis. This time the x-axis shows the steps of the first Legendre polynomial $\Delta n_1$, while inbetween two ticks of the x-axis, the second Legendre polynomial’s step $\Delta n_2$ is scanned from 0 up to 45 steps into the past. For the steps of the second Legendre polynomial $\Delta n_2$ the capacity exhibits the same behaviour as already discussed for Fig. 3a and b. This does also apply to the first Legendre polynomial which induces interference patterns in the capacity space of the two combined Legendre polynomials. The red dashed lines highlight the ray behaviour of the first Legendre polynomial. We therefore learn that the performance of a reservoir computer described by a Hopf normal form with delay drastically depends on the task. There are certain nonlinear transformation combinations $u_{-\Delta n_1} u_{-\Delta n_2}$ which cannot be approximated due to the missing memory at specific steps. To overcome these limitations it would be recommended to use multiple systems with different parameters to compensate for each other.

To fully characterize the computational capabilities of our reservoir computer, a full analysis of the degree d memory capacities $MC^d$ and the total memory capacity $MC$ as defined in Eq. (8) is done. The results are depicted in Fig. 4a as a function of the delay time $\tau$. All other parameters are fixed as in Fig. 3. The orange solid line in Fig. 4 refers to the linear, the green, red and violet lines to the quadratic, cubic and quartic memory capacity $MC^{1,2,3,4}$, respectively. The blue solid line shows the total memory capacity $MC$ summed up over all degrees up to 10. Dambre et al. showed in [34] that the $MC$ is limited by the number of read-out dimensions and equals it when all read-out dimensions are linearly independent. In our case the read-out dimension is given by the number of virtual nodes $N_V = 50$. Nevertheless, the total memory capacity $MC$ starts at around 15 for a very short time delay with respect to the input period time $T$. This low value arises from the fact that a short delay induces a high correlation
between the responses of the dynamical system which induces highly linearly dependent virtual nodes. This is an important general result that has to be kept in mind for all delay-based reservoir computing systems: With $\tau < 1.5T$ the capability of the reservoir computer is partially waisted. Increasing the delay time $\tau$ also increases the total memory capacity $MC$ reaching the upper bound of 50 at around 1.5 times the input period time $T$.

For $\tau > 1.5T$ an interesting behaviour emerges. Depicted by the vertical red dashed lines are multiples of the input period time $T$ at which the total memory capacity $MC$ drops again significantly to around 40. A drop in the linear memory capacity was discussed in the paper by Stelzer et al. [38] and explained by the fact that resonances between the delay time $\tau$ and the input period time $T$ concludes in a sparse connection between the virtual nodes. Our results now show that this effects the total memory capacity $MC$, by mainly reducing the quadratic memory capacity $MC^2$. At the resonances the quadratic nonlinear transformation capability of the system is reduced. To conclude, delay-based reservoir computing systems should be kept off the resonances between $T$ and $\tau$ to maximize the computational capability. A surprising result is that for the chosen Hopf nonlinearity the linear memory capacity $MC^1$ is only slightly influenced by the resonances. A result from Dambre et al. in [34] and analysed by Inubushi et al. in [39] showed that a trade-off between the linear recallability and the nonlinear transformation capability exists. This is clearly only the case if the theoretical limit of the total memory capacity $MC$ is reached and kept constant, thus every change in the linear memory capacity $MC^1$ has to induce a change in the nonlinear memory capacities $MC^d$, $d > 1$. In the case of resonances, a decrease in the total memory capacity $MC$ happens and thus this loss can be distributed in any possible way over the different memory capacities $MC^d$. In our case, we see that the influence on the quadratic memory capacity $MC^2$ is highest.

The system is capable of a small amount of cubic transformations, depicted by the solid red line in Fig. 4a, which also decreases at the resonances in a similar way as the quadratic contribution does. Higher order memory capacities $MC^d$, with $d > 3$, have only small contributions for short delay times $\tau$, dropping to 0 for increased time delay $\tau$. A possible explanation is the fact that short delays induce an interaction of the last input directly with itself for $k = \frac{\tau}{T}$ times, depending on the ratio between $\tau$ and $T$. As a result, short delay times $\tau$ enable highly nonlinear tasks in expense of a lower total memory capacity $MC$.

For more insights into the computing capabilities of our nonlinear oscillator we now also discuss the NARMA10 time series prediction task, shown in Fig. 4b. Comparing the memory capacities $MC^d$ with the NARMA10 computation error NRMSE in Fig. 4b, a small increase in the NARMA10 NRMSE can be seen at the resonances with $n\tau = mT$, where $n \in [0, 1, 2...]$ and $m \in [0, 1, 2...]$. For a systematic characterization a scan of the input period time $T$ and the delay time $\tau$ was done and the total memory capacity $MC$ (Fig. 5c), the memory capacities of degrees 1–4 $MC^1, 2, 3, 4$ (Fig. 5a, b, d, e) and the NARMA10 NRMSE (Fig. 5f) were plotted colorcoded over the two timescales. This is an extension of the results of Röhm et al. in [19], where only the linear memory capacity and the NARMA10 computation error were analysed. For short time delays $\tau$ and period input times $T$ the memory capacities of degree 1–3 $MC^1, 2, 3$ and the total memory capacity $MC$ are significantly below the theoretical limit of 50 as already seen in the results from Fig. 4, while the NARMA10 NRMSE also has high errors of around 0.8. This comes from the fact that short input period times $T$ also mean short virtual node distances $\theta$, which induces a high linear
correlation between the read-out dimensions. Degree 4 on the other hand only has values as long as $T > \tau$, a result coming from the fact that the input $u_n$ has to interact with itself to get a transformation of degree 4. A possible explanation comes from the fact that the dynamical system itself is not capable of transformations higher than degree 3, since the highest order in Eq. (10) is 3. If the delay time $\tau$ and the input period time $T$ are long enough the total memory capacity $MC$ reaches 50 with exceptions of resonances between $\tau$ and $T$. These resonances are also seen in the NARMA10 NRMSE for which higher errors occur. Looking at the memory capacity of degree 1 and 2 $MC^{1,2}$ and comparing it with the NARMA10 NRMSE one can see a tendency in which the NARMA10 NRMSE is lowest where both have the highest capacities, raising from the fact that the NARMA10 task is highly dependent on linear memory and quadratic nonlinear transformations. This can also be seen in the area below the $\tau = T$-resonance. To conclude, one can use the parameter dependencies of the memory capacities $MC^d$ to make predictions of the reservoir capability to approximate certain tasks.

Conclusions

We analysed the memory capacities and nonlinear transformation capabilities of a reservoir computer consisting of an oscillatory system with delayed feedback operated close to a Hopf bifurcation, i.e. a paradigmatic model also applicable for lasers close to threshold. We systematically varied the timescales and found regions of high and low reservoir computing performing abilities. Resonances between the information input period time $T$ and the delay time $\tau$ should be avoided to fully utilize the natural computational capability of the nonlinear oscillator. A ratio of $\tau = 1.6T$ was found to be the optimal for the computed memory capacities, resulting in a good NARMA10 task approximation. Furthermore, it was shown, that the recallability for high delay times $\tau \gg T$ is restricted to specific past inputs, which rules out certain tasks. By computing the memory capacities of a Hopf normal form, one can make general assumptions about the reservoir computing capabilities of any system operated close to a Hopf bifurcation. This significantly helps in understanding and predicting the task dependence of reservoir computers.

Acknowledgements The authors would like to thank André Röhm, Joni Dambre and David Hering for fruitful discussion.

Funding Open Access funding provided by Projekt DEAL. This study was funded by the “Deutsche Forschungsgemeinschaft” (DFG) in the framework of SFB910.

Compliance with Ethical Standards

Conflict of Interest The Authors declare that they have no conflict of interest.

Ethical Approval This article does not contain any studies with human participants or animals performed by any of the authors.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

Appendix

Derivation of the Stuart-Landau Equation with delay from the Class B laser rate equations

\begin{equation}
\dot{E} = (1 + i\alpha)EN \tag{11}
\end{equation}

\begin{equation}
\dot{N} = \frac{1}{T}(P + \eta g I - N - (1 + 2N)|E|^2), \tag{12}
\end{equation}

where $E$ is the non-dimensionalized complex electrical field and $N$ the non-dimensionalized carrier inversion, $P$ the pump relativ to the threshold for $P_{thresh} = 0$ and $\alpha$ the Henry factor. The reservoir computing signal is fed into the system via electrical injection $\eta g I$. If fast carriers are considered, an adiabatic elimination of the charge carriers yields

\begin{equation}
0 = \frac{1}{T}(P + \eta g I - N - (1 + 2N)|E|^2) \tag{13}
\end{equation}

\begin{equation}
N = \frac{P + \eta g I - |E|^2}{1 + 2|E|^2}, \tag{14}
\end{equation}

which after substituting into Eq. (11) gives

\begin{equation}
\dot{E} = (1 + i\alpha)E(\tilde{P} - |E|^2) \tag{15}
\end{equation}

where we introduced the quantity $\tilde{P} = P + \eta g I$ for convenience purposes. This equation yields the full Class A rate equation for the non-dimensionalized complex electric field. Simulations with the full Class A rate equation close to the threshold show similar results to the reduced case.

Because we consider laser that are operated close to the threshold level, a taylor expansion of the denominator for $|E|^2 \approx 0$ is done,

\begin{equation}
\dot{E} = (1 + i\alpha)E(\tilde{P} - |E|^2 - 2\tilde{P}|E|^2), \tag{16}
\end{equation}
where we set $|E|^4 \approx 0$ as a neglectable term. As we consider a laser operated close to the threshold, it follows that the pump $\hat{P}$ and the intensity $|E|^2$ are of Order $O(\epsilon)$, where $\epsilon$ is a small factor. This holds true only if the input signal $ngI$ is a small electrical injection. After applying this the equation is given by

$$\dot{E} = (1 + i \alpha)E(\hat{P} - |E|^2).$$  \hspace{1cm} (17)

We can substitute $\hat{P} = P + \eta g I$ back into the equation, change the rotating frame of the laser by setting $E = Ze^{-i(\omega t - \alpha P)t}$ and introduce a complex factor $\gamma = -(1 + i \alpha)$ that scales the nonlinearity

$$\dot{Z} = Z(P + \eta g I + i \omega + |Z|^2),$$  \hspace{1cm} (18)

By adding feedback $\kappa e^{\phi}Z(t - \tau)$ to the system one arrives at Eq. (10).

$$\dot{Z} = Z(P + \eta g I + i \omega + |Z|^2) + \kappa e^{\phi}Z(t - \tau).$$  \hspace{1cm} (19)

References

1. Jaeger H. The ‘echo state’ approach to analysing and training recurrent neural networks. GMD Report 148 GMD - German National Research Institute for Computer Science. 2001. https://doi.org/10.1007/3-540-44342-9_63. 2003.

2. Maass W, Natschläger T, Markram H. Neural Comp. 2002;14:2531. https://doi.org/10.1162/08997660276047955.

3. Hochreiter S. Int J Uncert Fuzz Knowl-Based Syst. 1998;6:1017. https://doi.org/10.1142/S0218488598000094.

4. Fernando C, Sojakka S. In Advances in artificial life, pp 588–97. https://doi.org/10.1007/978-3-540-39432-7_63. 2003.

5. Antonik P, Duport F, Hermans M, Smerieri A, Haelterman M, Massar S. IEEE Trans Neural Netw Learn Syst, 28(11). 2016. https://doi.org/10.1109/tnnls.2016.2598655.

6. Dockendorf K, Park I, He P, Principe JC, DeMarse TB. Biosystems, 95(2). 2009. https://doi.org/10.1016/j.biosystems.2008.08.001.

7. Bauduin M, Smerieri A, Massar S, E. Horlin. 2015 IEEE 81st Vehicular Technology Conference (VTC Spring). 2015.

8. Keuninckx L, Danckaert J, Van der Sande G. Cogn Comput 9(3). 2017. https://doi.org/10.1007/s12559-017-9457-5.

9. Scardapane S, Uncini A. Cogn Comput. 2017:9:125–135. https://doi.org/10.1007/s12559-016-9439-z.

10. Argyris A, Bueno J, Soriano MC, Fischer I. In 2017 Conf. on Lasers and Electro-optics Europe Europe Quantum Electronics Conference (CLEO/Europe-QEC), p 1. https://doi.org/10.1364/cleoe-eqec.2017.8086463. 2017.

11. Amil P, Soriano MC, Masoller C. Chaos: Interdiscip J Nonlin Sci, 2019;29(11):13111. https://doi.org/10.1063/1.5120755.

12. Cunillera A, Soriano MC, Fischer I. Chaos: Interdiscip J Nonlin Sci, 2019;29(11):13113. https://doi.org/10.1063/1.5120822.

13. Larger L, Soriano M, Brunner D, Appeltant L, Gutierrez JM, Pesquera L, Mirasso CR, Fischer I. Opt Express. 2012;20(3):3241. https://doi.org/10.1364/oe.2012.003241.

14. Paquet Y, Duport F, Smerieri A, Dambre J, Schrauwen B, Haelterman M, Massar S. Sci Rep 2(287). 2012. https://doi.org/10.1038/srep00287.

15. Brunner D, Soriano M, Mirasso CR, Fischer I. Nat Commun. 2013:4:1364. https://doi.org/10.1038/ncomms2368.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.