Tradeoffs with physical delay feedback reservoir computing

Tian Gan
Department of Electronic Engineering
University of York
York, United Kingdom
tg847@york.ac.uk

Susan Stepney
Department of Computer Science
University of York
York, United Kingdom
susan.stepney@york.ac.uk

Martin A. Trefzer
Department of Electronic Engineering
University of York
York, United Kingdom
martin.trefzer@york.ac.uk

I. Introduction

Reservoir Computing (RC) [1] is a supervised learning scheme exploiting the computational ability of non-linear dynamical systems. In delay-feedback reservoir computing a virtual network is achieved through time-multiplexing, trading off input frequency and reservoir complexity. Here we investigate several features and tradeoffs that affect physical delay-dynamical reservoirs, based on the Mackey–Glass model, including the role of input mask, and the 'devirtualisation' of the output. We evaluate performance using the NARMA-10 benchmark. We find that random binary masks with no offset outperform other masking schemes. We find that reading the output a subset of output nodes, the 'devirtualised network', does not affect computational power, which enables a tradeoff between output design complexity and readout frequency.

II. Delay Feedback Reservoir based on Mackey-Glass model

Appeltant et al. [2] introduced the delay-feedback reservoir computing structure. It is an effective approach to simplifying hardware implementation, as the system has only one nonlinear element along with a delay line. Here we investigate some issues that are of interest in physical reservoir implementations that are potentially hidden in simulated time virtual reservoirs. Our work is still in simulation, but we move a step closer to a physical implementation: rather than directly numerically integrating the model equation, we use a Mackey-Glass nonlinear node and explicit delay line, all expressed in a Simulink circuit, and we retain physical time units.

We follow [2] in introducing inputs, but we do not normalise time (we do not set $\gamma = 1$), to retain physical time units.

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

In this paper, we use the discrete-time NARMA-10 benchmark to analyse the computational abilities of the Mackey–Glass delay-based reservoir in various scenarios.

III. Masking the Input

We evaluate the computational performances of different masking procedures (see fig.1) under two experimental con-

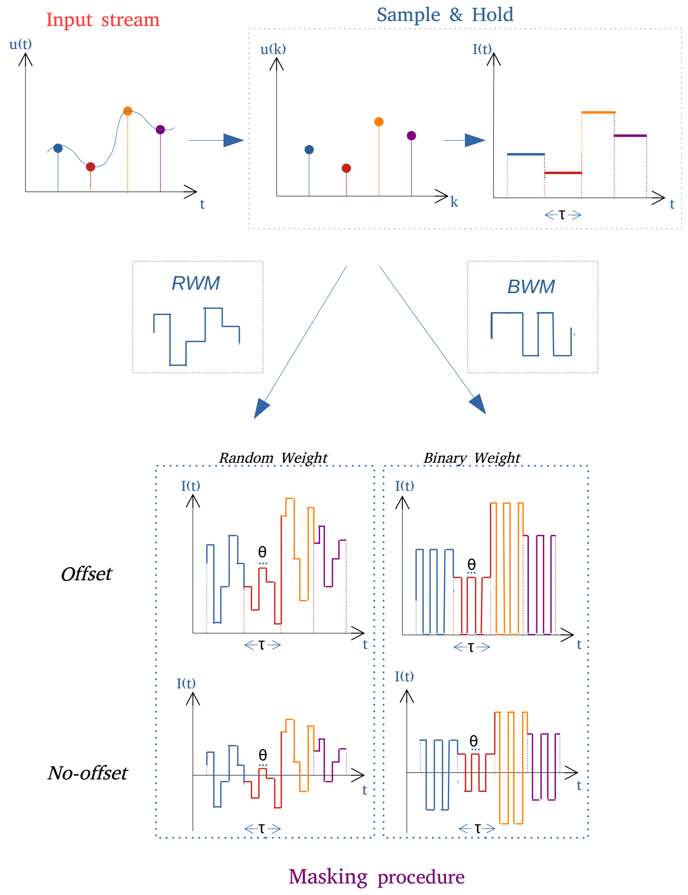
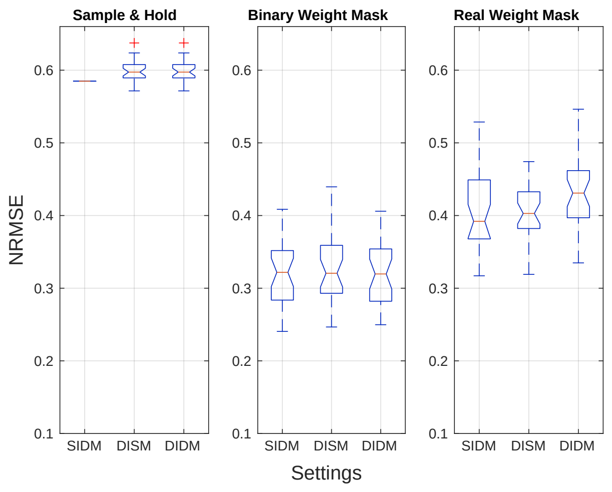


Fig. 1: Time-multiplexing procedure. Top: inputs, from physical continuous inputs to the discrete time inputs through Sample and Hold. Middle: two masks. Bottom: four masking procedures applied to the discretised inputs.

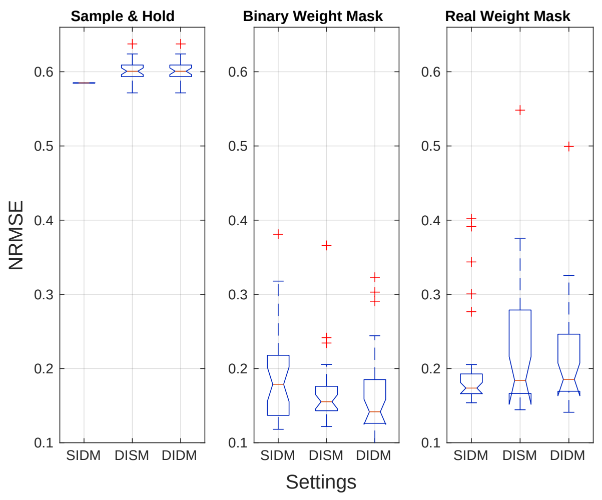
figurations: mask-multiplexing with offset (Eq.2), and with no offset (Eq.3):

$$I(t) = I_0(t)(M(t) + 1) \tag{2}$$
$$I(t) = I_0(t)M(t) \tag{3}$$

We compare the computational performance of three different masking procedures: *Sample and Hold* (as a control),



(a) Offset



(b) No offset

Fig. 2: Simulation results for the NARMA-10 benchmark task. 2a and 2b are based on different time-multiplexing functions: Eq.2 and Eq.3 respectively. Each experiment is based on 30 runs.

Binary Weight Mask and *Real Weight Mask* under two different mask-multiplexing equations, with offset (Eq.2) and with no offset (Eq.3). We use three different configurations to investigate whether the randomness of input signals or of the masking is dominating the computational performance of the system: i) SIDM has the same input sequence but different random masks for each run; ii) DISM has different random input sequences but the same masks for each run; iii) DIDM has different input sequences and different masks for each run.

The results are shown in fig.2. As expected, *Sample and Hold* with no masking performs poorly: there are no virtual nodes. Binary masking outperforms real weight masking, and no offset outperforms offset, in this case. The source of randomness (input or mask) does not affect the performance.

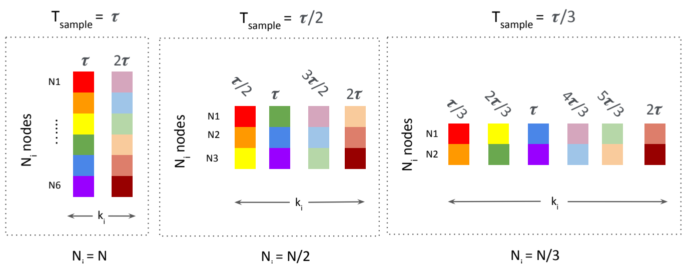


Fig. 3: ‘Devirtualisation’ options with $N = 6$ nodes. Each coloured square block represent one output signal from one virtual node. N_i indicates the number of connected nodes (output lines). T_{sample} is the sampling time, i.e, every connected output line collects data simultaneously at each T_{sample} time. When $N_i = N$, every virtual node $N1..N6$ has an output line, and the output is sampled every τ. When $N_i = N/2$, nodes $N1..N3$ have output lines: at $\tau/2$ they output the first three blocks, at time τ they output the last three blocks. Similarly, when $N_i = N/3$, nodes $N1..N2$ have output lines, outputting each $\tau/3$.

IV. Devirtualising the Output

The delay line masking trades off space (number of inputs and outputs) against time (frequency of providing an input or reading an output).

We investigate a sub-sampling devirtualisation approach, that allows this tradeoff to be altered: reducing the readout frequency by using multiple output lines while the number of virtual nodes in the reservoir layer is consistent.

Fig. 3 illustrates this output ‘devirtualisation’ concept. If we set the sampling time equal to τ, the delay time along the physical tapped delay, we need to use N output lines to collect data simultaneously from each of the N ‘devirtualised nodes’. We have increased the number of outputs to N, and reduced the output frequency by N (from $1/\theta$ to $1/\tau$). If we sub-sample the system with time $\tau/2$, only half of the virtual nodes are required to output data simultaneously.

No data is lost or changed in this output devirtualisation: the same data is sampled in each case, and the observed performances (not shown here) are identical. This is our control case: in future work we will include the effect of losses along the delay line.

References

[1] H. Jaeger, “The “echo state” approach to analysing and training recurrent neural networks – with an erratum note,” *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, vol. 148, no. 34, p. 13, 2001.

[2] L. Appeltant *et al.*, “Information processing using a single dynamical node as complex system,” *Nature Comms*, vol. 2, no. 1, pp. 468–468, 2011.