

RESEARCH ARTICLE | NOVEMBER 04 2025

## The effect of the non-linear function on system dynamics within delay-feedback reservoirs

Special Collection: [Nonlinear Dynamics of Reservoir Computing: Theory, Realization and Application](#)

Alexander C. McDonnell ; Martin A. Trefzer 



Chaos 35, 113107 (2025)

<https://doi.org/10.1063/5.0286757>



View  
Online



Export  
Citation

### Articles You May Be Interested In

Investigation of determinism in heart rate variability

*Chaos* (June 2000)

Organization and identification of solutions in the time-delayed Mackey-Glass model

*Chaos* (April 2015)

A double-cycle echo state network topology for time series prediction

*Chaos* (September 2023)



Chaos

## Special Topics Open for Submissions

[Learn More](#)

# The effect of the non-linear function on system dynamics within delay-feedback reservoirs

Cite as: Chaos 35, 113107 (2025); doi: 10.1063/5.0286757

Submitted: 20 June 2025 · Accepted: 16 October 2025 ·

Published Online: 4 November 2025



View Online



Export Citation



CrossMark

Alexander C. McDonnell<sup>a)</sup> and Martin A. Trefzer

## AFFILIATIONS

School of Physics, Engineering and Technology, University of York, York, United Kingdom

**Note:** This paper is part of the Special Topic on Nonlinear Dynamics of Reservoir Computing: Theory, Application and Realization.

<sup>a)</sup> Author to whom correspondence should be addressed: alexander.mcdonnell@york.ac.uk

## ABSTRACT

Delay-feedback reservoirs are a subset of reservoir computers characterized by a hardware-efficient architecture that trades spatial complexity for temporal processing. It employs a single non-linear node, a delay line, and a time-multiplexed input signal to generate a network of “virtual nodes,” effectively emulating a larger spatial neural network. One of the most powerful aspects of delay-feedback reservoirs is their versatility. Our previous work found that the non-linear node performs two mathematical functions, a non-linear transform and integration. The non-linear transform can be represented by any number of non-linear functions, making it difficult to optimize a delay-feedback reservoir to solve a specific computational task. This work explores different non-linear functions in order to determine their effect on the dynamics of the reservoir, in order to provide insight into this optimization problem. Five different non-linear functions are compared in terms of performance, metrics, and utilization: Mackey–Glass, sine squared, double sinusoids, *Tan*, and *Tanh*. Our results find that the Mackey–Glass non-linear function shows limited system dynamics, performing well on non-linear tasks but performing poorly on memory intensive tasks. We then demonstrate the distinct system dynamics within the other four non-linear functions. We found that sine squared shows limited overall performance, double sinusoid performs well in non-linear tasks, *Tan* resembles an odd valued exponent Mackey–Glass reservoir but with greater parameter sensitivity, and *tanh* offers balanced performance across both task types. We find that modifying the system dynamics of a reservoir is an important step toward optimizing a delay-feedback reservoir for specific computational tasks.

© 2025 Author(s). All article content, except where otherwise noted, is licensed under a Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>). <https://doi.org/10.1063/5.0286757>

Traditional computing methods are increasingly challenged by real-time computational problems, primarily because modern systems often lack the bandwidth and processing power needed for complex real-time tasks. An alternative solution is the delay-feedback reservoir, a machine learning paradigm inspired from neurobiology that has proven effective in solving real-time computational problems while offering greater hardware compatibility compared to other reservoir models. While delay-feedback reservoirs are often easier to implement within hardware, the design of a delay-feedback reservoir is complex, often leading to inefficient use of the reservoir. This work aims to improve efficiency by enabling the selection and optimization of the non-linear function within a delay-feedback reservoir for a specific computational task, allowing for more efficient reservoir systems.

## I. INTRODUCTION

With traditional computing paradigms approaching their limit, alternative paradigms need to be explored to keep up with modern computing demands. An alternative approach is to look toward biological systems, which exhibit the speed and robustness that are desirable within modern computing systems. A promising computational paradigm is reservoir computing, inspired from recurrent neural networks,<sup>1,2</sup> which exploits the natural dynamical behavior of a particular substrate in order to perform computation, greatly improving efficiency compared to traditional computing paradigms. This allows for a quick and efficient method of training a reservoir computing system by only considering the dynamical substrate as a “black box,” and only applying weights on the output of the system. This has led to the reservoir computing framework being applied to a wide range of systems, from traditional hardware, such as field

programmable gate arrays (FPGAs) and analog circuitry,<sup>3,4</sup> to non-typical systems, such as memristors and even a bucket of water.<sup>5,6</sup>

While reservoir computing has been shown to provide excellent performance when processing real-time data, typical implementations require a large amount of randomly connected recurrent neurons. This greatly increases the space required within hardware due to the neurons, their interconnectivity, and I/O,<sup>7</sup> making translation into hardware challenging; in order to minimize the space required within hardware, an alternative approach called delay-feedback reservoir computing<sup>8</sup> can be used.

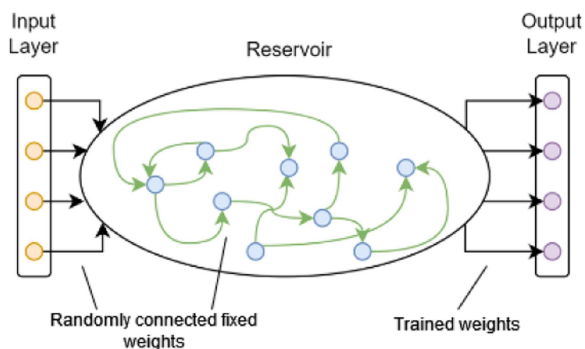
A delay-feedback reservoir has the advantage of having only one non-linear neuron (also known as a non-linear node) and a time delay to create many “virtual neurons,” also known as virtual nodes, allowing it to effectively emulate a much larger spatial recurrent neural network.<sup>9</sup> Multiple interesting physical implementations of delay-feedback reservoirs have been developed, utilizing both commercial components like FPGAs and op-amps,<sup>4,10</sup> as well as specialized designs based on photonic and optoelectronic methods.<sup>11,12</sup> The delay-feedback reservoir has allowed reservoir computing to become easier to realize within hardware, while providing excellent results when processing time series prediction and other temporal tasks.

Another powerful feature of using a delay-feedback reservoir is that they are highly configurable. Our previous work found that the non-linear node within a delay-feedback reservoir can be broken down into two key components:<sup>13</sup> a non-linear function, which provides the non-linear transform of input data, and an integrator, which integrates and mixes information to create memory. We found that the integrator influences the timescale of the reservoir system and its connectivity between virtual nodes,<sup>14</sup> but the impact of the non-linear function on system dynamics remains largely unexplored. To assess the impact of the non-linear function on the reservoir system, we begin by analyzing a traditional Mackey–Glass based delay-feedback reservoir, evaluating how variations in the Mackey–Glass exponent affect the reservoir in three distinct ways. First, we evaluate the performance of the reservoir running two computational benchmark tasks. Here, we choose NARMA-10 and Santa Fe, as they require vastly different and opposite reservoir characteristics for optimal computational performance.<sup>15–17</sup> We then determine the characteristics of the reservoir in terms of system metrics: generalization rank (GR), kernel quality (KQ), and linear memory capacity (LMC). Finally, the utilization of the non-linear function is analyzed by generating a heat-map of its input–output relationship, which reveals the regions of the function actively used during computation. We then explore alternative non-linear functions and compare them to the typical Mackey–Glass based delay-feedback reservoir by once again evaluating the reservoir in terms of its performance, metrics, and utilization. This work aims to provide insights into optimizing a delay-feedback reservoir for specific computational tasks by modifying the non-linear function.

## II. RESERVOIR COMPUTING

### A. The reservoir computing framework

The reservoir computing framework is a biologically inspired machine learning paradigm that utilizes the natural dynamical

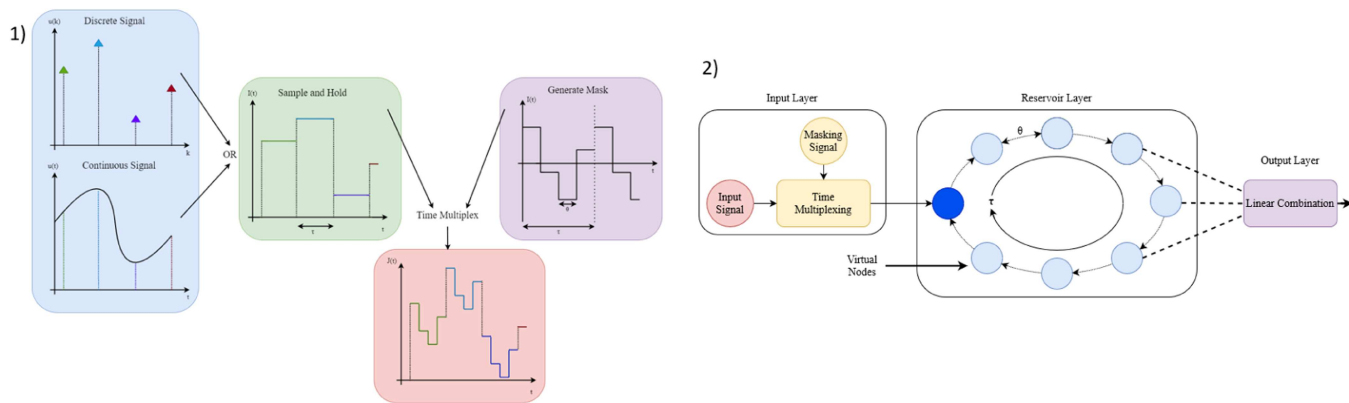


**FIG. 1.** The typical structure of a reservoir computer. It consists of three layers: an input layer, which is randomly connected to the reservoir layer with random fixed connections; a reservoir layer, which contains randomly connected recurrent neurons with randomly generated weighted connections; an output layer, which is randomly connected to the reservoir layer but has weights that are trainable.

behavior of a given system to perform computation. This framework has become a popular method of training input-driven dynamical systems as only the output weights of the system have to be calculated, which can be done inexpensively with linear regression based training methods. A typical reservoir computing system consists of three layers: reservoir, input, and output. The reservoir layer is a dynamical black box, typically modeled as a set of recurrent neurons connected by fixed random weights, which exhibits the following dynamical properties: a non-linear transform of the input signal, being able to represent the inputs in a high-dimensional state space, and to have a short-term memory of the previous inputs of the system. In order to produce a response from the reservoir layer, inputs are randomly connected to the reservoir layer with fixed random-weighted connections to create an input layer, perturbing the reservoir layer to generate a dynamically rich response. The output of the system is observed through trainable randomly connected weights to the reservoir layer, creating an output layer that can be calculated through a simple training algorithms, such as ridge regression or Moore–Penrose pseudoinverse. A typical structure of a reservoir computing system is depicted within Fig. 1.

### B. Delay-feedback reservoir computing

First demonstrated by Appeltant in 2011,<sup>8</sup> the delay-feedback reservoir takes advantage of delay-systems theory to emulate a much larger spatial recurrent neural network with only a single non-linear neuron and a time delay.<sup>9</sup> These emulated neurons, called virtual nodes, are created by time-multiplexing the input signal with a higher frequency masking signal. This gives the advantage over other types of reservoir computing methods by providing a more hardware-friendly and compact design over traditional reservoir computing systems, which typically require a reservoir layer consisting of many randomly connected recurrent neurons. The structure of a delay-feedback reservoir is similar to conventional reservoir computing systems, comprising of three layers: input, reservoir, and output.



**FIG. 2.** (1) The input signal, continuous  $u(t)$  or discrete  $u(k)$ , undergoes a sample-and-hold operation to create  $I(t)$ . This is then time-multiplexed by a masking signal,  $m(t)$ , to create the input sequence,  $J(t)$ . This input sequence is then fed into the non-linear node. (2) The typical structure of a delay-feedback reservoir. A single non-linear neuron (dark blue) is used with a time delay to create a network of connected virtual nodes (light blue). The total time delay is notated as  $\tau$ , which is often the same as the time period of the input signal. The time period of the masking signal, notated as  $\theta$ , defines the spacing between the virtual nodes. The number of virtual nodes,  $N$ , can be calculated by  $\frac{\tau}{\theta}$ .

Within input layer, the input signal, continuous  $u(t)$  or discrete  $u(k)$ , undergoes a sample-and-hold operation of duration  $\theta$  to create  $I(t)$ . This signal is then time-multiplexed by a masking signal,  $m(t)$ , to create the input sequence,  $J(t)$ , which is the input to the non-linear node. The time period of the sample-and-hold operation, notated as  $\tau$ , that typically defines the time delay within the delay line; this process can be seen within Fig. 2. The number of virtual nodes,  $N$ , within the delay-feedback reservoir system can be calculated by

$$N = \frac{\tau}{\theta}. \quad (1)$$

The reservoir layer contains a single non-linear neuron and a time delay, notated as  $\tau$  within this work. Within our previous work, we found that the non-linear neuron can be separated into two key parts: a non-linear function and an integrator.<sup>13</sup> This greatly increases the versatility of the delay-feedback reservoir as it allows any non-linear function to be used with an integrator to provide computation.

The output layer of the system collates the states of the delay-feedback reservoir system by linear combination. Linear combination is where the output layer collects the states within each of the virtual nodes, multiplies each node by a particular weight, and sums all the virtual nodes together within a single time step. This can be expressed as

$$\hat{y}_k = \sum_{i=1}^N w_i x \left( k\tau - \frac{\tau}{N}(N-i) \right), \quad (2)$$

where  $w_i$  is the weight attributed to each virtual node  $i$ ,  $x$  is the output of the non-linear neuron at a given state, and  $\hat{y}$  is the approximated output signal. Figure 2 shows the typical structure of a delay-feedback reservoir network.

### C. The non-linear function

A desired property of a reservoir computing substrate is to provide a non-linear transform of the input stimuli. This non-linear transform serves two purposes within a system: the ability to map inputs to a high-dimensional non-linear space and to constrain information within an attractor region.

If a system was purely linear, it would not be able to solve non-linear problems due to a linear mapping of the inputs into the state space of the neural network, as found within early artificial neural networks.<sup>18</sup> The non-linear part of the reservoir layer allows a network to learn non-linear behaviors as inputs are mapped into a high-dimensional, non-linearly transformed state space within the network.

Another important feature that non-linearity provides is the ability to constrain information within a specific operating range by acting as an attractor. This can greatly help stabilize a system by mapping very large or small information values, which may have been produced by other components within a network, to values centered around zero. This is particularly useful within physical systems as there are often constraints, such as a maximum or minimum voltage, which could saturate or even cause damage to a system.<sup>5,19–21</sup>

Unfortunately, a strong non-linear transform often comes with a cost of a reduction in linear memory capacity, as a strong non-linear transform weakens the connections between virtual nodes. As different computational tasks require different computational characteristics, it is often a trade-off between the characteristics of a system to exhibit enough memory and non-linear transform to reliably compute a given task.<sup>22</sup>

As previously discussed, an ability of the delay-feedback reservoir is to be able to separate the non-linear node into its functional parts, a non-linear function and an integrator, allowing any non-linear function to be used. As our previous work has examined the effect of the integrator stage on the characteristics of a

delay-feedback reservoir, a new question arises: what effects do different non-linear functions have on the characteristics of such systems and can alternative non-linear functions be employed to mitigate the trade-off between linear memory and non-linear transformation? Within this work, the Mackey–Glass non-linear function and several alternative non-linear functions will be investigated so that the effects of the non-linear function can be evaluated in terms of computational performance and system dynamics.

### III. RESERVOIR EVALUATION

#### A. Benchmarks

To assess the performance of a reservoir computing system, computational benchmarks can be employed to evaluate its capabilities and enable comparisons between different systems. Among the various types of benchmarks available, we focus on two temporal benchmarks: the NARMA-10 benchmark, which demands high memory capacity but low non-linearity and dimensionality; and the Santa Fe Laser dataset, which presents the opposite challenge by requiring low memory capacity but high non-linearity and dimensionality.

**NARMA-10.** The non-linear autoregressive moving average task, or NARMA, is an imitation computational benchmark that is widely used to evaluate the performance of neural and reservoir networks.<sup>23</sup> The challenge of the benchmark is that it requires both (weak) non-linearity and a long-term dependency on previous input stimuli (memory) to evaluate the dynamical capability of a reservoir. The discrete-time NARMA- $n$  benchmark is  $n$ th order, meaning it relies on memory of  $n$  previous inputs from a lag of  $n$  time steps. NARMA-10, in particular, is of the tenth order, meaning it relies on inputs from a lag of ten time steps. The equation for the discrete NARMA-10 sequence is

$$y_{n+1} = 0.3y_n + 0.05y_n \left( \sum_{i=0}^9 y_{n-i} \right) + 1.5u_n u_{n-9} + 0.1. \quad (3)$$

When using the NARMA-10 benchmark with a reservoir computing system, the objective is to train the system to replicate the dynamics of the NARMA-10 signal in response to the same input sequence. The performance is then evaluated by comparing the system's trained output with the target output generated by the NARMA-10 equation.

**Santa Fe Laser (A).** The Santa Fe Laser dataset A, also known as Santa Fe or Laser, is a predictive, highly non-linear computational benchmark taken from observing a far-infrared laser within a chaotic state.<sup>15</sup> The goal of the Santa Fe benchmark is to train the reservoir system to learn the dynamics of the dataset in order to predict the next  $n$  observations. In practice, the task often focuses on predicting only the next observation, which minimizes the reliance on previous input stimuli. As a result, this benchmark emphasizes the need for strong non-linear, high-dimensional processing capabilities in the reservoir system, rather than a strong fading memory.

#### B. System metrics

Applying different computational benchmarks to a reservoir system often gives different results. While benchmarks are valuable for assessing the performance of a reservoir on specific tasks, they provide limited insight into the underlying dynamical behavior of the system. To enable a more accurate and task-independent evaluation, a set of metrics can be computed that characterizes the reservoir itself.

##### 1. Linear memory capacity

The linear memory capacity, or LMC, of a system is a measure of how well previous input stimuli can be recalled from a reservoir system. In terms of the LMC, it is a measure of how long a particular input can be stored within the reservoir before it degrades. This is an extremely useful tool as it is often unclear why a particular task performs poorly on a given reservoir system; measuring the LMC gives some indication of whether there is sufficient memory available within the system for a particular task.

The LMC of a system is calculated by injecting a random uniform distribution of numbers into the reservoir and then training the output to recover the previous inputs  $u(k-i)$ , for  $i = 1, 2, 3, \dots, 2N$ , where  $N$  is the number of nodes within the reservoir, resulting in  $i$  outputs. The LMC is then measured by calculating the variance between the output of the reservoir and the delayed input, summed over all delays; the maximum memory capacity of a system is always  $MC \leq N$ .<sup>24</sup> The equation for calculating LMC can be expressed as

$$MC = \sum_{i=1}^{2N} \frac{\text{cov}^2(u(k-i), y(k))}{\sigma^2(u(k)) \sigma^2(y(k))}. \quad (4)$$

##### 2. Kernel quality

The kernel quality, or KQ, is a measure of how well a reservoir system can create a non-linear representation of different input streams. This can be viewed as how much dimensionality the system has, or how well the system is able to separate distinct input patterns. This measurement helps quantify the heterogeneity of non-linear operations a system can perform, allowing the states of the reservoir to be linearly separable.<sup>25</sup> The KQ of a reservoir system can be calculated by the following steps:

1. Generate  $m$  (ideally greater than the amount of nodes within the reservoir system) number of random input vectors, each with a length of  $k$ ;  $U = [u_1, u_2, u_3, \dots, u_m]$ , where  $u_i$  is a series of  $k$  random data points.
2. Inject the set of input vectors,  $U$ , into the reservoir system.
3. As the  $k$  data points are injected into the reservoir, collect the states of the reservoir system, creating a matrix of size  $n \times m$ .
4. Compute the singular values of the state matrix using singular value decomposition.
5. The KQ is equal to the non-zero elements within the singular value matrix.

The maximum value of the KQ is  $KQ \leq N$ , which implies that all of the injected set of input vectors have been independently

mapped within the reservoir. The closer KQ is to  $N$ , the better the performance of the reservoir system will be.

### 3. Generalization rank

The generalization rank, or GR, of a reservoir system is a measure of how well the system is able to generalize similar input streams. In a real system, there are often external factors that degrade the input signal quality, such as noise. The GR allows some insight into how sensitive a reservoir system is and how well it can handle small variations within its input. Calculating the GR of a reservoir system is similar to calculating KQ, the steps are as follows:

1. Generate a single set of input vectors, notated as  $u$ , with  $k$  random data points.
2. Create  $m$  copies of  $u$  (ideally greater than the amount of nodes within the reservoir system) and apply a random amount of noise to each data point within each input vector;  $U' = [u'_1, u'_2, u'_3, \dots, u'_m]$ .
3. Inject the set of input vectors,  $U'$ , into the reservoir system.
4. As the  $k$  data points are injected into the reservoir, collect the states of the reservoir system, creating a matrix of size  $n \times m$ .
5. Compute the singular values of the state matrix using singular value decomposition.
6. The GR is equal to the non-zero elements within the singular value matrix.

The maximum value the GR is  $GR \leq N$ , but unlike the KQ, this implies poor performance as every noisy input has been classified differently. An ideal GR is that of 1, where the system is able to generalize all input vectors; the closer the GR is to 1, the more robust the reservoir will be at handling input signal variances.

### C. Evaluating reservoir computing

To compare the performance of a reservoir across different experimental runs and benchmark tasks, a quantifiable performance metric is required. In this case, we compute the error between the reservoir's trained output and the target output, and normalize it using a relevant statistical measure. This results in the Normalized Root Mean Square Error (NRMSE), a widely used metric for evaluating both training and testing errors. The NRMSE is expressed as

$$NRMSE = \sqrt{\frac{1}{m} \sum_{k=1}^m (\hat{y}_k - y_k)^2} \frac{1}{\sigma(y_k)}, \quad (5)$$

where  $m$  is the number of data samples within the experiment,  $\hat{y}_k$  is the trained reservoir output,  $y_k$  is the desired target function, and  $\sigma$  is the standard deviation.

An NRMSE value of 0 indicates a perfect match between the reservoir's output and the target function, while a value of 1 suggests that the reservoir is merely predicting the mean of the target output. In general, a lower NRMSE signifies better performance of the reservoir on the given computational task.

## IV. THE MACKEY-GLASS BASED DELAY-FEEDBACK RESERVOIR

The Mackey-Glass model is a first-order delay-differential equation created by Michael Mackey and Leon Glass to model the respiratory system and hematopoietic diseases, which are biological functions that are subject to a time delay.<sup>26</sup>

Although initially used to model biological processes, the Mackey-Glass system has been shown to exhibit a variety of rich dynamical behaviors, such as strong non-linearity and high dimensionality, making it ideal for hardware translation.<sup>4,27,28</sup> The Mackey-Glass model was first applied to a DFRC system by Appeltant within their original DFRC work.<sup>29</sup> The Mackey-Glass equation was slightly simplified to facilitate its implementation in hardware; the modified Mackey-Glass equation is given as

$$\dot{x}(t) = \frac{\beta (x(t - \tau) + \delta I_t)}{1 + ((x(t - \tau) + \delta I_t)^n)} - \gamma x(t), \quad (6)$$

where  $I(t)$  is the pre-masked input signal,  $x(t)$  is the reservoir state vector,  $\dot{x}(t)$  is the derivative of the reservoir state vector,  $x(t - \tau)$  is the delayed reservoir output,  $\beta$  is the coupling gain between the non-linear function and integrator,  $\gamma$  is the feedback strength for the leaky integrator,  $\delta$  is an input scaling factor, and  $n$  is the non-linearity factor.

The Mackey-Glass time delay-differential equation can be separated into two parts, the Mackey-Glass non-linear function and a leaky integrator. This can be expressed as

$$x(t) = \frac{\beta (x(t - \tau) + \delta I_t)}{1 + ((x(t - \tau) + \delta I_t)^n)}, \quad (7)$$

$$\dot{x}(t) = -\gamma x(t).$$

This makes the Mackey-Glass based reservoir an ideal candidate for investigating the effect of the non-linear function as it can be investigated independently of the leaky integrator.

### A. The Mackey-Glass non-linear function

While the Mackey-Glass based reservoir has been widely used, research has focused on applying the Mackey-Glass equation within hardware or simulation models, with the Mackey-Glass exponent assumed to be an approximate level of the degree of non-linearity the function has; the effect that the strength of the non-linearity has on computational performance and system dynamics has not been investigated.

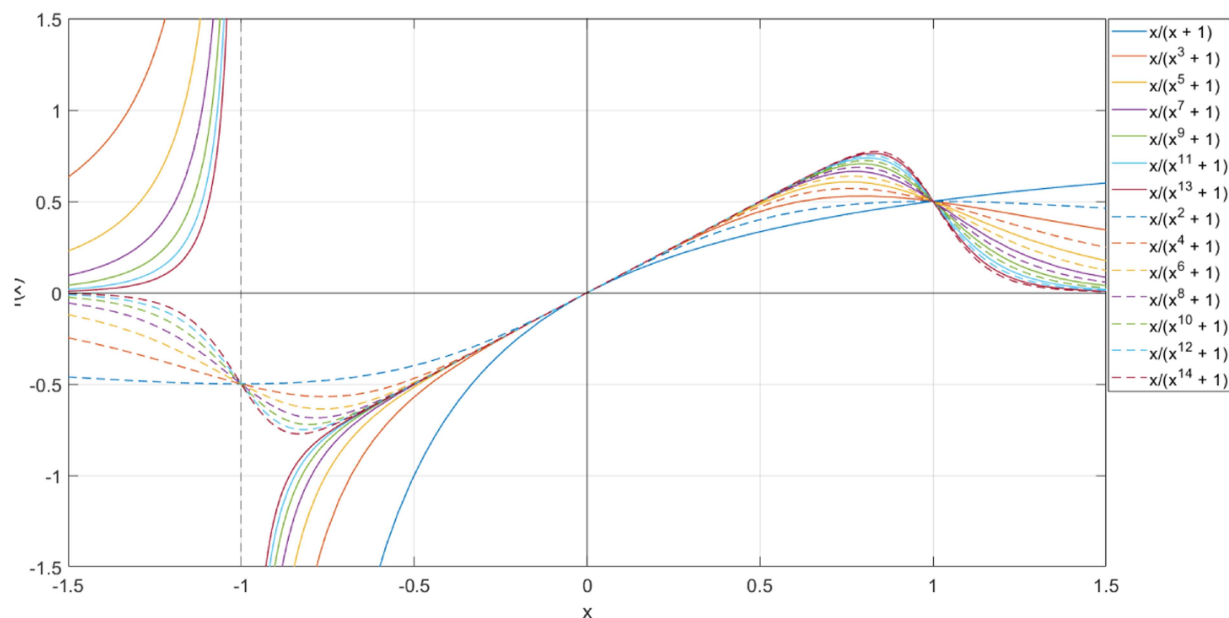
In the computational model, the Mackey-Glass non-linear function is defined by the following equation:

$$f(x) = \frac{x}{x^n + 1}, \quad (8)$$

where  $n$  is the strength of the non-linearity of the equation.

To understand the effect that  $n$  has on the shape of the non-linearity, Eq. (8) is plotted with several values of  $n$ ; this is shown in Fig. 3.

The graph in Fig. 3 shows several parameter sweeps of the Mackey-Glass non-linear function where the exponent  $n$  is set between 1 and 14. This figure shows three key features that the exponent has on the shape on the non-linear function: the function has



**FIG. 3.** A parameter sweep of different values of  $n$  within the Mackey–Glass non-linear equation. The value of  $n$  is swept between 1 and 14. Odd values (solid lines) of  $n$  can be seen within the graph with an asymptote at  $x = -1$ , while even values (dashed lines) of  $n$  are  $y = x$  symmetrical.

a linear region of unity gradient centered around the origin that becomes more linear as the value of the exponent increases, the function has an asymptote at  $x = -1$  for odd values of the exponent, and when considering the function within the feedback loop of the delay-feedback reservoir, the attractor behavior it will exhibit.

For small values of  $n$ , the shape of the function is non-linear with a strong attractor behavior where  $x$  is positive. However, for negative values of  $x$ , the function becomes asymptotic, mapping negative  $x$  values to positive  $y$  where  $x < -1$ . As the value of  $n$  increases, the linear region around the origin becomes wider. This reduces the attractor range as more of the function is mapped to  $y = x$ , but the attractor behavior becomes stronger outside the linear range.

Another feature of the Mackey–Glass non-linear function is that for odd values of  $n$ , the function becomes asymptotic as  $x = -1$ . When  $n$  is even, the function becomes odd, mapping  $f(-x) = -f(x)$ . This means that when negative values exist within the Mackey–Glass delay-feedback reservoir system, they are simply mapped the same way the positive values are. However, when  $n$  is odd, negative values, which are  $x > -1$ , are drawn toward the asymptote, then mapped positive when  $x < -1$ , causing negative values to be mapped to positive within an odd  $n$  system.

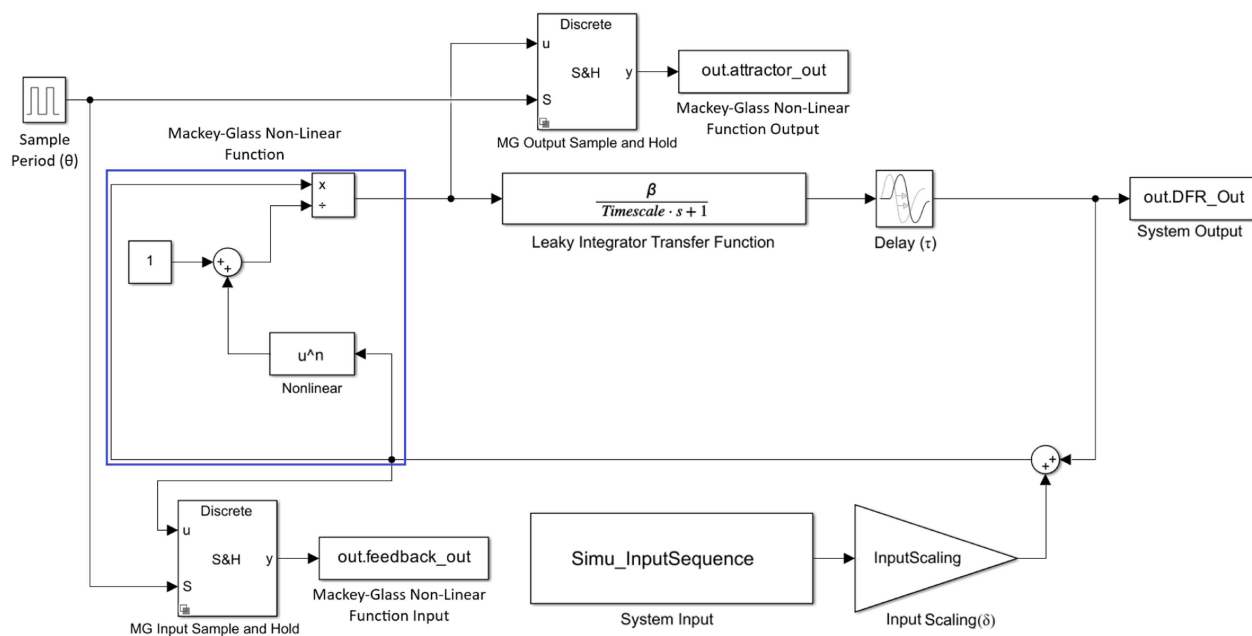
Therefore, it can be said that for an even valued  $n$  system, the attractor behavior maps values outside of the linear range of the function toward zero, whereas for an odd valued  $n$  system, the attractor maps positive and large negative values toward zero, and maps small negative values toward positive values when outside of the linear range of the function.

## B. Simulation model

A Mackey–Glass based delay-feedback reservoir is created within Simulink 23.2, a MathWorks simulation tool (MATLAB 2023b).<sup>30</sup> The input and output layers of the DFRC system are implemented in MATLAB. The input data is generated within the MATLAB workspace and injected into the Simulink model, from where the output of the DFRC system is returned into the MATLAB workspace for reservoir training and evaluation. The model utilized is illustrated in Fig. 4.

The Mackey–Glass non-linear function is built from Simulink numeric model blocks, as shown within the blue rectangle within Fig. 4. The non-linearity factor can be modified by changing the  $n$  parameter within the “non-linear” block. Although the integrator is typically modeled as a leaky integrator, in this case, we represent it using an equivalent first-order system. The first-order system approach is not only easier to represent in a real system, but it also enables the use of a tunable time constant that can be kept consistent across all experiments. The delay is implemented using the Simulink “Transport Delay” block, which adds a signal delay of time length  $\tau$ .

To evaluate the utilization of the Mackey–Glass non-linear function, both its input and output are sampled using a sample-and-hold mechanism. This is implemented via the Simulink “Sample and Hold” block, which captures the data at a sampling interval of  $\theta$ . The output of the sample and hold functions are returned to the MATLAB workspace using the two “To Workspace” blocks, named “out.feedback\_out” to record the input of the non-linear function and “out.attractor\_out” to record the output of the non-linear function. The clock of the sample and hold blocks



**FIG. 4.** A schematic of a delay-feedback reservoir, using the Mackey–Glass dynamical system as a non-linear function, and a first-order transfer function as the integration stage, enabled to record the input and output of the Mackey–Glass non-linear equation, using a sample and hold function, at a sample period of  $\theta$ . Model created within Simulink 23.2.

are driven using a Simulink “Pulse Generator,” which generates a clock with a duty cycle of 50%, with a period of  $\theta$ .

The input signal, generated within MATLAB, is injected into the reservoir using a “From Workspace” block, named “Simu\_InputSequence.” The states of the reservoir are read from the model using a “To Workspace” block, named “out.DFR\_Out,” at a sample rate of  $\theta$ ; this allows the states of each virtual node to be sampled at the correct time. The captured states are then transferred to the MATLAB workspace, where training and system evaluation are performed.

### C. Methodology

Using the experimental platform created, the effect that the value of the exponent within the Mackey–Glass non-linear equation has on the performance, dynamics, and utilization of a delay-feedback reservoir system is investigated. The values of the Mackey–Glass exponent,  $n$ , are tested at 1, 2, 5, 6, 9, and 10. These values were chosen as they exhibit the most significant changes within the function shape, as seen in Fig. 3, and the odd and even pairs are chosen to allow comparison of the different attractor behaviors. A 200-node reservoir is given a  $\tau$  of 80 s and a timescale of 400 ms, as this timescale was shown to provide the best results for all benchmarks within our previous work.<sup>14</sup> As the input scaling and coupling gain have a large effect on the computational performance of the reservoir, a parameter sweep of both the input scaling and coupling gain is performed for the NARMA-10 and Santa Fe benchmarks and system metrics. Table I shows the model parameter values and ranges for the proposed parameter sweep.

The procedure for generating the input sequences of both the NARMA-10 and Santa Fe benchmarks within MATLAB are the same. A sequence of 6000 data points ranging from  $[-1, 1]$ , either generated from the NARMA-10 equation or copied from the Santa Fe dataset is time-multiplexed by a random-weighted no-offset mask. A washout period (consisting of 100 data points) is used, with the training and test datasets being split 80/20, respectively. Training is performing using the Moore–Penrose pseudo-inverse algorithm.<sup>31</sup>

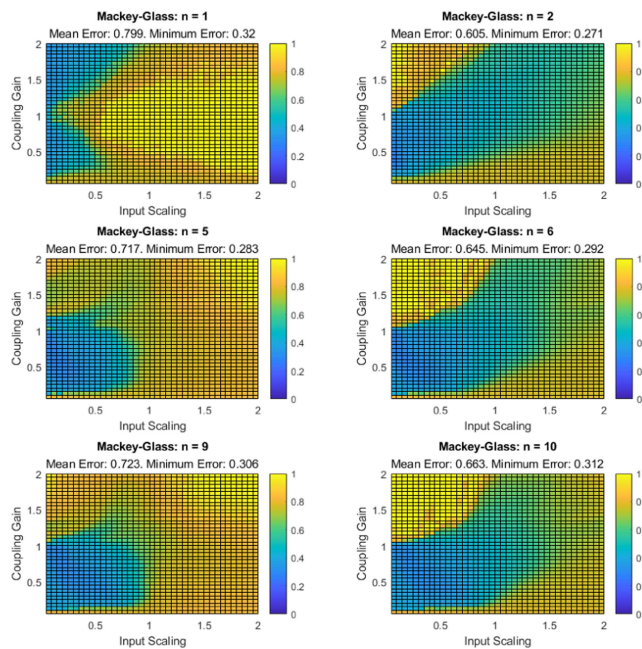
## V. MACKEY-GLASS EXPERIMENTS

### A. Mackey-Glass system parameter sweeps

**NARMA-10.** Figure 5 shows the effects of the input scaling and coupling gain on a 200-node delay-feedback reservoir system, evaluating the NARMA-10 computational benchmark, at six different Mackey–Glass exponent values,  $n$ . The parameter sweeps display the

**TABLE I.** Parameter values of the delay-feedback reservoir Simulink model during the input scaling and coupling gain parameter sweep.

| Name                  | Symbol   | Value  |
|-----------------------|----------|--------|
| Coupling gain         | $\beta$  | 0–2    |
| Input scaling factor  | $\delta$ | 0–2    |
| System timescale      | T        | 400 ms |
| Time delay            | $\tau$   | 80     |
| Masking signal period | $\theta$ | 0.4    |



**FIG. 5.** Graphs showing the NRMSE values of a parameter sweep for a 200-node system running the NARMA-10 benchmark, at six different Mackey–Glass exponent values, with the input scaling and coupling gain ranging between 0.05 and 2.

NRMSE of the reservoir system at each sweep point as a color spectrum; the lowest calculated NRMSE value is shown as blue, with the highest shown as yellow. As a system may be drawn into saturation or become unstable, the maximum NRMSE value within all of the parameter sweeps is set to 1.

When the Mackey–Glass exponent is set to 1, several observations can be made. First, the region of best performance for the NARMA-10 benchmark is small and shows a very high sensitivity to input scaling, showing a reasonable NRMSE value for an input scaling of only 0.05–0.75, and a low sensitivity to the coupling gain. Second, the overall performance of the system is poor, with a maximum NRMSE value of approximately 0.3, within this small region of best performance. Third, it is likely that this region of best performance has a strong localized Linear Memory Capacity (LMC) as indicated by the sharp gradient moving from average to poor performance.

A sensitivity to input scaling is to be expected due to the behavior of the attractor. When the Mackey–Glass exponent is set to a value of 1, the Mackey–Glass non-linear equation has a very small linear region and a large weak attractor area, almost all values within the system, except for the very small values, will be transformed and pushed toward zero. When the input scaling is below 1, the signal is attenuated so it is less likely to be affected by the attractor behavior. As the integrator also attenuates, a wider range of coupling gains can be utilized before the system is either negatively affected by the attractor, or before the output of the integrator becomes too large to overpower the attenuated input signal.

Another observation is the difference between the odd and even values of  $n$ . When  $n$  is even, the region of best performance spreads diagonally from the origin and has a larger region of poor operation, spreading to the top left of the parameter space, while for odd values, the shape of the region of best performance is more concentrated within a particular region of the parameter space. This behavior is due to the mapping of values within the negative side of the attractor.

When the coupling gain is high and the input scaling is low, information may be lost if the scaled input signal is considerably smaller than the delayed reservoir output. This effect can be seen in Fig. 5 as a region of poor performance in the top right of the parameter sweeps. An additional effect that occurs within the region is when there is no loss of scaled input data: the signal becomes large enough to be affected by the attractor behavior of the non-linear function.

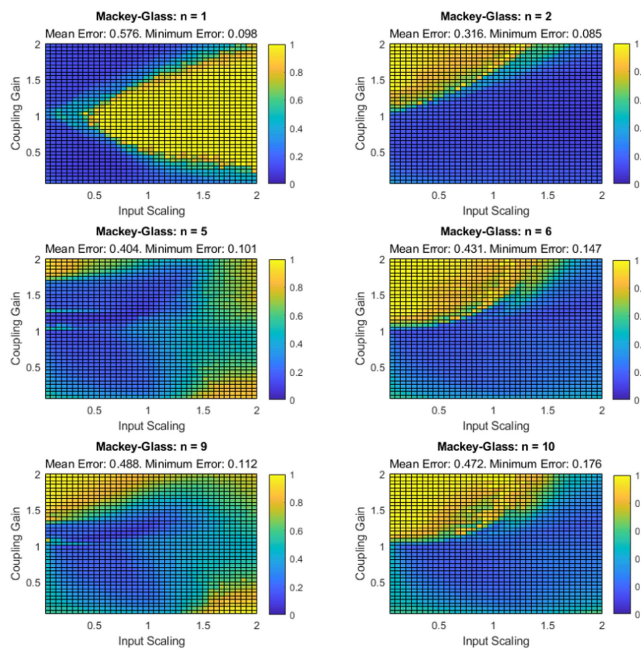
With an odd value of  $n$ , if a signal exceeds the linear range within the non-linear function, then the signal is attenuated toward zero proportional to its magnitude, whereas with even values of  $n$ , only large positive signals are attenuated toward zero, with large negative signals being mapped to attenuated positive values, and small negative signals that exceed the linear range of the non-linear function being mapped to large negative numbers; this is due to the asymptotic behavior of the negative side of the attractor. The mapping of negative values to positive allows for some information to be retained, while with odd values of  $n$ , information would be simply lost.

**Santa Fe.** Figure 6 shows the effects of the input scaling and coupling gain on a 200-node delay-feedback reservoir system, evaluating the Santa Fe computational benchmark, at six different Mackey–Glass exponent values,  $n$ .

One of the most notable differences between the Santa Fe and NARMA-10 benchmarks is the significantly better performance within the Santa Fe benchmark. With the exception of when  $n$  is equal to 1, the performance decreases as  $n$  increases. Given that the Santa Fe benchmark requires strong non-linear dynamics within a reservoir to achieve its best performance, it would be logical to assume that the performance would decrease as the system becomes more linear when  $n$  increases.

When the value of  $n$  is small and even, there is a very wide region of best performance with virtually no sensitivity to input scaling, and a moderate sensitivity to coupling gain. The NRMSE values are small throughout most of the region of best performance, with no localized region of peak performance. As the even value of  $n$  increases, the sensitivity to input gain increases, with the distribution of low NRMSE values becoming more concentrated, creating a noticeable localized region of peak performance around the center of the parameter space. What appears to be task agnostic is the pronounced region of poor operation, located within the top left of the parameter space, which is present when  $n$  is even within the reservoir system running the NARMA-10 computational benchmarks. Given that the region becomes more prominent as the even value of  $n$  increases, this must be a property of the negative side of the attractor.

When the value of  $n$  is above 1 and odd, the reservoir system exhibits two regions of best performance, one in the bottom left and the other in the top left region of the parameter space. Both regions have a low sensitivity to coupling gain and an input scaling sensitivity approximately between 0.05 and 1.2, with the



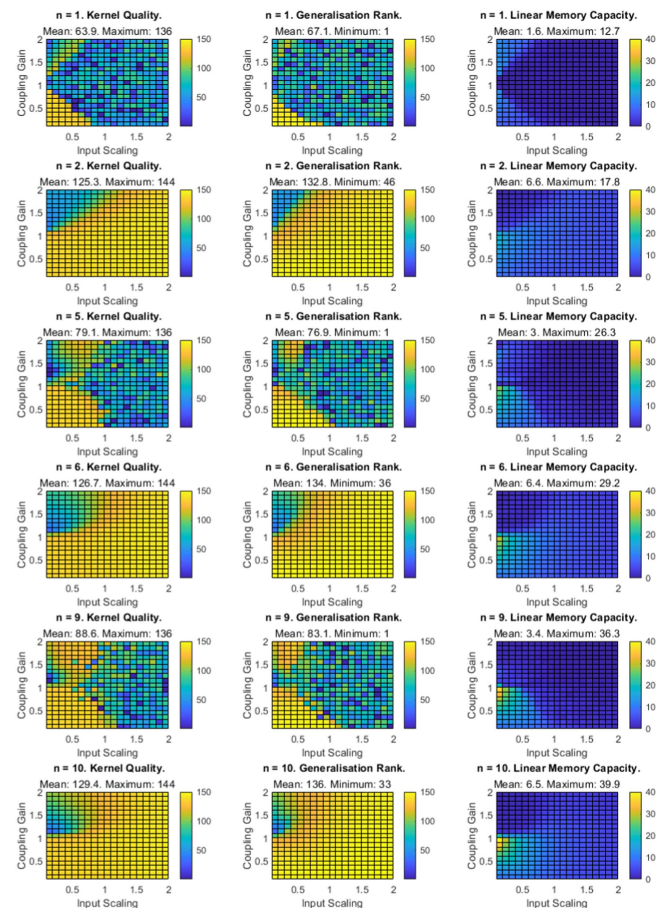
**FIG. 6.** Graphs showing the NRMSE values of a parameter sweep for a 200-node system running the Santa Fe benchmark, at six different Mackey–Glass exponent values, with the input scaling and coupling gain ranging between 0.05 and 2.

best performing NRMSE results within the upper operating region. However, as the odd value of  $n$  increases, the upper region of best performance becomes smaller, which is due to the difference in signal sizes between the scaled input signal and the delayed reservoir output.

When the value of  $n$  is set to 1, the reservoir exhibits a similar region of best performance, as observed within the NARMA-10 reservoir system. Given that the shape of this region is different when  $n$  has increased and is task agnostic, this strongly indicates that there is a region of instability inherent to this non-linearity. This can be explained by referring to the shape of the non-linear function when  $n$  equals 1 in Fig. 3. Within this figure, it can be observed that there is a very small linear region near the origin; therefore, any data exceeding this linear region are affected by the attractor. Given the nature of the negative side of the attractor, any signal that becomes slightly negative is drawn toward the asymptote and then mapped into the positive region. This may take several iterations around the feedback loop as the attractor behavior is weak.

## B. Mackey–Glass system metrics

Figure 7 shows the KQ, GR, and LMC of a 200-node delay-feedback reservoir system during a parameter sweep of the input scaling and coupling gain. This figure shows very different dynamics for the odd and even values of  $n$ . A significant difference is to be expected given the large variations observed throughout the benchmark parameter sweeps.



**FIG. 7.** Graphs showing a parameter sweep for a 200-node system, at six different non-linear Mackey–Glass exponent values, with the input scaling and coupling gain ranging between 0.05 and 2, for the following the system metrics: kernel quality, generalization rank, and linear memory capacity.

For even values of  $n$ , both the KQ and GR exhibit a relatively consistent pattern as the value of  $n$  increases, with the exception of the top left of the parameter space. The value of KQ appears to be high throughout all of the parameter sweeps, showing only a slight sensitivity to coupling gain as the value of  $n$  increases, indicating that the system is able to map input vectors onto a high-dimensional non-linear space. However, the GR parameter sweep shows a minor sensitivity to parameter changes, but indicates that the system's ability to generalize input vectors is poor within the majority of the parameter space, with the exception of the top left area of the parameter sweep. The distribution of the values of KQ and GR is mostly uniform throughout all of the parameter sweeps, with the range between the minimum and maximum values for KQ and GR being 2 and 3, respectively.

In the case of odd values of  $n$ , the parameter plots for KQ and GR appear to be much more sporadic, indicating richer dynamics. These richer dynamics are due to the negative section of the

attractor when  $n$  is odd. The signal mapping of large negative numbers being drawn toward the asymptote, and then mapped into the positive region, gives the system a much stronger non-linear behavior compared to the even values of  $n$ .

The area of poor performance in the top left of the parameter space of the KQ plots was also observed within both the NARMA-10 and Santa Fe benchmark tasks. This region of poor performance, which is due to loss of information from the difference in signal sizes between the scaled input signal and delayed reservoir output, is confirmed within the system metrics as a region where the system is unable to map input vectors onto a high-dimensional non-linear space.

A key observation, for both even and odd values of  $n$ , is that although the shape of the parameter plots does change with an increase of  $n$ , the values of KQ and GR seem to be independent of the value of  $n$ . Initially, this is a surprising result as the Santa Fe benchmark performed significantly better with lower values of  $n$ ; therefore, it was expected that an increase in  $n$  would lead to a lower KQ and a higher GR. However, on reflection, these parameter sweeps show the true behavior of the Mackey–Glass non-linear function. The shape of the Mackey–Glass non-linear function only changes in three cases: when  $n$  is odd, even, or equal to 1. Increasing the  $n$  value increases the width of the linear space around the origin and increases the attractor strength. This means that the non-linear transform is not dependent on the value of  $n$ , only if it is odd or even. The exception to this is when  $n$  is equal to 1, where the shape of the function is inherently unstable; this gives the system different non-linear dynamics compared to when  $n \neq 1$ .

Figure 7 also shows the relationship between the value of the Mackey–Glass exponent and the LMC, showing that as the value of  $n$  increases, so does the maximum memory capacity. This result is to be expected as when the value of  $n$  increases, the wider the linear region within the non-linear function becomes; this leads to more information being passed between virtual nodes as the connection strength increases.

Unlike within the KQ and GR parameter sweeps, the LMC shows a minor difference between odd and even values of  $n$ . For odd values of  $n$ , a second region of memory appears in the center left of the parameter space, but decreases in size as  $n$  increases. However, for even values of  $n$ , there is only a single region of strong LMC, but it appears to be less sensitive to input scaling. This behavior is due to the shape of the non-linear function for odd and even values of  $n$ .

Another observation is that the region of best performing NRMSE values within the NARMA-10 benchmark correlate to the areas with the highest memory within the LMC system metrics. This can be seen by observing the distribution of LMC values within the parameter sweep plots for the even values of  $n$ , which are less sensitive to input scaling and coupling gain than the odd values. This diagonal LMC distribution is visible within the shape of the region of best performance within the NARMA-10 benchmark.

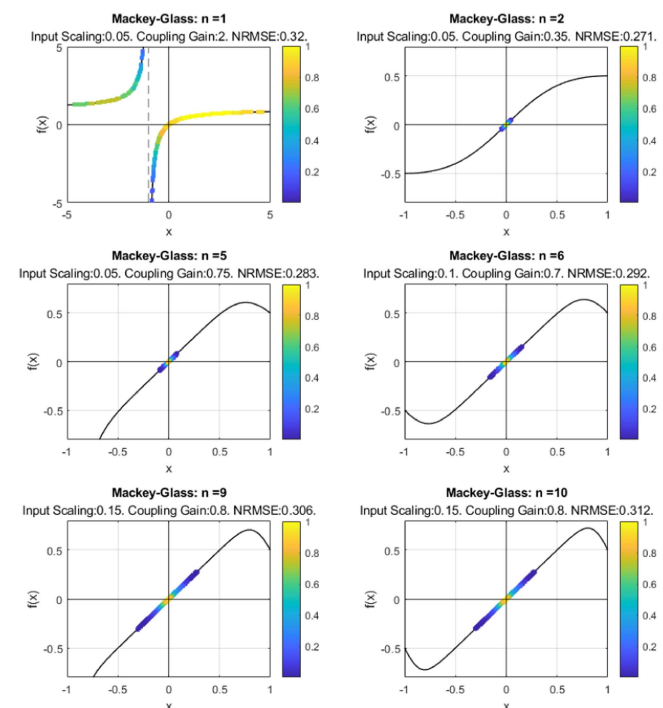
The KQ does have an effect on the NARMA-10 benchmark, as the region of poor KQ performance shown in the top left of the parameter space negatively impacts the performance of the benchmark. With the KQ being so low, the reservoir is unable to map the input vectors onto a high-dimensional non-linear space; therefore, this becomes the dominant metric in terms of performance as the

reservoir system is unable to recreate the input–output dynamics of the NARMA-10 computational benchmark.

It was observed within the Santa Fe benchmark task that there are two regions of best performance, a large area within the bottom left of the parameter space (referred to as the lower region of best performance) and a smaller but better performing area within the center left of the parameter space (referred to as the upper region of best performance). The upper region of best performance correlates strongly to the KQ and GR plots, as the upper region of best performance is placed in the locations where KQ is high, and where GR is low. The lower region of best performance correlates strongly to areas with high LMC rather than KQ or GR, most noticeably within the odd cases of  $n$ , where the lower region of best performance is much wider due to the greater CA within the parameter space. This result is to be expected given the required dynamics for optimal performance to perform the Santa Fe computational benchmark, requiring high non-linearity and low-to-moderate memory.

### C. Mackey–Glass function utilization

**NARMA-10.** Figure 8 shows the heat-maps of the input and output behavior of the Mackey–Glass non-linear function for different values of the Mackey–Glass exponent, for a 200-node system, evaluating the NARMA-10 benchmark.



**FIG. 8.** Heat-maps of the input and output behavior of the Mackey–Glass non-linear node, for different values of the Mackey–Glass exponent, running on a 200-node delay-feedback reservoir system while evaluating the NARMA-10 benchmark.

With the exception of the Mackey–Glass exponent,  $n$ , being equal to 1, the heat-maps show that the non-linear function operates exclusively within the linear range to achieve the best performance for the NARMA-10 benchmark, utilizing both the positive and negative sides of the linear region, with the greatest point density being closer to the origin. As the NARMA-10 benchmark does require some non-linear dynamics in order for the reservoir to recreate the input/output behavior of NARMA, it is likely that a linear function is not the optimal function to achieve the best computational performance; an alternative non-linear function with different dynamics may lead to better computational performance.

An interesting observation is that when  $n$  is equal to 1, the heat-maps confirm the inherent instability of the function, with the instability showing a high utilization point density within a large region of the non-linear function. This heat-map explains why the behavior of this function is so different when  $n$  is equal to 1, as there is such a strong non-linear mapping of values within the function with very weak attractor behavior.

**Santa Fe.** Figure 9 shows the heat-maps of the input and output behavior of the Mackey–Glass non-linear function for different values of the Mackey–Glass exponent, for a 200-node system, evaluating the Santa Fe benchmark.

Unlike the NARMA-10 benchmark, which has the region with the greatest point density within the linear region of the

Mackey–Glass non-linear function, the region with the greatest point density within the Santa Fe benchmark is at the peak of where the attractor behavior is the strongest, for the majority of the heat-maps. Given the requirement for strong non-linearity to compute the Santa Fe benchmark, it is to be expected that the preferred region would be at the point of greatest non-linearity within the function.

The best performance is found when the Mackey–Glass exponent is at its lowest, where the linearity is at its highest and the attractor effect is at its weakest, giving the function a less aggressive non-linear transform, allowing for more of the non-linear function to be utilized. Typically, within the even values of the Mackey–Glass exponent, there is a localized high density point, with the rest of the non-linear function being weakly utilized. However, within the odd values of the Mackey–Glass exponent, the majority of the non-linear function is utilized, with no clear localized usage area. The additional utilization within the odd values of the Mackey–Glass exponent is due to the strong negative attractor behavior, mapping strong negative values to positive values rather than toward 0, as observed with the even Mackey–Glass exponent case.

## VI. EXPLORING ALTERNATIVE NON-LINEAR FUNCTIONS

Section V shows that the Mackey–Glass non-linear function has three operating regions when the exponent is: odd, there is an asymptote present within the function that causes an attractor behavior that is able to map negative values to positive values; even, the non-linear function is symmetrical, and this causes the attractor behavior to map negative values toward zero; equal to one, the system is chaotic and unstable. This shows that the Mackey–Glass non-linear function is limited in terms of non-linear dynamics, only exhibiting a region of linearity and an attractor as its edges.

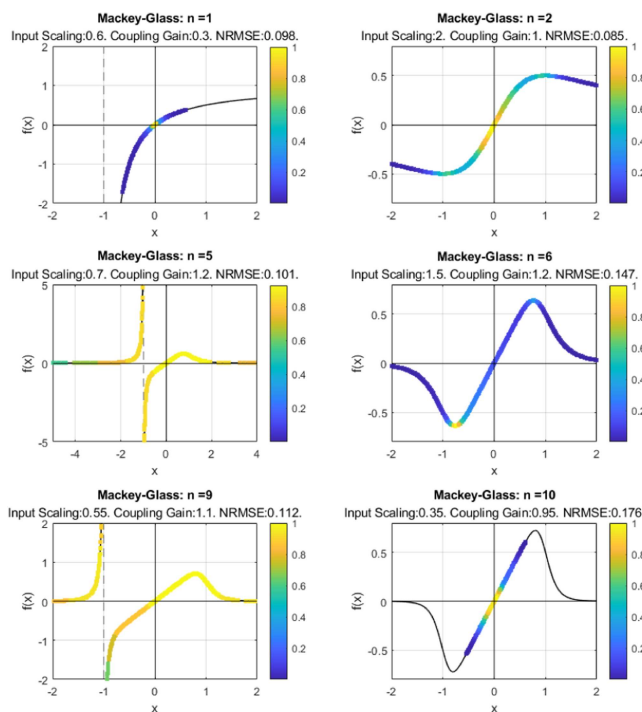
In order to find a better performing alternative non-linear function, four different non-linear functions are to be investigated: sine squared, double sinusoid, *Tan*, and *Tanh*.

### A. Alternative non-linear functions

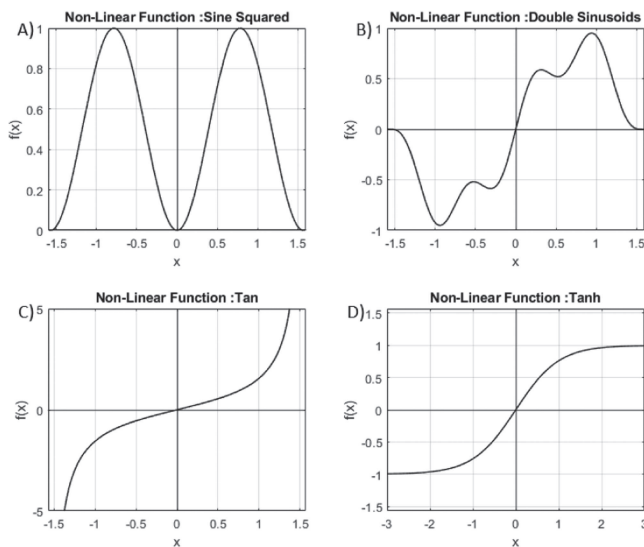
While the Mackey–Glass non-linear function has shown to provide limited non-linear dynamics to a delay-feedback reservoir system, there are other non-linear functions that have been used within delay-feedback reservoir systems.

One of the most commonly used non-linear functions is the *sinesquared* function,<sup>32</sup> which is popular in optical systems. As this function is symmetrical, always positive, and highly non-linear around the origin, this makes the sine squared function an ideal candidate for an alternative non-linear function; the function can be seen within Fig. 10(a). While the strong non-linear region at the origin will benefit computational tasks requiring a strong non-linear transform, other computational tasks such as NARMA-10 may be unable to utilize this non-linearity, potentially leading to a decrease in performance.

It was found that non-linear benchmark tasks, such as Santa Fe, have high utilization at the points with the greatest non-linearity. For the Mackey–Glass non-linear function, this was usually at the peaks and troughs of the function. In order to further explore this



**FIG. 9.** Heat-maps of the input and output behavior of the Mackey–Glass non-linear node, for different values of the Mackey–Glass exponent, running on a 200-node delay-feedback reservoir system while evaluating the Santa Fe benchmark.



**FIG. 10.** Four different non-linear functions to be evaluated as the non-linear function within a 200-node delay-feedback reservoir system. (a)  $f(x) = \sin^2(2x)$ , (b)  $f(x) = 0.8 \sin(2x) + 0.2 \sin(8x)$ , (c)  $f(x) = \tan(x)$ , and (d)  $f(x) = \tanh(x)$ .

phenomenon, a non-linear function constructed from double sinusoids is used giving a non-linear function with local minima and maxima as well as a global minima and maxima. The equation for the constructed non-linear function is given as

$$f(x) = 0.8 \sin(2x) + 0.2 \sin(8x). \quad (9)$$

This function can be seen within Fig. 10(b).

As the asymptotic behavior of the Mackey–Glass non-linear function with odd exponent values yields richer reservoir dynamics compared to even exponents, two popular activation functions used within artificial neural networks are to be investigated:<sup>3</sup> *Tan* and *Tanh*. The *Tan* non-linear function exhibits vertical asymptotes, similar to an odd Mackey–Glass non-linear function, with divergence to  $+\infty$  as  $x \rightarrow (\pi/2)^-$  and to  $-\infty$  as  $x \rightarrow (\pi/2)^+$ , whereas the *Tanh* non-linear function exhibits horizontal asymptotes. Previous studies have shown that the *Tanh* non-linear function provides strong performance and weak parameter sensitivity when used as an activation function in neurons within echo-state networks,<sup>33</sup> a property which may also extend to delay-feedback reservoirs. The functions *Tan* and *Tanh* can be seen within Figs. 10(c) and 10(d), respectively. The asymptotic behavior of these non-linear functions, combined with their approximately linear region near the origin, is expected to provide sufficient system dynamics to effectively compute both the NARMA-10 and Santa Fe benchmarks.

It should be noted that in this work, the non-linear functions are restricted to the form  $f(bx)$ , whereas in general, they are expressed as  $f(bx + c)$ , where  $b$  is a multiplicative coefficient and  $c$  is a bias term. The addition of a bias term shifts the operating region of the function by an offset of value  $c$ , thereby redefining the point around which the input data are centered. This is a particularly powerful technique within physical systems as they often have

constraints, for example, within optical systems which are restricted to positive input values. An equivalent effect can be achieved by changing the range of the input data: for example, an input range of  $[-1, 1]$  corresponds to no bias term, as the data are centered around 0, while an input range of  $[0, 1]$  corresponds to a bias of 0.5, as the data are centered around 0.5. In this work, no bias term is included, as the model is based on an electrical implementation in which zero is typically represented by 0 V.

## B. Simulation model and methodology

The simulation model described in Sec. IV B is modified to allow alternative non-linear functions to be used. This is achieved by replacing the “Mackey–Glass non-linear function” section of the simulation model, highlighted by the blue box in Fig. 4, with the desired non-linear function.

Using the experimental platform, we investigate how each of the four proposed non-linear functions influences the performance, dynamics, and utilization of the delay-feedback reservoir system. As with the Mackey–Glass based reservoir, a 200-node reservoir with a  $\tau$  of 80 s and a timescale of 400 ms is used, with a parameter sweep of both the input scaling and coupling gain performed for the NARMA-10 and Santa Fe benchmarks and system metrics. Table I shows the model parameter values and ranges for the proposed parameter sweep.

The procedure for generating input sequences for both the NARMA-10 and Santa Fe benchmarks in MATLAB is identical to that used with the Mackey–Glass based reservoir. A sequence of 6000 data points ranging from  $[-1, 1]$  is time-multiplexed by a random-weighted no-offset mask, with a washout period of 100 data points used. The training and testing datasets are split 80/20, respectively, with training performed using the Moore–Penrose pseudo-inverse algorithm.

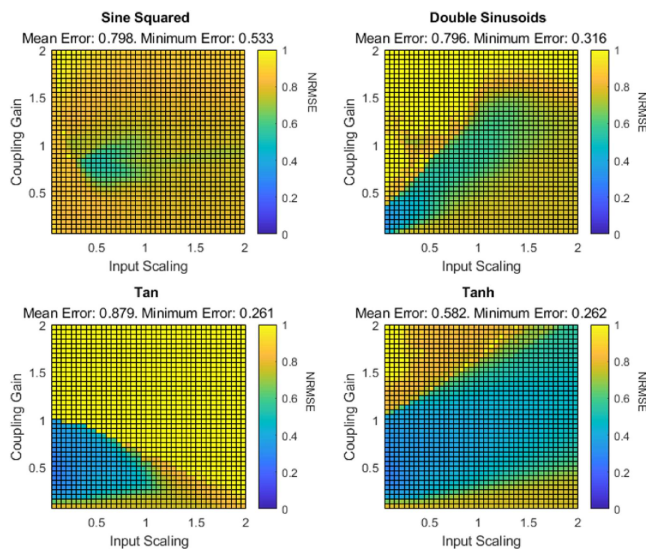
## VII. ALTERNATIVE NON-LINEAR FUNCTIONS RESULTS

### A. Alternative functions system parameter sweeps

**NARMA-10.** Figure 11 shows the effects of the input scaling and coupling gain on a 200-node delay-feedback reservoir system, evaluating the NARMA-10 computational benchmark, with four different non-linear functions. As with previous experiments, the parameter sweeps display the NRMSE of the reservoir system at each sweep point as a color spectrum, the lowest NRMSE value shown in blue and the highest in yellow. The maximum NRMSE value is set to 1, which typically indicates the system may have been drawn into saturation or has become unstable.

As predicted, the sine squared non-linear function has poor performance when running the NARMA-10 computational benchmark, with only a small region of best performance present within the center left of the parameter space, indicating that the function is highly sensitive to both input scaling and coupling gain. This poor performance and parameter sensitivity can be explained by the highly non-linear origin of the sine squared non-linear function.

As the origin of the sine squared non-linear function can be represented as a basin, a small change in the input within this basin represents an attenuation in the output. This small change results in an output that is less than the input, forcing the signal to be pushed



**FIG. 11.** Graphs showing the NRMSE values of a parameter sweep for a 200-node system running the NARMA-10 benchmark, using four different non-linear functions, with the input scaling and coupling gain ranging between 0.05 and 2.

toward zero. However, as both input scaling and coupling gain increase simultaneously, the signal within the non-linear function increases until it becomes large enough to escape the basin, at which point the output becomes larger than the input, forcing the signal to become larger before being mapped back toward zero. If both the input scaling and coupling gains are too large, the non-linear function will map the large input signals back toward zero, resulting in a significantly attenuated output. This attenuated output is then combined with a new input, which can be a large value if the input scaling is large, resulting in the loss of information. This effect also occurs when there is a significant difference between the input scaling and coupling gain, causing either a new input or the delayed reservoir output to dominate, leading to information loss. This causes a very high sensitivity to system parameters, resulting in a small region of best performance.

The performance of the double sinusoid non-linear function performs better than the sine squared non-linear function with a smaller minimum NRMSE value and a larger region of best operation. A greater overall performance is to be expected as the double sinusoid non-linear function has an approximately linear origin, which as seen in previous experiments increases the connection strength between virtual nodes resulting in a greater LMC. An interesting observation is that the region of best performance forms a distinct diagonal pattern, approximately following  $y = x$ , with the performance decreasing as the input scaling and coupling gain increases; this pattern is a result of the interaction between the input scaling, the coupling gain, and the linear operating region of the function.

Similar to the sine squared non-linear function, a significant difference between input scaling and coupling gain leads to a decline

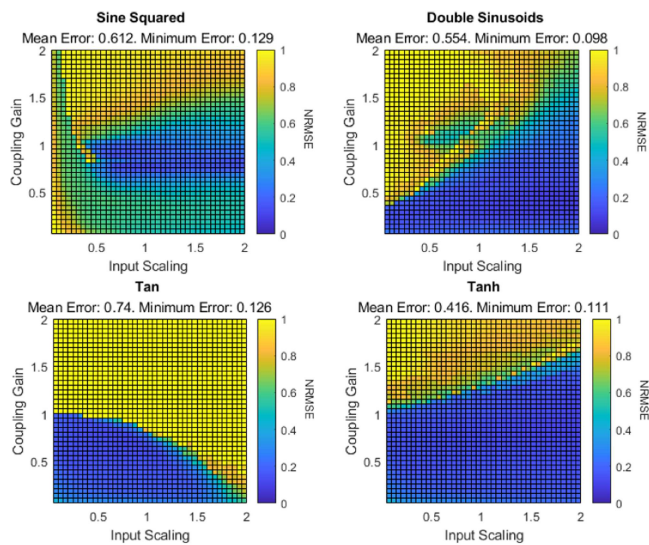
in performance. However, the linear region near the origin of the double sinusoidal non-linear function results in a diagonal region of best performance, as the non-linear function closely approximates a linear response within that area. As the input scaling and coupling gain increases, a larger signal is fed into the non-linear function, increasing the likelihood of exceeding its linear region and reducing the performance of the reservoir; this results in a region of decreased performance within the upper right of the parameter space.

Both the *Tan* and *Tanh* non-linear functions outperform all previously used non-linear functions when running the NARMA-10 computational benchmark. The *Tan* non-linear function shows a small region of best performance within the bottom left of the parameter space, with performance being poor when both the input scaling and coupling gain is greater than one. However, the *tanh* non-linear function exhibits a much broader region of best performance, covering most of the parameter space, with poor performance limited to the upper right and lower regions, while showing only minor sensitivity to input gain and low sensitivity to coupling gain. Despite being different functions, the shape of both regions of best performance can be attributed to the asymptotic behavior of the non-linear functions.

The small region of best performance, shown within the parameter space, likely indicates the input scaling and coupling gain parameters, where the *Tan* function behaves approximately linear. When these parameters are exceeded, the signal within the function behaves non-linearly and is drawn toward the vertical asymptotes. As the asymptotes are vertical, signals drawn toward the asymptotes can be mapped to another value, which then could be mapped toward zero or another value. If there is no external attenuation to either the input signal or the delayed reservoir output, it is easy for the reservoir system to become saturated, whereas within the *tanh* non-linear function, the asymptotes are horizontal. The horizontal asymptotes within the *tanh* function attenuate the signal, with large input values being mapped to one, which is then mapped toward zero. This behavior significantly decreases the sensitivity of both the input scaling and coupling gain parameters as it is very difficult to saturate the reservoir system. While this behavior makes a reservoir system more robust, the downside of this behavior is that it can be prone to information loss with large parameter values, as large signals with a value of two or greater are always mapped to the same value; this effect can be seen in the parameter sweep within the upper right region.

**Santa Fe.** Figure 12 shows the effects of the input scaling and coupling gain on a 200-node delay-feedback reservoir system, evaluating the Santa Fe computational benchmark, with four different non-linear functions.

As expected, one of the most notable differences between the Santa Fe and NARMA-10 benchmarks is that the Santa Fe benchmark exhibits significantly better performance, both in terms of NRMSE and parameter sensitivity. However, while the locations and distribution of the areas of best performance appear different, on closer inspection, the Santa Fe parameter sweeps show an expanded version of the parameter sweeps performed within the NARMA-10 benchmark. This is due to the Santa Fe computational benchmark exploiting a larger portion of the parameter space when compared to the NARMA-10 benchmark, as it can leverage the non-linear components of the non-linear functions, a capability that the



**FIG. 12.** Graphs showing the NRMSE values of a parameter sweep for a 200-node system running the Santa Fe benchmark, using four different non-linear functions, with the input scaling and coupling gain ranging between 0.05 and 2.

NARMA-10 benchmark is unable to do. The Santa Fe benchmark also exhibits a reduced sensitivity to input scaling, which was also observed within the Mackey–Glass based reservoirs.

Although the sine squared non-linear function achieves a low minimum NRMSE, its performance remains highly sensitive to parameter variations, a feature once again due to the strong non-linear behavior at the origin. While the Santa Fe benchmark is able to utilize more of the dynamics of the highly non-linear origin within the sine squared function, it is still subject to the operating points within the basin present at the origin of the function; this is due to small signals being attenuated and larger signals being amplified. The nature of this basin greatly increases the sensitivity to the system parameters when compared to other non-linear functions. Despite the strong non-linear behavior, which is often a desired attribute when solving highly non-linear computational problems, the behavior at the origin of the sine squared non-linear function appears to reduce performance; with the minimum NRMSE value performing similar to a high  $n$  valued Mackey–Glass based reservoir.

Out of the four chosen non-linear functions, the double sinusoid non-linear function exhibits the best NRMSE performance, comparable to the best performing Mackey–Glass based reservoir systems, whereas the region of best performance within the NARMA-10 benchmark formed an approximate diagonal, following  $y = x$ , the region of best performance within the Santa Fe benchmark is  $y < x$ , with the bottom right diagonal of the parameter space exhibiting relatively consistent NRMSE values. Similar to the Mackey–Glass based reservoirs, there is a smaller second region of operation within the center of the parameter space. At this point, the signal can become large enough to strongly utilize both the local and global minima and maxima, which appears to result in a performance decrease. This is also evident in the parameter sweep, with

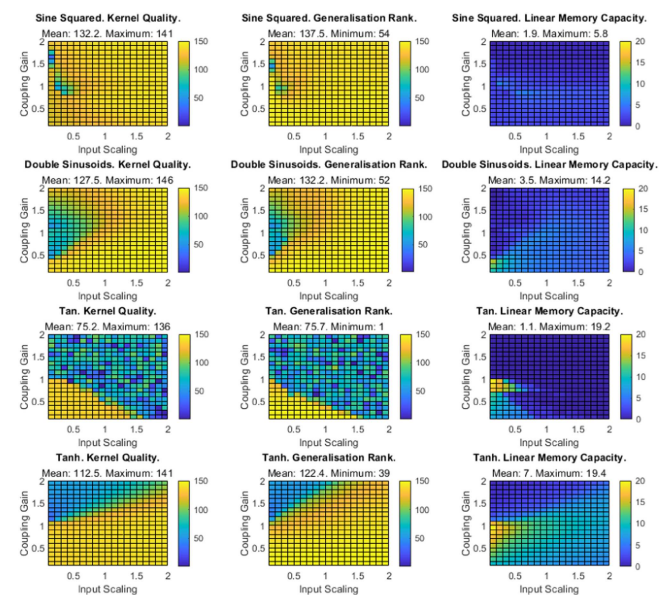
the performance decreasing in the upper right as both input scaling and coupling gain increase.

While both the *Tan* and *tanh* non-linear functions outperformed all previously tested non-linear functions when running the NARMA-10 benchmark, the performance when running the Santa Fe benchmark is comparable to that of the Mackey–Glass based reservoir system in terms of NRMSE, indicating that the non-linear dynamics within these functions are not as strong as the other investigated non-linear functions. The regions of best operation within the *Tan* and *tanh* non-linear functions resemble that of the NARMA-10 benchmark, but with much less sensitivity within the input scaling, as seen by a broader range within the parameter space. Within these regions of best operation, the NRMSE values are more consistently distributed, exhibiting a much steeper gradient between areas of high and low performance. This behavior suggests that the Santa Fe benchmark is able to leverage a larger portion of the parameter space, utilizing a broader range of the non-linear function when compared to the NARMA-10 benchmark.

## B. Alternative functions system metrics

Figure 13 shows the KQ, GR, and LMC of a 200-node delay-feedback reservoir system during a parameter sweep of the input scaling and coupling gain, when evaluating four different non-linear functions.

The system metrics show very different dynamics for each of the different non-linear functions, confirming that the chosen non-linear functions do provide different dynamics to the reservoir.



**FIG. 13.** Graphs showing a parameter sweep for a 200-node system, using four different non-linear functions, with the input scaling and coupling gain ranging between 0.05 and 2, for the following system metrics: kernel quality, generalization rank, and linear memory capacity.

The sine squared non-linear function exhibits a large KQ and GR throughout most of the parameter space, with a small region within the upper left displaying a few smaller values. This implies that the sine squared non-linear function is able to map the input vectors onto a high-dimensional non-linear space, but unable to generalize between inputs. This observation is consistent with the sine squared parameter sweep presented in Sec. VII A, which demonstrated that the function is capable of providing a non-linear transformation, but is susceptible to saturation.

The LMC remains low across the majority of the parameter space, with higher values predominantly concentrated in the center left region. Given the strong non-linear behavior focused around the origin of the function, a small LMC is to be expected as the connections between virtual nodes will be weak. This effect is reflected in the poor performance of the NARMA-10 parameter sweep, where the region of best performance is located within the center left of the parameter space.

The system metrics of the double sinusoid non-linear function also shows a large KQ and GR throughout most of the parameter space, but has a larger region in the center left displaying smaller values. While this indicates the reservoir is able to generalize inputs well, it is unable to map inputs into high-dimensional space, which is a much more important metric when computing temporal computational problems. The poor KQ within this region is a result of information loss between large coupling gains and small input scaling. This effect is reduced as the coupling gain increases, as these signals are being non-linearly transformed by the local minima and maxima, which are attenuating the signal proportionally. This results in a universal region of poor performance, as seen by both the NARMA-10 and Santa Fe parameter sweeps, where the upper left corner exhibits this behavior.

The LMC of the double sinusoid function is significantly higher than that of the sine squared function. This increase is due to the presence of the linear region within the function, which creates stronger connections between virtual nodes, as information is linearly mapped to adjacent virtual nodes. This is further supported by the observation that the distribution of LMC values across the parameter space exhibits a diagonal pattern, a similar observation made within the distribution of NRMSE values within the parameter sweep for the NARMA-10 benchmark.

The shape and distribution of the KQ and GR system metrics in the *Tan* function closely resemble those observed in the Mackey–Glass based reservoirs with odd exponent values of  $n$ , a similarity that can be attributed to the presence of the vertical asymptotes in both functions. When the input scaling and coupling gain are high enough for the *Tan* function to utilize the asymptotic behavior of the function, signals are drawn toward the asymptote and then mapped to another value. This can create rich dynamics, as seen within the odd exponent Mackey–Glass based reservoirs where large negative signals are mapped to the positive region of the function and then mapped toward zero, but as there are asymptotes in both the positive and negative regions of the *Tan* function, it is more likely to create instability as *Tan* lacks the attractor mechanism to bring the signal back toward zero quickly. This is reflected in the system metric plots, where the KQ and GR values demonstrate a region of consistency within the lower left of the parameter space, which also corresponds to the region where the concentration of LMC is at

its highest. This lower left region within the parameter space correlates to the regions of best performance within the parameter sweeps for both the NARMA-10 and Santa Fe benchmark tasks and is where the *Tan* function exhibits approximately linear behavior.

The system metrics within the *tanh* non-linear function once again demonstrates how the horizontal asymptote can decrease the sensitivity of parameters, with all but the upper left region of the parameter space having high KQ and GR values. Although the high GR throughout the parameter space implies that the reservoir cannot generalize inputs, the KQ indicates strong mapping of input vectors into a high-dimensional space. The poor GR can be attributed to the *tanh* non-linear function providing a weak attenuating transform to the reservoir. While the *tanh* exhibits a strong enough non-linear transform to perform computation, it is insufficient for the reservoir to differentiate between similar inputs.

Similar to the double sinusoid non-linear function, the poor KQ performance within the upper left region of the *tanh* non-linear function creates a region of poor performance within both the NARMA-10 and Santa Fe parameter sweeps. This is once again due to the loss of information with large coupling gains and small input scaling. However, unlike the double sinusoid function, the *tanh* function cannot differentiate between different large signals, with all strong signals being mapped toward 1; therefore, the region of poor performance is spread wider across the parameter space.

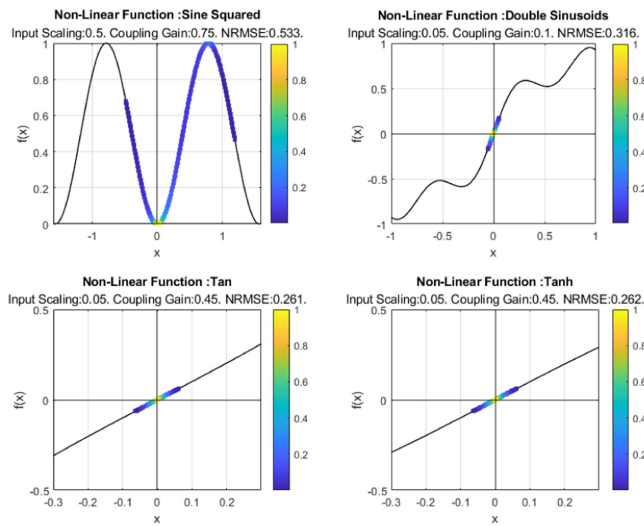
The LMC system metric for the *tanh* non-linear function shows a greater sensitivity to input scaling than with the KQ and GR, but occupies a significantly broader range than the other non-linear functions. The lower left of the parameter space shows the highest concentration of LMC, which once again corresponds to the linear operating region of the *tanh* function. This broader range of LMC is reflected within the NARMA-10 parameter sweeps, where the best NRMSE values are within the lower left of the parameter space, with the performance decreasing as the input scaling and coupling gain increases.

### C. Alternative function utilization

**NARMA-10.** Figure 14 shows the heat-maps of the input and output behavior of four different non-linear functions, for a 200-node system, evaluating the NARMA-10 benchmark.

With the exception of the sine squared non-linear function, the heat-maps demonstrate that the function exclusively operates within the linear region of the non-linear functions, with the greatest point density at the origin, a result also observed within the Mackey–Glass based reservoir systems. The heat-maps also show that only a small area of the linear region is used, with the signals typically operating within  $[-0.1, 0.1]$  to maintain operation within the approximate linear region. In contrast, the heat-map shows that the sine squared non-linear function operates across a broad range of its domain, with the highest point density concentrated in the basin near the origin. This utilization provides insight into why the performance of the sine squared non-linear function was so poor, as it is unable to provide a linear region for the NARMA-10 benchmark to operate within, forcing the reservoir to operate within a non-linear region.

An interesting observation is that the minimum NRMSE achieved by the double sinusoid function differs from that of the *Tan* and *Tanh* functions, despite all of the functions operating



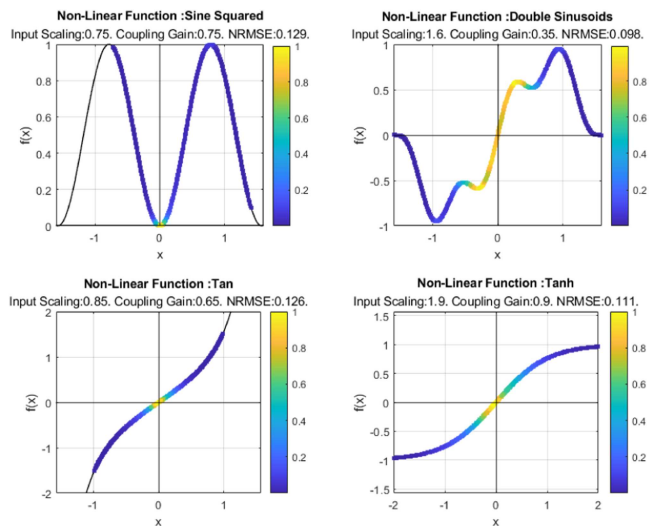
**FIG. 14.** Heat-maps of the input and output behavior for four different non-linear functions, running on a 200-node delay-feedback reservoir system while evaluating the NARMA-10 benchmark.

within their linear region, a feature due to the gradients of the linear regions. The gradient of the double sinusoid function within its linear region is greater than one, whereas the gradients of the *Tan* and *Tanh* functions within their linear regions are approximately one. This causes the input signal to be transformed into a larger output signal within the double sinusoid function, but gives a unity transform within the *Tan* and *Tanh* functions. This suggests that, for the NARMA-10 benchmark, the optimal function identified in this study is  $y = x$ , indicating that the presence of a non-linear transformation is detrimental to NRMSE performance.

Another notable observation from the heat-maps indicates how the non-linear functions would behave under the influence of additive noise when computing the NARMA-10 benchmark. With the exception of the sine squared non-linear function, the functions operate within a small portion of their respective linear regions. This suggests that mild additive noise would not drive the system into divergence or cause data loss, since the perturbations would remain within the linear operating range. In contrast, for the sine squared non-linear function, even a small amount of noise could result in data loss, as small input variations can produce disproportionately large changes in the output.

**Santa Fe.** Figure 15 shows the heat-maps of the input and output behavior of four different non-linear functions, for a 200-node system, evaluating the Santa Fe benchmark.

In contrast to the NARMA-10 benchmark, the heat-maps show a strong utilization across the entire domain of the non-linear functions, with the highest point density occurring at the origin of the functions. A noticeable observation is the region of high point density within the double sinusoid non-linear function, indicating the strong utilization of the local minima and maxima within the function. This result is to be expected, as the Santa Fe benchmark has a requirement for strong non-linearity in order to be computed.



**FIG. 15.** Heat-maps of the input and output behavior for four different non-linear functions, running on a 200-node delay-feedback reservoir system while evaluating the Santa Fe benchmark.

A notable observation is the similarity in the utilization of the sine squared non-linear function across both the NARMA-10 and Santa Fe benchmarks. For the other three non-linear functions, it is possible to modify the operating region of the function, in order to change or introduce new non-linear behavior, by adjusting the input scaling and coupling gain. However, this is not possible with the sine squared non-linear function. A change in the input scaling and coupling gain does not introduce any additional non-linear behavior, as the strongest non-linear behavior is at the origin of the function, which is always present with an input range of  $[-1, 1]$ . However, a change in the parameters does affect the strength of the non-linearity and it is highly sensitive to minor changes; leading to an increased parameter sensitivity. This results in the sine squared non-linear function being unable to provide a different response to different benchmark tasks, restricting its use as a general non-linear function when operating around the origin.

While the heat-maps for the NARMA-10 benchmark suggest overall robustness to additive noise, the heat-maps for the Santa Fe benchmark indicate that only the *tan* and *tanh* functions exhibit comparable robustness. With the exception of the sine squared non-linear function, the other functions operate within a relatively small region of operation, suggesting that mild additive noise would not drive the system into divergence. However, the double sinusoid function exhibits a stronger non-linear transform, and it is likely that additive noise would cause significant data loss and lead to incorrect mappings of input values. By contrast, the *tan* and *tanh* functions exhibit much milder non-linear transformations, which greatly reduces the likelihood of data loss within the reservoir system.

## VIII. CONCLUSIONS

The delay-feedback reservoir offers a versatile and hardware-efficient framework for performing temporal computational tasks.

One of the most powerful aspects of this framework is its ability to decouple the non-linear node into two fundamental components: a non-linear function and an integrator. This modular structure enables the system dynamics of the reservoir to be modified by changing the non-linear function or integrator timescale.

In this study, we investigate the system dynamics of a delay-feedback reservoir through the evaluation of key metrics: Normalized Root Mean Squared Error (NRMSE), Kernel Quality (KQ), Generalization Rank (GR), and Linear Memory Capacity (LMC). We begin by analyzing the dynamics of a Mackey–Glass based reservoir to assess the performance characteristics of the Mackey–Glass non-linear function. Subsequently, we explore four alternative non-linear functions, sine squared, double sinusoid, *Tan*, and *Tanh*, to examine their respective impacts on the reservoir's dynamic behavior.

We show that the Mackey–Glass non-linear function is limited within its system dynamics, only producing three different operating states when the Mackey–Glass exponent is: odd there is an asymptote present within the function, which causes an attractor behavior that is able to map negative values to positive values; even, the non-linear function is symmetrical, and this causes the attractor behavior to map negative values toward zero; equal to one, the system is chaotic and unstable. While it achieves strong NRMSE performance on benchmark tasks that demand significant non-linear transformation, its performance degrades on memory intensive tasks. Additionally, we observe that in certain operating modes, the Mackey–Glass non-linear function displays low sensitivity to system parameters.

We then demonstrate four alternative non-linear functions, which introduce distinct system dynamics to the reservoir system. The sine squared function exhibits a basin effect, where small values are attenuated and larger values tend back toward zero, due to the function always being positive and strongly non-linear at the origin. The lack of an approximately linear region at the origin greatly limits the performance of computational tasks that require memory, as observed within the NARMA-10 benchmark, and the basin effect creates a strong sensitivity to parameters. In contrast, the double sinusoid function combines an approximately linear region at the origin with multiple local extrema. The linear segment strengthens the memory capacity of the system by increasing the connectivity between virtual nodes, while the extrema create a strong non-linear transform. This results in the double sinusoid function delivering the best performance on benchmarks emphasizing non-linearity, with moderate sensitivity to parameter variations. The *Tan* non-linear function introduces vertical asymptotes, resembling the odd exponent Mackey–Glass non-linear function. This results in rich non-linear dynamics outside of the origin of the function, but without a mechanism of the input values to return to toward zero, values are prone to divergence, thus resulting in instability and a strong sensitivity to parameter variations. Finally, the *tanh* non-linear function provides a smooth sigmoidal non-linearity, with a near-linear origin, and horizontal asymptotes. The central linearity at the origin of the function allows for a strong non-linear memory, while the saturation at large amplitudes prevents divergence. This balances non-linear dynamics and robustness, as observed within the NRMSE values of the NARMA-10 and Santa Fe parameter sweeps,

resulting in a more general-purpose performance when compared to the task-specific strengths of the other non-linearities.

This study shows that the non-linear function can be tuned to create different non-linear behaviors and system dynamics, leading to an increase in performance. These changes in behavior and dynamics can be achieved within the Mackey–Glass non-linear function by changing the exponent value, which can be changed relatively easily within hardware without a hardware redesign, typically by changing the bias resistor values. The dynamics within alternative non-linear functions and the Mackey–Glass non-linear function can also be modified by the input scaling and coupling gain, which can be reconfigured during run-time through reconfigurable gain devices such as op-amps. This creates a framework that allows systems to be tuned to provide the required system dynamics to solve a particular computational task, within physical delay-feedback reservoir systems.

Future work may investigate the effect of incorporating a bias term into the system design, for example, within the sine squared non-linear function. It was observed that the sine squared function performed poorly when the input data were centered around the origin of the function. Additional performance may be achieved by introducing a bias term to shift the operating point into the approximately linear region, rather than being confined to the basin near the origin. This modification could potentially enhance the function's ability to compute a broader range of tasks.

In conclusion, we have found that it is difficult to create a universal non-linear function that can provide the necessary dynamics to compute computational tasks with different computational requirements in terms of the lowest NRMSE values, without the ability to change the input scaling and coupling gain parameters in real-time. However, it is possible to create a general non-linear function that can reduce sensitivity to system parameters for a wide range of computational tasks, but with a performance reduction compared to a tailored non-linear function for a specific computational task. This gives rise to considering reconfigurable hardware reservoir computers.

## ACKNOWLEDGMENTS

Experiments were carried out using the Viking Cluster, a compute cluster provided by the University of York.

## AUTHOR DECLARATIONS

### Conflict of Interest

The authors have no conflicts to disclose.

## Author Contributions

**Alexander C. McDonnell:** Conceptualization (equal); Data curation (equal); Formal analysis (equal); Investigation (equal); Methodology (equal); Validation (equal); Visualization (equal); Writing – original draft (equal); Writing – review & editing (equal). **Martin A. Trefzer:** Conceptualization (equal); Investigation (equal); Methodology (equal); Supervision (equal); Writing – review & editing (equal).

## DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author upon reasonable request.

## REFERENCES

- <sup>1</sup>M. Brin and G. Stuck, *Introduction to Dynamical Systems* (Cambridge University Press, 2002).
- <sup>2</sup>H. Jaeger, “The “echo state” approach to analysing and training recurrent neural networks—With an erratum note,” German National Research Center for Information Technology, GMD Technical Report 148, Bonn, Germany, 2001, p. 13.
- <sup>3</sup>P. Antonik, A. Smerieri, F. Duport, M. Haelterman, and S. Massar, “FPGA implementation of reservoir computing with online learning,” in *24th Belgian-Dutch Conference on Machine Learning* (2015).
- <sup>4</sup>M. C. Soriano, S. Ortín, L. Keuninckx, L. Appeltant, J. Danckaert, L. Pesquera, and G. van der Sande, “Delay-based reservoir computing: Noise effects in a combined analog and digital implementation,” *IEEE Trans. Neural Netw. Learn. Syst.* **26**, 388–393 (2015).
- <sup>5</sup>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,” *Nat. Commun.* **8**, 1–10 (2017).
- <sup>6</sup>C. Fernando and S. Sojakka, “Pattern recognition in a bucket,” in *European Conference on Artificial Life* (Springer, 2003), pp. 588–597.
- <sup>7</sup>M. Lukoševičius, “A practical guide to applying echo state networks,” in *Neural Networks: Tricks of the Trade* (Springer, 2012), pp. 659–686.
- <sup>8</sup>L. Appeltant, M. C. Soriano, G. Van der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C. R. Mirasso, and I. Fischer, “Information processing using a single dynamical node as complex system,” *Nat. Commun.* **2**, 1–6 (2011).
- <sup>9</sup>F. M. Atay, *Complex Time-Delay Systems: Theory and Applications* (Springer, 2010).
- <sup>10</sup>M. L. Alomar, E. S. Skibinsky-Gitlin, C. F. Frasser, V. Canals, E. Isern, M. Roca, and J. L. Rosselló, “Efficient parallel implementation of reservoir computing systems,” *Neural Comput. Appl.* **32**, 2299–2313 (2020).
- <sup>11</sup>D. Brunner, B. Penkovsky, B. A. Marquez, M. Jacquot, I. Fischer, and L. Larger, “Tutorial: Photonic neural networks in delay systems,” *J. Appl. Phys.* **124**, 152004 (2018).
- <sup>12</sup>Y. Paquot, F. Duport, A. Smerieri, J. Dambre, B. Schrauwen, M. Haelterman, and S. Massar, “Optoelectronic reservoir computing,” *Sci. Rep.* **2**, 1–6 (2012).
- <sup>13</sup>A. C. McDonnell, “System analysis and design of physical delay-feedback reservoir computing systems,” Ph.D. thesis (University of York, 2024).
- <sup>14</sup>A. C. McDonnell and M. A. Trefzer, “The effect of system timescale on virtual node connectivity within delay-feedback reservoirs,” in *2023 International Joint Conference on Neural Networks (IJCNN)* (IEEE, 2023), pp. 1–8.
- <sup>15</sup>A. S. Weigend, *Time Series Prediction: Forecasting the Future and Understanding the Past* (Routledge, 2018).
- <sup>16</sup>K. Nakajima, H. Hauser, T. Li, and R. Pfeifer, “Information processing via physical soft body,” *Sci. Rep.* **5**, 10487 (2015).
- <sup>17</sup>C. Wringe, M. Trefzer, and S. Stepney, “Reservoir computing benchmarks: A tutorial review and critique,” *Int. J. Parallel Emergent Distrib. Syst.* **40**(4), 313–351 (2025).
- <sup>18</sup>Y.-c. Wu and J.-w. Feng, “Development and application of artificial neural network,” *Wirel. Pers. Commun.* **102**, 1645–1656 (2018).
- <sup>19</sup>M. C. Soriano, D. Brunner, M. Escalona-Morán, C. R. Mirasso, and I. Fischer, “Minimal approach to neuro-inspired information processing,” *Front. Comput. Neurosci.* **9**, 68 (2015).
- <sup>20</sup>K. Vandoorne, J. Dambre, D. Verstraeten, B. Schrauwen, and P. Bienstman, “Parallel reservoir computing using optical amplifiers,” *IEEE Trans. Neural Netw.* **22**, 1469–1481 (2011).
- <sup>21</sup>J. Torrejon, M. Riou, F. A. Araujo, S. Tsunegi, G. Khalsa, D. Querlioz, P. Bortolotti, V. Cros, K. Yakushiji, A. Fukushima *et al.*, “Neuromorphic computing with nanoscale spintronic oscillators,” *Nature* **547**, 428–431 (2017).
- <sup>22</sup>J. Xia, J. Chu, S. Leng, and H. Ma, “Reservoir computing decoupling memory–nonlinearity trade-off,” *Chaos* **33**, 113120 (2023).
- <sup>23</sup>A. F. Atiya and A. G. Parlos, “New results on recurrent network training: Unifying the algorithms and accelerating convergence,” *IEEE Trans. Neural Netw.* **11**, 697–709 (2000).
- <sup>24</sup>H. Jaeger, “Short term memory in echo state networks. GMD-Report 152,” in *GMD-German National Research Institute for Computer Science (2002)* (Citeseer, 2002), see <http://www.faculty.jacobs-university.de/hjaeger/pubs/STMEchoStatesTechRep.pdf>.
- <sup>25</sup>R. Legenstein and W. Maass, “Edge of chaos and prediction of computational performance for neural circuit models,” *Neural Netw.* **20**, 323–334 (2007).
- <sup>26</sup>M. C. Mackey and L. Glass, “Oscillation and chaos in physiological control systems,” *Science* **197**, 287–289 (1977).
- <sup>27</sup>A. Namajūnas, K. Pyragas, and A. Tamaševičius, “An electronic analog of the Mackey-Glass system,” *Phys. Lett. A* **201**, 42–46 (1995).
- <sup>28</sup>P. Amil, C. Cabeza, and A. C. Martí, “Electronic implementation of the Mackey-Glass delayed model,” *arXiv:1408.5083* (2014).
- <sup>29</sup>L. Appeltant *et al.*, “Reservoir computing based on delay-dynamical systems,” These de Doctorat (Vrije Universiteit Brussel/Universitat de les Illes Balears, 2012).
- <sup>30</sup>MathWorks, see <https://uk.mathworks.com/products/simulink.html> for information about “Simulation and model-based design” (2025).
- <sup>31</sup>R. MacAusland, “The Moore-Penrose inverse and least squares,” *Math 420: Adv. Top. Linear Algebra* **2014**, 1–10.
- <sup>32</sup>L. Larger, M. C. Soriano, D. Brunner, L. Appeltant, J. M. Gutierrez, L. Pesquera, C. R. Mirasso, and I. Fischer, “Photonic information processing beyond Turing: An optoelectronic implementation of reservoir computing,” *Opt. Express* **20**, 3241–3249 (2012).
- <sup>33</sup>J. B. Butcher, D. Verstraeten, B. Schrauwen, C. R. Day, and P. W. Haycock, “Reservoir computing and extreme learning machines for non-linear time-series data analysis,” *Neural Netw.* **38**, 76–89 (2013).