An Analytical Approach to Single Node Delay-Coupled Reservoir Computing

Johannes Schumacher, Hazem Toutounji, and Gordon Pipa

Institute of Cognitive Science, University of Osnabrück, Germany
{joschuma,htoutounji,gpipa}@uos.de

Abstract. Reservoir computing has been successfully applied in difficult time series prediction tasks by injecting an input signal into a spatially extended reservoir of nonlinear subunits to perform history-dependent nonlinear computation. Recently, the network was replaced by a single nonlinear node, delay-coupled to itself. Instead of a spatial topology, subunits are arrayed in time along one delay span of the system. As a result, the reservoir exists only implicitly in a single delay differential equation, numerical solving of which is costly. We derive here approximate analytical equations for the reservoir by solving the underlying system explicitly. The analytical approximation represents the system accurately and yields comparable performance in reservoir benchmark tasks, while reducing computational costs by several orders of magnitude. This has important implications with respect to electronic realizations of the reservoir and opens up new possibilities for optimization and theoretical investigation.

1 Introduction

Predicting future behavior and learning temporal dependencies in time series of complex natural systems remains a major goal in many disciplines. In Reservoir Computing, the issue is tackled by projecting input time series into a recurrent network of nonlinear subunits [2, 4]: Recurrency provides memory of past inputs, while the nonlinear subunits expand their informational features. History-dependent nonlinear computations are then achieved by simple linear readouts of the network activity.

In a recent advancement, the recurrent network was replaced by a single nonlinear node delay-coupled to itself [1]. Such a setup is formalized by a delay differential equation which can be interpreted as an "infinite dimensional" dynamical system. Whereas classical reservoirs have an explicit spatial representation, a delay-coupled reservoir (DCR) uses temporally extended sampling points across the span of its delayed feedback, termed *virtual nodes*. The main advantage of such a setup is that it allows for easy realization in optical and electronic hardware [8].

A drawback of this approach is the fact that the actual reservoir computer is always only implicit in a single delay differential equation. Consequently, in many implementations the underlying system has to be solved numerically. This leads to a computational bottleneck and creates practical limitations for reservoir size and utility. The lack of reservoir equations also presents problems for applying optimization procedures.

To overcome this, we present here a recursive analytical solution used to derive approximate virtual node equations. The solution is assessed in its computational capabilities as DCR and compared against numerical solvers in nonlinear benchmark tasks. We show that while computational performance is comparable, the analytical approximation leads to considerable savings in computation time, allowing the exploration of exceedingly large setups. Finally, we discuss the perspectives of this approach regarding optimization schemes in the fashion of previous work by the authors [9].

2 Methods

2.1 Single Node Delay-Coupled Reservoirs

In a DCR, past and present information undergoes nonlinear mixing via injection into a nonlinear node with delayed feedback. Formally, these dynamics can be modeled by a delay differential equation

$$\frac{dx(t)}{dt} = -x(t) + f(x(t-\tau), J(t)), \tag{1}$$

where τ is the delay time, J is the input driving the system, and f is a nonlinear function. For a DCR, system (1) can be operated in a simple regime that is governed by a single fixed point in case $J(t) = const.$

Injecting a signal into the reservoir is achieved by multiplexing it in time: The DCR receives a single constant input $u(\bar{t})$ in each reservoir time step $\bar{t} = \lceil \frac{t}{\tau} \rceil$, corresponding to one τ-cycle of the system. During each τ-cycle, the input is again linearly transformed by a mask that is piecewise constant for short periods θ_i, representing the spacing between sampling points of $i = 1, ..., N$ virtual nodes along the delay line. Here, the mask M is chosen to be binary with random mask bits $M_i \in \{-0.1, 0.1\}$, so that node i receives a weighted input $M_i u(\bar{t})$. The masking procedure effectively prevents the driven dynamics of the underlying system from saturating. Accordingly, the sampling point spacing satisfies $\sum_{i=1}^{N} \theta_i = \tau$.

A sample is read out at the end of each θ_i, yielding N predictor variables (virtual nodes) $x_i(\bar{t})$ per time step $\bar{t}$. Computations are performed on the predictors using a linear regression model for some scalar target time series y, given by $\hat{y}(\bar{t}) = \sum_{i=1}^{N} \alpha_i x_i(\bar{t})$, where x_i, $i = 1, ..., N$ denote the DCR's virtual nodes (see eq. (4)), and the α_i are the coefficients determined by regression, e.g. using the *least squares solution* minimizing the sum of squared errors, $\sum_{\bar{t}} (y(\bar{t}) - \hat{y}(\bar{t}))^2$.

2.2 Approximate Virtual Node Equations

In the following, we discuss a recursive analytical solution to equation (1), known as *method of steps*. The resulting formulas are used to derive a piecewise solution scheme for sampling points across τ that correspond to the reservoir's virtual nodes. Finally, we use the *trapezoidal rule* for further simplification, hereby deriving approximate virtual node equations, the temporal dependencies of which only consist of other virtual nodes. As will be shown in the remainder of this article, the resulting closed-form solutions allow reservoir computation without significant loss of performance as compared to a system obtained by explicit numerical solutions, e.g. *Heun's method* ((1,2) Runge-Kutta).

First, we discuss a simple application of the *method of steps*. For better readability, the argument $J(t)$ of the nonlinearity f is omitted in this part of the derivation. If system (1) is evaluated at $(i-1)\tau \leq t \leq i\tau$ (say $t_i = i\tau$), where a continuous function $\phi_i \in C_{[(i-2)\tau,(i-1)\tau]}$ is the solution for $x(t)$ on the previous τ-interval, we can replace $x(t-\tau)$ by $\phi_i(t-\tau)$. Consequently, elementary *variation of parameters* is applicable and yields directly the solution to the initial value problem in (1) with initial value $x(t_0 = (i-1)\tau) = \phi_i(t_{i-1})$, given by

$$\begin{aligned} x(t) &= \phi_i(t_{i-1})e^{t_{i-1}-t} + e^{t_{i-1}-t}\int_{(i-1)\tau}^{t} f(\phi_i(s-\tau))e^{s-t_{i-1}}ds \\ &= \phi_i(t_{i-1})e^{t_{i-1}-t} + e^{t_{i-1}-t}\int_{(i-2)\tau}^{t-\tau} f(\phi_i(u))e^{u-(i-2)\tau}du. \end{aligned} \tag{2}$$

Further, we use the *cumulative trapezoidal rule* [5] $\int_a^b g(x)dx = \frac{h}{2}g(\chi_0 = a) + h\sum_{j=1}^{n-1} g(\chi_j) + \frac{h}{2}g(\chi_n = b)$ to interpolate the integral in (2) piece-wise linearly along a uniform grid $(i-2)\tau = \chi_0 < ... < \chi_N = t - \tau$, where $\chi_{j+1} - \chi_j = h$. With $g(\chi_j) = e^{\chi_j-(i-2)\tau}f(\phi_i(\chi_j))$, this yields

$$\begin{aligned} x(t) \approx{}& \phi_i(t_{i-1})e^{t_{i-1}-t} + e^{t_{i-1}-t}\frac{h}{2}f(\phi_i(\chi_0)) \\ &+ e^{t_{i-1}-t}\frac{h}{2}\left(e^{\chi_N-(i-2)\tau}f(\phi_i(\chi_N)) + 2\sum_{j=1}^{N-1} e^{\chi_j-(i-2)\tau}f(\phi_i(\chi_j))\right). \end{aligned} \tag{3}$$

We are now interested in $1 \leq k \leq N$ single node equations $x_k(\bar{t})$, where $\bar{t} = i$ denotes discrete reservoir time step i in case $(i-1)\tau \leq t \leq i\tau$. Assuming equidistant virtual nodes where $\tau = N\theta$ and N the number of virtual nodes, we choose a uniform grid $\chi_j = (i-2)\tau + j\theta$ with $j = 0, ..., N$ (i.e. $h = \theta$). To get an expression for $x_k(\bar{t})$, we now have to evaluate equation (3) at the sampling point $t = (i-1)\tau + k\theta$, which results in

$$\begin{aligned} x_k(\bar{t}) &= x((i-1)\tau + k\theta) \\ &\approx e^{-k\theta}\phi_i((i-2)\tau + N\theta) + \frac{\theta}{2}e^{-k\theta}f[\phi_i((i-3)\tau + N\theta)] \\ &\quad + \frac{\theta}{2}f[\phi_i((i-2)\tau + k\theta)] + \theta\sum_{j=1}^{N-1} e^{(j-k)\theta}f[\phi_i((i-2)\tau + j\theta)] \end{aligned}$$

$$\begin{aligned}
&= e^{-k\theta} x_N(\bar{t}-1) + \frac{\theta}{2} e^{-k\theta} f[x_N(\bar{t}-2), J_N(\bar{t}-1)] \\
&\quad + \frac{\theta}{2} f[x_k(\bar{t}-1), J_k(\bar{t})] \\
&\quad + \sum_{j=1}^{k-1} \underbrace{\theta e^{(j-k)\theta}}_{c_{kj}} f[x_j(\bar{t}-1), J_j(\bar{t})]. \qquad (4)
\end{aligned}$$

Here $J_j(\bar{t})$ denotes the masked input $M_j u(\bar{t}) \in \mathbb{R}$ (see sec. 2.1) to node j at reservoir time step $\bar{t}$, which was omitted as an argument to f during the derivation to avoid cluttering. Note that equation (4) only has dependencies on sampling points corresponding to other virtual nodes. An exemplary coupling coefficient is indicated by c_{kj}, weighting a nonlinear coupling from node j to node k. We use this to derive weight matrices that allow simultaneous computation of all nodes in one reservoir time step $\bar{t}$ by a single vector operation, hereby dramatically reducing the computation time of simulating the system by several orders of magnitude as compared to an explicit second order numerical solver.

3 Results

We compare the analytical approximation of the system, derived in the previous section, to a numerical solution obtained using Heun's method with a stepsize of 0.1. The latter is chosen due to the relatively low computational cost and provides sufficient accuracy in the context of DCR computing. As a reference for absolute accuracy, we use numerical solutions obtained with *dde23* [7], an adaptive (2,3) Runge-Kutta based method for delay differential equations. The nonlinearity f is chosen according to the Mackey-Glass equation for the remainder of this paper, such that the system is given by

$$\dot{x}(t) = -x(t) + \frac{\eta(x(t-\tau) + \gamma J(t))}{1 + (x(t-\tau) + \gamma J(t))^p}, \qquad (5)$$

where η, γ and p are metaparameters, τ the delay length, and $J(t)$ is the temporally stretched input $u(\bar{t})$, multiplexed with a binary mask M.

Note that the trapezoidal rule used in the analytical approximation, as well as Heun's method, are both second order numerical methods that should yield a global truncation error of the same complexity class. As a result, discrepancies originating from different step sizes employed in the two approaches (e.g. 0.2 in the analytical approximation and 0.1 in the numerical solution) may be remedied by simply decreasing θ in the analytical approximation, for example by increasing N while keeping a fixed τ (see sec. 3.4).

3.1 Trajectory Comparison

In a first step, we wish to establish the general accuracy of the analytical approximation in a DCR relevant setup. Figure 1 shows a comparison of reservoir

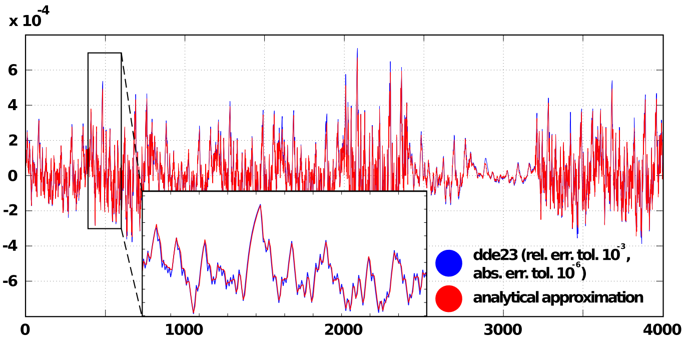

Fig. 1. Comparison between analytical approximation and numerical solution for an input-driven Mackey-Glass system with parameters $\eta = 0.4$, $\gamma = 0.05$ and $p = 1$, sampled at the temporal positions of virtual nodes, with a distance $\theta = 0.2$

trajectories computed with equation (4) (red) against trajectories computed numerically using *dde23* (blue) with relative error tolerance 10^{-3} and absolute error tolerance 10^{-6}. The systems received uniformly distributed input $u(\bar{t}) \sim \mathcal{U}_{[0,0.5]}$. The sample points correspond to the activities of $N = 400$ virtual nodes with a temporal distance of $\theta = 0.2$, and $\tau = 80$ accordingly. Given 4000 samples (corresponding to 10 reservoir time steps $\bar{t}$), the mean squared error between the trajectories is $MSE = 5.4 \times 10^{-10}$. As can be seen in the figure, the trajectories agree very well in the fixed point regime of the system (autonomous case). Although it is expected that the MSE would increase in more complex dynamic regimes (e.g. chaos), the latter are usually not very suitable for a DCR for various reasons. The following results also show a high task performance of the analytical approximation when used for DCR computing.

3.2 NARMA-10

A widely used benchmark in reservoir computing is the capacity of the DCR to model a nonlinear autoregressive moving average system y in response to uniformly distributed scalar input $u(k) \sim \mathcal{U}_{[0,0.5]}$. The NARMA-10 task requires the DCR to compute at each time step k a response

$$y(k+1) = 0.3y(k) + 0.05y(k)\sum_{i=0}^{9} y(k-i) + 1.5u(k)u(k-9) + 0.1.$$

Thus, NARMA-10 requires modeling of quadratic nonlinearities and shows a strong history dependence that challenges the DCR's memory capacity. We measure performance in this task using the correlation coefficient $r(y, \hat{y}) \in [-1, 1]$ between the target time series y and the DCR output $\hat{y}$ in response to u. Here, the DCR is trained (see sec. 2.1) on 3000 data samples, while $r(y, \hat{y})$ is computed on an independent validation data set of size 1000. Figure 2A summarizes the performance of 50 different trials for a DCR computed using the analytical approximation (see eq. 4), shown in red, as compared to a DCR simulated with Heun's method, shown in blue. Both reservoirs consist of $N = 400$ virtual nodes, evenly spaced with a distance $\theta = 0.2$ along a delay line $\tau = 80$. Both systems show a comparable performance across the 50 trials, with a median correlation coefficient between $r(y, \hat{y}) = 0.96$ and 0.97, respectively.

3.3 5-Bit Parity

As a second benchmark, we chose the delayed 5-bit parity task [6], requiring the DCR to handle binary input sequences on which strong nonlinear computations have to be performed with arbitrary history dependence. Given a random input sequence u with $u(k) \in \{-1, 1\}$, the DCR has to compute at each time step k the parity $p_m^\delta(k) = \prod_{i=0}^m u(k - i - \delta) \in \{-1, 1\}$, for $\delta = 0, ..., \infty$. The *performance* ϕ_m is then calculated on n data points as $\phi_m = \sum_{\delta=0}^\infty \kappa_m^\delta$, where *Cohen's Kappa*

$$\kappa_m^\delta = \frac{\frac{1}{n}\sum_{k=1}^n \max(0, p_m^\delta(k)\hat{y}(k)) - p_c}{1 - p_c} \in \{0, 1\}$$

normalizes the average number of correct DCR output parities $\hat{y}$ by the chance level $p_c = 0.5$. We used 3000/1000 data points in training and validation set respectively. To compare performance between analytical approximation and numerical solution of the DCR, we chose $m = 5$ and truncated ϕ_m at $\delta = 7$, such that $\phi_5 \in [0, 7]$. For parameters $\eta = 0.24$, $\gamma = 0.32$ and $p = 1$, and a DCR comprised of 400 neurons ($\tau = 80$), figure 2B shows that performance ϕ_5 is comparable for both versions of the DCR, with median performances between 4.3 and 4.5. across 50 different trials of this task. As the performance is far from the ideal value of 7 and the model suffers slightly from overfitting (not shown), it is clear that the delayed 5-bit parity task is a hard problem which leaves much space for improvement.

3.4 Large Setups

We repeated the tasks in larger network setups where the computational cost of the numerical solver becomes prohibitive. In addition to increasing the number of virtual nodes N one can also decrease the node distance θ, thus fitting more nodes into the same delay span τ. Although too small θ may affect a virtual node's computation negatively, decreasing θ increases the accuracy of the analytical approximation.

NARMA-10. We illustrate this by repeating the NARMA-10 task with $N = 2000$ virtual nodes and $\tau = 200$. This results in $\theta = 0.1$, corresponding to the step size used in the numerical solution before. Note that this hardly increases the computational cost of the analytical approximation since the main simulation loop along reservoir time steps $\bar{t}$ (τ-cycles) remains unchanged. L^2-regularization is employed to manage the large number of predictors. The results are summarized for 50 trials in figure 2C (right boxplot). The median correlation coefficient

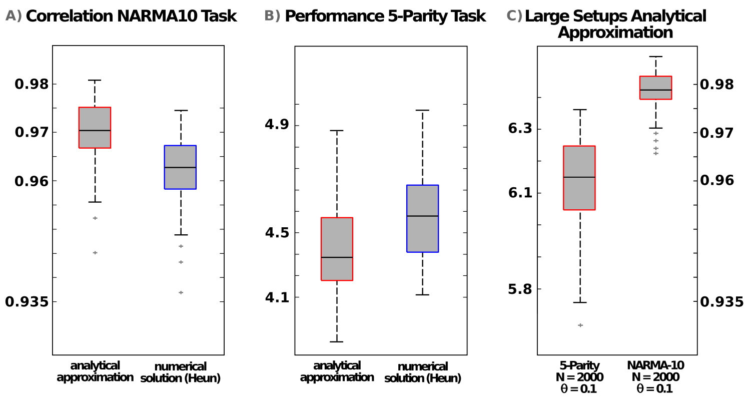

Fig. 2. Comparison on nonlinear tasks between analytical approximation and numerical solution for an input-driven Mackey-Glass system, sampled at the temporal positions of virtual nodes with a distance $\theta = 0.2$. Mackey-Glass parameters are $\eta = 0.4$, $\gamma = 0.05$ and $p = 1$ (NARMA-10) and $\eta = 0.24$, $\gamma = 0.32$ and $p = 1$ (5-bit parity), respectively. Results are reported for 400 neurons ($\tau = 80$) on data sets of size 3000/1000 (training/validation) in figures 2A and 2B, size 3000/1000 in 2C (right plot), as well as for data sets of size 10000/10000 in figure 2C (left plot). Each plot is generated from 50 different trials. The plots show median (black horizontal bar), 25th/75th percentiles (boxes), and most extreme data points not considered outliers (whiskers).

increased significantly to nearly 0.98 while the variance across trials is notably decreased (compare fig. 2A).

5-Bit Parity. For the 5-bit parity task, we addressed the task complexity by increasing both, training and validation sets, to a size of 10000. Second, we increased once more the virtual network size to $N = 2000$ virtual nodes and $\tau = 200$. The performance of the resulting DCR setup, computed across 50 trials using the analytical approximation, is summarized in figure 2C (left boxplot). The model no longer suffers as much from overfitting and the performance on the validation set increased dramatically to a median value of 6.15, which is now close to the theoretical limit of 7. While the computation to produce figure 2C took only few minutes with the analytical approximation, it is estimated that the use of the numerical solver for the same computation would have exceeded 2 days, despite the large step size of 0.1.

4 Discussion

In summary, we have developed analytical alternatives to evaluate and approximate solutions of delay differential equations that can be used for delay-coupled reservoir computing. It is shown that the resulting update equations in principle lose neither accuracy with respect to the system dynamics nor computational power in DCR benchmark tasks. Using the analytical approximation reduces computational costs considerably. This enabled us to study larger networks of delay-coupled nodes, yielding a dramatic increase in nonlinear benchmark performance. These results can lead to serious improvement regarding the implementation of DCRs on electronic boards.

Moreover, the approach yields an explicit handle on the DCR components which are otherwise implicit in equation (1). This creates new possibilities to investigate delay-coupled reservoirs and provides the basis for optimization schemes, a crucial necessity prior to any hardware implementation. Together with the reduction in computation time, this makes the use of supervised batch-update algorithms feasible to directly optimize model metaparameters (see eq. (5)) instead of conducting costly parameter scans. In addition, the optimization may include unsupervised gradient descent schemes on DCR parameters (e.g. θ, τ, N) with respect to information theoretic objectives. It is also straight forward to extend eq. (4) to account for nonuniform node spacings θ_i, subject to individual optimization (compare [9]). Continuing this line of thought, it is now possible to modify the update equations directly according to self-organizing homeostatic principles, inspired, for example, by neuronal plasticity mechanisms (e.g. [3]). We intend to explore these possibilities further in future work to maximize the system's computational power and render it adaptive to information content in task-specific setups.

References

[1] Appeltant, L., Soriano, M.C., Van der Sande, G., Danckaert, J., Massar, S., Dambre, J., Schrauwen, B., Mirasso, C.R., Fischer, I.: Information processing using a single dynamical node as complex system. Nature Communications 2, 468 (2011)

[2] Herbert Jäger. The " echo state " approach to analysing and training recurrent neural networks. Technical report (2001)

[3] Lazar, A., Pipa, G., Triesch, J.: SORN: a self-organizing recurrent neural network. Frontiers in Computational Neuroscience 3 (2009)

[4] Maass, W., Natschläger, T., Markram, H.: Real-time computing without stable states: a new framework for neural computation based on perturbations. Neural Computation 14(11), 2531–2560 (2002)

[5] Quarteroni, A., Sacco, R., Saleri, F.: Numerical Mathematics, 2nd edn. Texts in Applied Mathematics, vol. 37. Springer, Berlin (2006)

[6] Schrauwen, B., Buesing, L., Legenstein, R.A.: On computational power and the order-chaos phase transition in reservoir computing. In: Koller, D., Schuurmans, D., Bengio, Y., Bottou, L. (eds.) NIPS, pp. 1425–1432. Curran Associates, Inc. (2008)

[7] Shampine, L.F., Thompson, S.: Solving ddes in matlab. In: Applied Numerical Mathematics, vol. 37, pp. 441–458 (2001)

[8] Soriano, M.C., Ortín, S., Brunner, D., Larger, L., Mirasso, C.R., Fischer, I., Pesquera, L.: Optoelectronic reservoir computing: tackling noise-induced performance degradation. Optics Express 21(1), 12–20 (2013)

[9] Toutounji, H., Schumacher, J., Pipa, G.: Optimized Temporal Multiplexing for Reservoir Computing with a Single Delay-Coupled Node. In: The 2012 International Symposium on Nonlinear Theory and its Applications, NOLTA 2012 (2012)