SCIENTIFIC REPORTS

OPEN

Neuromorphic photonic networks using silicon photonic weight banks

Alexander N. Tait, Thomas Ferreira de Lima, Ellen Zhou, Allie X. Wu, Mitchell A. Nahmias, Bhavin J. Shastri & Paul R. Prucnal

Received: 16 February 2017
Accepted: 29 June 2017
Published online: 07 August 2017

Photonic systems for high-performance information processing have attracted renewed interest. Neuromorphic silicon photonics has the potential to integrate processing functions that vastly exceed the capabilities of electronics. We report first observations of a recurrent silicon photonic neural network, in which connections are configured by microring weight banks. A mathematical isomorphism between the silicon photonic circuit and a continuous neural network model is demonstrated through dynamical bifurcation analysis. Exploiting this isomorphism, a simulated 24-node silicon photonic neural network is programmed using "neural compiler" to solve a differential system emulation task. A 294-fold acceleration against a conventional benchmark is predicted. We also propose and derive power consumption analysis for modulator-class neurons that, as opposed to laser-class neurons, are compatible with silicon photonic platforms. At increased scale, Neuromorphic silicon photonics could access new regimes of ultrafast information processing for radio, control, and scientific computing.

Light forms the global backbone of information transmission yet is rarely used for information transformation. Digital optical logic faces fundamental physical challenges[1]. Many analog approaches have been researched[2–4], but analog optical co-processors have faced major economic challenges. Optical systems have never achieved competitive manufacturability, nor have they satisfied a sufficiently general processing demand better than digital electronic contemporaries. Incipient changes in the supply and demand for photonics have the potential to spark a resurgence in optical information processing.

A germinating silicon photonic integration industry promises to supply the manufacturing economomies normally reserved for microelectronics. While firmly rooted in demand for datacenter transceivers[5], the industrialization of photonics would impact other application areas[6]. Industrial microfabrication ecosystems propel technology roadmapping[7], library standardization[8,9], and broadened accessibility[10], all of which could open fundamentally new research directions into large-scale photonic systems. Large-scale beam steerers have been realized[11], and on-chip communication networks have been envisioned[12–14]; however, opportunities for scalable silicon photonic information processing systems remain largely unexplored.

Concurrently, photonic devices have found analog signal processing niches where electronics can no longer satisfy demands for bandwidth and reconfigurability. This situation is exemplified by radio frequency (RF) processing, in which front-ends have come to be limited by RF electronics, analog-to-digital converters (ADCs), and digital signal processors (DSP)[15,16]. In response, RF photonics has offered respective solutions for tunable RF filters[17,18], ADC itself[19], and simple processing tasks that can be moved from DSP into the analog subsystem[20–22]. RF photonic circuits that can be transcribed from fiber to silicon are likely to reap the economic benefits of silicon photonic integration. In a distinct vein, an unprecedented possibility for large-scale photonic system integration could enable systems beyond what can be considered in fiber. If scalable information processing with analog photonics is to be considered, new standards relating physics to processing must be developed and verified.

Standardized concepts that link physics to computational models are required to define essential quantitative engineering tools, namely metrics, algorithms, and benchmarks. For example, a conventional gate has simultaneous meaning as an abstract logical operation and as an arrangement of electronic semiconductors and thereby acts as a conduit between device engineering and computational performance. In another case, neuromorphic electronics adopt unconventional standards defining spiking neural networks as event-based packet networks[23–25]. These architectures' adherence to neural network models unlocks a wealth of metrics[26], algorithms[27,28], tools[29,30], and benchmarks[31] developed specifically for neural networks. Likewise, scalable information processing with analog photonics would rely upon standards defining the relationship between photonic physics and a suitable processing model. Neural networks are among the most well-studied models for information processing with

Department of Electrical Engineering, Princeton University, Princeton, New Jersey, 08544, USA. Correspondence and requests for materials should be addressed to A.N.T. (email: atait@princeton.edu)

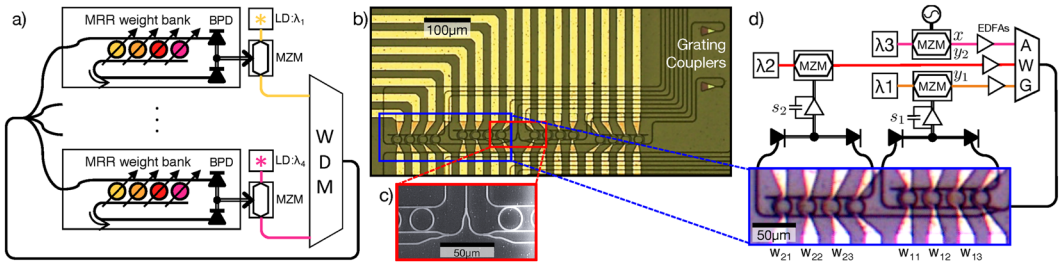

Figure 1. Broadcast-and-weight protocol and experiment. (**a**) Concept of a broadcast-and-weight network with modulators used as neurons. MRR: microring resonator, BPD: balanced photodiode, LD: laser diode, MZM: Mach-Zehnder modulator, WDM: wavelength-division multiplexer. (**b**) Micrograph of 4-node recurrent broadcast-and-weight network with 16 tunable microring (MRR) weights and fiber-to-chip grating couplers. (**c**) Scanning electron micrograph of 1:4 splitter. (**d**) Experimental setup with two off-chip MZM neurons and one external input. Signals are wavelength-multiplexed in an arrayed waveguide grating (AWG) and coupled into a 2×3 subnetwork with MRR weights, w_{11}, w_{12}, etc. Neuron state is represented by voltages s_1 and s_2 across low-pass filtered transimpedance amplifiers, which receive inputs from the balanced photodetectors of each MRR weight bank.

distributed analog elements. The fact that distributed interconnection and analog dynamics are performance strongsuits of photonic physics motivates the study of neuromorphic photonics.

"Broadcast-and-weight"[32] was proposed as a standard protocol for implementing neuromorphic processors using integrated photonic devices. Broadcast-and-weight is potentially compatible with mainstream silicon photonic device platforms, unlike free-space optical neural networks[4,33]. It is capable of implementing generalized reconfigurable and recurrent neural network models. In the broadcast-and-weight protocol, shown in Fig. 1(a), each neuron's output is assigned a unique wavelength carrier that is wavelength division multiplexed (WDM) and broadcast. Incoming WDM signals are weighted by reconfigurable, continuous-valued filters called photonic weight banks and then summed by total power detection. This electrical weighted sum then modulates the corresponding WDM carrier through a nonlinear or dynamical electro-optic process. Previous work on microring (MRR) weight banks have established a correspondence between weighted addition operations and integrated photonic filters. In reference to the operation, MRR weight bank scalability[34] and accuracy[35] metrics can be defined, but MRR weight banks have not been demonstrated within a network.

In this manuscript, we demonstrate a broadcast-and-weight system configured by microring weight banks that is isomorphic to a continuous-time recurrent neural network (CTRNN) model. As opposed to "brain-inspired" and "neuro-mimetic", "neuromorphic" is an unambiguous mathematical concept meaning that a physical system's governing equations are isomorphic to those describing an abstract neural network model. Isomorphic dynamical systems share qualtitative changes in dynamics as a result of parameter variation. We adopt a strategy for proving neuromorphism experimentally by comparing these dynamical transitions (a.k.a. bifurcations) induced in an experimental device against those predicted of a CTRNN model. In particular, we observe single-node bistability across a cusp bifurcation and two-node oscillation across a Hopf bifurcation. While oscillatory dynamics in optoelectronic devices have long been studied[36,37], this work relies on configuring an analog photonic network that can be scaled to more nodes in principle. This implies that CTRNN metrics, simulators, algorithms, and benchmarks can be applied to larger neuromorphic silicon photonic systems. To illustrate the significance of this implication, we simulate a 24-modulator silicon photonic CTRNN solving a differential equation problem. The system is programmed by appropriating an existing "neural compiler"[29] and benchmarked against a conventional CPU solving the same problem, predicting an acceleration factor of 294×.

Results

The CTRNN model is described by a set of ordinary differential equations coupled through a weight matrix.

$$\frac{d\mathbf{s}(t)}{dt} = \mathbf{W}\mathbf{y}(t) - \frac{\mathbf{s}(t)}{\tau} + \mathbf{w}_{in}u(t) \tag{1}$$

$$\mathbf{y}(t) = \sigma[\mathbf{s}(t)] \tag{2}$$

where $\mathbf{s}(t)$ are state variables with timeconstants τ, $\mathbf{W}$ is the recurrent weight matrix, $\mathbf{y}(t)$ are neuron outputs, $\mathbf{w}_{in}$ are input weights, and $u(t)$ is an external input. σ is a saturating transfer function associated with each neuron. Figure 1(b,c) shows the integrated, reconfigurable analog network and experimental setup. Signals u and $\mathbf{y}$ are physically represented as the power envelope of different optical carrier wavelengths. The weight elements of $\mathbf{W}$ and $\mathbf{w}_{in}$ are implemented as transmission values through a network of reconfigurable MRR filters. The neuron transfer function, σ, is implemented by the sinusoidal electro-optic transfer function of a fiber Mach-Zehnder modulator (MZM). The neuron state, s, is the electrical voltage applied to the MZM, whose timeconstant, τ, is determined by an electronic low-pass filter. We aim to establish a correspondence between experimental bifurcations induced by varying MRR weights and the modeled bifurcations[38] derived in Supplementary Section 1.

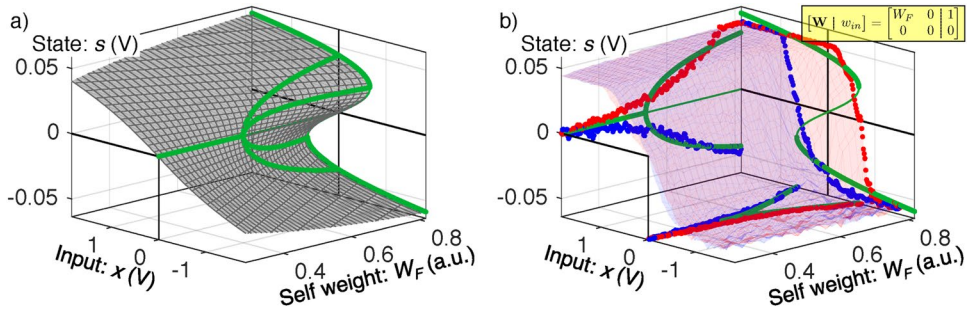

Figure 2. A cusp bifurcation in a single node with feedback weight, W_F, external input, u, and neurosynaptic state, s. (**a**) Theoretical model surface (gray) and bifurcation curves (green) plotted in 3D. Parameters of the model are fit to data. (**b**) Experimental data for increasing (blue surface) and decreasing (red surface) input. Theoretical bifurcation curves – with parameters identical to those in (**a**) – are projected onto 2D axes. The data surfaces are sliced at the planes: $u = 0$, $s = 0$, and $W_F = 0.08$, and similarly projected onto the axes (red and blue points) to illustrate the reproduction of pitchfork, cusp, and saddle-node bifurcations, respectively.

Cusp Bifurcation. A cusp bifurcation characterizes the onset of bistability in a single node with self-feedback. To induce and observe a cusp, the feedback weight of node 1 is parameterized as $w_{11} = W_F$. The neurosynaptic signal, s_1, is recorded as the feedback weight is swept through 500 points from 0.05–0.85, and the input is swept in the rising (blue surface) and falling (red surface) directions. The parameters τ, α, and κ in equation (S1.1) are fit to minimize root mean squared error between model and data surfaces. The best fit model has a cusp point at $W_B = 0.54$. Figure 2(a) shows a modeled cusp surface described by equation (S1.1) and fit to data from Fig. 2(b). In Fig. 2(b), the data surfaces are interpolated at particular planes and projected onto 2D axes as red/blue points. The corresponding slices of the model surface are similarly projected as green lines. The $u = 0$ slice projected on the $s - W_F$ plane yields a pitchfork curve described by equation (S1.2). The $W_F = 0.08$ slice projected on the $u - s$ plane yields the bistable curve described by equation (S1.3). Finally, the $s = 0$ slice projected on the $u - W_F$ plane yields the cusp curve described by equation (S1.4).

The experimental reproduction of pitchfork, bistable, and cusp bifurcations is demonstrative of an isomorphism between the single-node model and the device under test. An opening of an area between rising and falling data surfaces is characteristic of bistability. The transition boundary closely follows a cusp form. While the pitchfork and bistable slices reproduce the number and growth trends of fixed points, their fits have non-idealities. These non-idealities can be attributed to a hard saturation of the electrical transimpedance amplifier when the input voltage and feedback weight are high. Furthermore, the stability of cusp measurements serve as a control indicating the absence of time-delayed dynamics resulting from long fiber delays and causing spurious oscillations[39]. Electrical low-pass filtering is used to eliminate these unmodeled dynamics in order to observe modeled bifurcations.

Hopf Bifurcation. Dynamical systems are capable of oscillating if there exists a closed orbit (a.k.a. limit cycle) in the state space, which must therefore exceed one dimension. The Hopf bifurcation occurs when a stable fixed-point becomes unstable while giving rise to a stable limit cycle. Hopf bifurcations are further characterized by oscillations that approach zero amplitude and nonzero frequency near the bifurcation point[38]. We induce a Hopf bifurcation by configuring the MRR weight matrix to have asymmetric, fixed off-diagonals and equal, parameterized diagonals.

Figure 3 compares the observed and predicted oscillation onset, amplitude, and frequency. Figure 3(a–c) show the time traces for below, near, and above the oscillation threshold. Above threshold, oscillation occurs in the range of 1–5 kHz, as limited by electronic low-pass filters and feedback delay. Figure 3(d) shows the result of a fine sweep of self-feedback weights in the 2-node network, exhibiting the paraboloid shape of a Hopf bifurcation. The voltage of neuron 1 is plotted against that of neuron 2 with color corresponding to W_F parameter. The peak oscillation amplitude for each weight is then projected onto the $W_F - y_2$ plane in black, and these amplitudes are fit using the model from equation (S1.8) (red). Bifurcation occurs at $W_B = 0.48$ in the fit model. Figure 3(e) plots the oscillation frequency above the Hopf point. Data are discarded for $W_B < W_F < 0.53$ because the oscillations are erratic in the sensitive transition region. Frequency data are then fit with the model of equation (S1.10). The frequency axis is scaled so that 1.0 corresponds to the model frequency at region boundary, which is 4.81 kHz. The Hopf bifurcation only occurs in systems of more than one dimension, thus confirming the observation of a small integrated photonic neural network.

Significantly above the bifurcation point, experimental oscillation amplitude and frequency closely match model predictions, but discrepancies are apparent in the transition regime. Limit cycles with amplitudes comparable to noise amplitude can be destabilized by their proximity to the unstable fixed point at zero. This effect could explain the middle inset of Fig. 3, in which a small oscillation grows and then shrinks. Part of this discrepancy can be explained by weight inaccuracy due to inter-bank thermal cross-talk. The two MRR weight banks were calibrated independently accounting only for intra-bank thermal cross-talk. As seen in Fig. 1(c), the physical distance between w_{12} (nominally -1) and w_{22} (nominally W_F) is approximately 100 μm. While inter-bank cross-talk is not a major effect, w_{12} is very sensitive because weight -1 corresponds to on-resonance, and the dynamics are especially sensitive to the weight values near the bifurcation point.

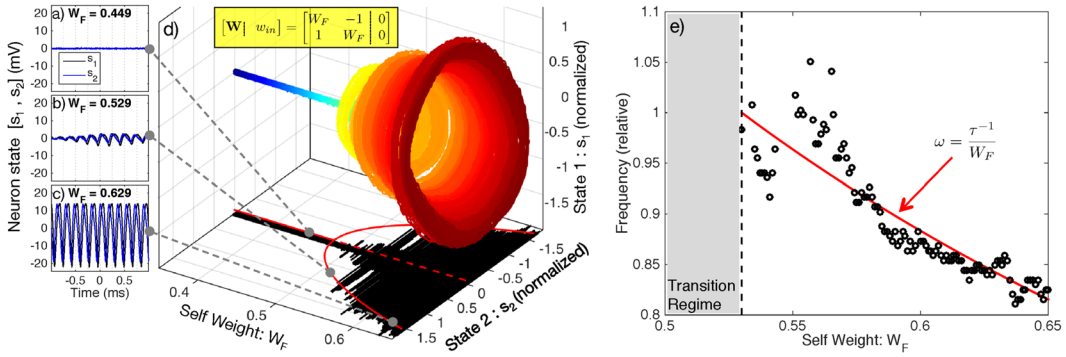

Figure 3. A Hopf bifurcation between stable and oscillating states. (**a**–**c**) Time traces below, near, and above the bifurcation. (**d**) Oscillation growth versus feedback weight strength. Color corresponds to feedback weight parameter, W_F, to improve visibility. Black shadow: average experimental amplitudes; solid red curve: corresponding fit model; dotted red line: unstable branch. (**e**) Frequency of oscillation above the Hopf bifurcation. The observed data (black points) are compared to the expected trend of equation (S1.10) (red curve). Frequencies are normalized to the threshold frequency of 4.81 kHz.

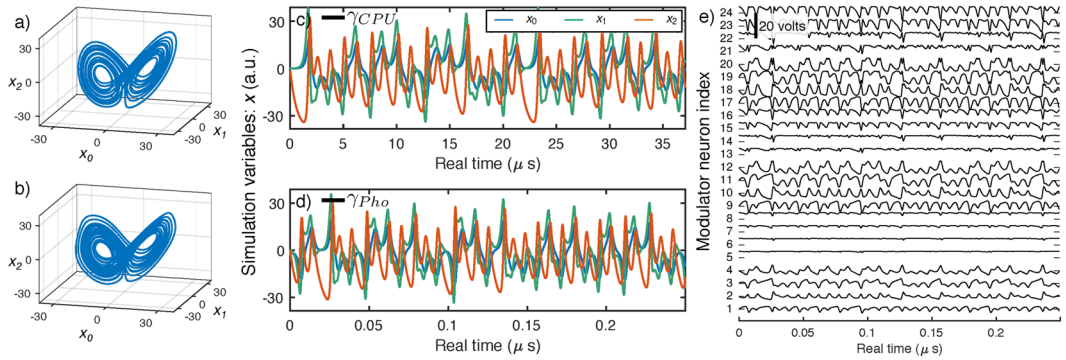

Figure 4. Photonic CTRNN benchmarking against a CPU. (**a**,**b**) Phase diagrams of the Lorenz attractor simulated by a conventional CPU (**a**) and a photonic CTRNN (**b**). (**c**,**d**) Time traces of simulation variables for a conventional CPU (**c**) and a photonic CTRNN (**d**). The horizontal axes are labeled in physical real time, and cover equal intervals of virtual simulation time, as benchmarked by γ_{CPU} and γ_{Pho}. The ratio of real-time values of γ's indicates a 294-fold acceleration. (**e**) Time traces of modulator voltages s_i (minor y-axis) for each modulator neuron i (major y-axis) in the photonic CTRNN. The simulation variables, **x**, in (**d**) are linear decodings of physical variables, **s**, in (**e**).

Emulation Benchmark. A dynamical isomorphism between a silicon photonic system and the CTRNN model of equations (1 and 2) implies that larger, faster neuromorphic silicon photonic systems could utilize algorithms and tools developed for generic CTRNNs. Here, we apply a "neural compiler" called the Neural Engineering Framework (NEF)[40] to program a simulated photonic CTRNN to solve an ordinary differential equation (ODE). This simulation is benchmarked against a conventional central processing unit (CPU) solving the same task. The procedures for each approach are detailed in Methods. As opposed to implementation-specific metrics, benchmarks are task-oriented indicators suitable for comparing technologies that use disparate computing standards. Benchmarking approaches are therefore needed to evaluate the potentials of any unconventional processor in application domains currently served by conventional processors. The chosen benchmark problem consists of solving a well-known ODE called the Lorenz attractor, described by a system of three coupled ODEs with no external inputs:

$$\begin{aligned} \gamma\dot{x}_0 &= \upsilon_*(x_1 - x_0) \\ \gamma\dot{x}_1 &= -x_0x_2 - x_1 \\ \gamma\dot{x}_2 &= x_0x_1 - \beta(x_2 + \rho) - \rho \end{aligned} \tag{3}$$

where **x** are the simulation state variables, and γ is a time scaling factor. When parameters, υ, β, and ρ, are set to $(\upsilon, \beta, \rho) = (6.5, 8/3, 28)$, the solutions of the attractor are chaotic. The photonic CTRNN and CPU solutions are compared in **x** phase space in Fig. 4(a,b) and the time-domain in Fig. 4(c,d). Figure 4(e) plots the physical modulator voltages, **s**, linear combinations of which represent simulation variables, **x**, as discussed in Methods. Because the two simulators are implemented differently, they cannot be compared based on equivalent metrics;

however, the time scaling factor, γ, links physical real-time to a virtual simulation time basis, in which a direct comparison can be made.

Figure 4(c,d) plots photonic CTRNN and CPU solutions in the real-time bases, scaled to cover equal simulation intervals. The discrete-time simulation is linked to physical real-time by the step calculation time, $\Delta t = 24.5$ ns, and its stability is limited by numerical divergence. We find that $\gamma_{CPU} \geq \Delta t \times 150$ is sufficient for $<1\%$ divergence probability, resulting in $\gamma_{CPU} = 3.68\,\mu$s. The CTRNN simulation is linked to physical real-time by its feedback delay, $t_{fb} = 47.8$ ps, and its stability is limited by time-delayed dynamics. We find that $\gamma_{Pho} \geq t_{fb} \times 260$ is sufficient to avoid spurious dynamics, resulting in $\gamma_{Pho} = 12.5$ ns. The acceleration factor, $\gamma_{CPU}/\gamma_{Pho}$, is thus predicted to be $294\times$.

Implementing this network on a silicon photonic chip would require 24 laser wavelengths, 24 modulators, and 24 MRR weight banks, for a total of 576 MRR weights. The power used by the CTRNN would be dominated by static MRR tuning and pump lasers,as discussed in the Methods section. Minimum pump power is determined by the requirement that recurrently connected neurons are able to exhibit positive eigenvalues. Considering 24 lasers with a realistic wall-plug efficiency of 5%, minimum total system power is expected to be 106 mW. The area used by the photonic CTRNN is split evenly between MRR weight banks ($576 \times 25\,\mu\text{m} \times 25\,\mu\text{m} = 0.36\,\text{mm}^2$) and modulators[41] ($24 \times 500\,\mu\text{m} \times 25\,\mu\text{m} = 0.30\,\text{mm}^2$). The fundamental limits of these performance metrics are compared with other neuromorphic approaches below.

While the qualitative Lorenz behavior is reproduced by CTRNN and CPU implementations, the chaotic nature of the attractor presents a challenge for benchmarking emulation accuracy. Non-chaotic partial differential equations (PDEs) exist to serve as accuracy benchmarks[42,43]; however, most non-trivial ODEs are chaotic. One exception is work on central pattern generators (CPGs) that are used to shape oscillations for driving mechanical locomotion[44]. CPGs have been implemented with analog ASICs[45] and digital FPGAs[46]. While work on CPG hardware has fallen in sub-kHz timescales, similar tasks could be developed for GHz timescales with possible application to adaptive RF waveform generation. Further work could develop CPG-like tasks to benchmark the accuracy of photonic CTRNN ODE emulators.

Accuracy can be assessed through metrics of weight accuracy. Previous work discussed the precision and accuracy to which MRR weight banks could be configured to a desired weight vector[35]. Even in the presence of fabrication variation and thermal cross-talk, the dynamic weight accuracy was demonstrated to be 4.1 + 1(sign) bits and anticipated to increase with improved design. Since the weight is analog, its dynamic accuracy (range/command error) is not an integer; however, this metric corresponds to bit resolution in digital architectures. In the majority of modern-day neuromorphic hardware, this resolution is selected to be 4 + 1(sign) bits[24,47,48] as a tradeoff between hardware resources and functionality. Significant study has been devoted to the effect of limited weight resolution on neural network function[49] and techniques for mitigating detrimental effects[50].

Neuromorphic photonic metrics. In addition to task-driven benchmark analyses, we can perform component-level metric comparisions with other neuromorphic systems in terms of area, signal bandwidth, and synaptic operation (SOP) energy. The 24-modulator photonic CTRNN used as an emulation benchmark was predicted to use a 4.4 mW/neuron using realistic pump lasers, and was limited to 1 GHz bandwidth to spoil spurious oscillations. This results in a computational efficiency of 180 fJ/SOP. The area of an MRR weight is approximately $(l \times w)/N^2 = 25 \times 25 = 625\,\mu\text{m}^2$/synapse.

The coherent approach described by Shen *et al.*[51] based on a matrix of Mach-Zehnder interferometers (MZIs) would exhibit similar fundamental energy and speed limitations, given similar assumptions about detection bandwidth and laser efficiency. The power requirements of this approach were limited by nonlinear threshold activation, rather than signal cascadability. Supposing a laser efficiency of 5%, this was estimated to be around 20 mW/neuron. A 24-neuron system limited to 1 GHz bandwidth would therefore achieve 830 fJ/SOP. While there is no one-to-one correspondence between MZIs and synapses, there is still a quadratic area scaling relationship: $(l \times w)/N^2 = 200 \times 100 = 20{,}000\,\mu\text{m}^2$/synapse, as limited by thermal phase shifter dimension.

The superconducting optoelectronic approach described by Shainline *et al.*[52] would be optimized for scalability and efficiency, instead of speed. This can be attributed to the extreme sensitivity of cryogenic photodetectors, but this difference also limits signal bandwidth to 20 MHz. For a 700-neuron interconnect, the wall-plug efficiency is estimated to be around 20 fJ/SOP. Area was calculated to be $(l \times w)/N^2 = 1.4 \times 15 = 21\,\mu\text{m}^2$/synapse.

A metric analysis can extend to include neuromorphic electronics, although metrics do not necessarily indicate signal processing merit. Electronic and photonic neuromorphics are designed to address complementary types of problems. Akopyan *et al.*[24] demonstrated a chip containing 256 million synapses dissipating 65 mW. The signal bandwidth, determined by the effective tick, or timestep, is 1.0 kHz, and the chip area is 4.3 cm^2. This results in an effective 240 fJ/SOP and effective area of 6.0 μm^2/synapse. We note that TrueNorth is event-based, meaning explicit computation does not occur for every effective SOP, but only when the input to a synapse is nonzero.

Discussion

We have demonstrated an isomorphism between a silicon photonic broadcast-and-weight system and a reconfigurable CTRNN model through observations of predicted bifurcations. In addition to proof-of-concept, this repeatable method could be used to characterize the standard performance of single neurons and pairs of neurons within larger systems. Employing neuromorphic properties, we then illustrated a task-oriented programming approach and benchmark analysis. Similar analyses could assess the potentials of analog photonic processors against state-of-the-art conventional processors in many application domains. Here, we discuss the implications of these results in the broader context of information processing with photonic devices.

This work constitutes the first investigation of photonic neurons implemented by modulators, an important step towards silicon-compatible neuromorphic photonics. Interest in integrated lasers with neuron-like spiking behavior has flourished over the past several years[53,54]. Experimental work has so far focused on isolated neurons[55–57]

and fixed, cascadable chains[58,59]. The shortage of research on networks of these lasers might be explained by the challenges of implementing low-loss, compact, and tunable filters in the active III/V platforms required for laser gain. In some cases where fan-in is sensitive to input optical phase, it is also unclear how networking would occur without global laser synchronization. In contrast to lasers, Mach-Zehnder, microring, and electroabsorption modulators are all silicon-compatible. Modulator-class neurons are therefore a final step towards making complete broadcast-and-weight systems entirely compatible with silicon foundry platforms. While laser-class neurons with spiking dynamics present richer processing opportunities, modulator-class neurons would still possess the formidable repertoire of CTRNN functions.

In parallel with work on individual laser neurons, recent research has also investigated systems with isomorphisms to neural network models. A fully integrated superconducting optoelectronic network was recently proposed[52] to offer unmatched energy efficiency. While based on an exotic superconducting platform, this approach accomplishes fan-in using incoherent optical power detection in a way compatible with the broadcast-and-weight protocol. A programmable nanophotonic processor was recently studied in the context of deep learning[51]. Coherent optical interconnects exhibit a sensitivity to optical phase that must be re-synchronized after each layer. In the demonstration, optical nonlinearity and phase regeneration were performed digitally. Analog solutions for counteracting signal-dependent phase shifts induced by nonlinear materials[60] have not yet been proposed. Recurrent neural networks have been investigated in fiber[61]. While the current work employs fiber neurons, it is the first demonstration of a recurrent weight network that is integrated. Optical neural networks in free-space have also been investigated in the past[4] and recently[33]. Free-space systems occupy an extra dimension but can not necessarily use it for increased scalability. The volume between focal planes is used for diffractive evolution of the optical field and unused for network configuration. Spatial light modulators that configure the network are generally planar. Shainline *et al*. noted that integrated neuromorphic photonic systems could potentially be stacked to take advantage of a third dimension[52].

Reservoir computing techniques that take inspiration from certain brain properties (e.g. analog, distributed) have received substantial recent attention from the photonics community[62–65]. Reservoir techniques rely on supervised learning to discern a desired behavior from a large number of complex dynamics, instead of relying on establishing an isomorphism to a model. Neuromorphic and reservoir approaches differ fundamentally and possess complementary advantages. Both derive a broad repertoire of behaviors (often referred to as complexity) from a large number of physical degrees-of-freedom (e.g. optical intensities) coupled through interaction parameters (e.g. transmissions). Both offer means of selecting a specific, desired behavior from this repertoire using controllable parameters. In neuromorphic systems, network weights are *both* the interaction and controllable parameters, whereas, in reservoir computers, these two groups of parameters are separate. This distinction has two major implications. Firstly, the interaction parameters of a reservoir do not need to be observable or even repeatable from system to system. Reservoirs can thus derive complexity from physical processes that are difficult to model or reproduce, such as coupled amplifiers[66], coupled nonlinear MRRs[67], time-delayed dynamics in fibers[64], and fixed interferometric circuits[63]. Furthermore, they do not require significant hardware to control the state of the reservoir. Neuromorphic hardware has a burden to correspond physical parameters (e.g. drive voltages) to model parameters (e.g. weights), as was shown in this paper. Secondly, reservoir computers can only be made to elicit a desired behavior through instance-specific supervised training, whereas neuromorphic computers can be programmed *a priori* using a known set of weights. Because neuromorphic behavior is determined only by controllable parameters, these parameters can be mapped directly between different system instances, different types of neuromorphic systems, and simulations. Neuromorphic hardware can leverage existing algorithms (e.g. NEF) and virtual training results. Particular behaviors, fully determined by the virtual/hardware weights, are guaranteed to occur. Photonic RCs can of course be simulated; however, they have no corresponding guarantee that a particular hardware instance will reproduce a simulated behavior or that training will be able to converge to this behavior.

At increased scale, neuromorphic silicon photonic systems could be applied to unaddressed computational areas in scientific computing and RF signal processing. A key benefit of neuromorphic engineering is that existing algorithms can be leveraged. A subset of CTRNNs, Hopfield networks[68], have been used extensively in mathematical programming and optimization problems[27]. The ubiquity of PDE problems in scientific computing has motivated the development of analog electronic neural emulators[42]. Further work could explore the use of NEF to emulate discrete space points of PDEs. Neural algorithms for CTRNNs have been developed for real-time RF signal processing, including spectral mining[69], spread spectrum channel estimation[70], and arrayed antenna control[71]. There is insistent demand to implement these tasks at wider bandwidths using less power than possible with RF electronics. Additionally, methodologies developed for audio applications, such as noise mitigation[28], could conceivably be mapped to RF problems if implemented on ultrafast hardware. Unsupervised neural-inspired learning has been used with a single MRR weight bank for statistical analysis of multiple RF signals[72].

We have demonstrated a reconfigurable analog neural network in a silicon photonic integrated circuit using modulators as neuron elements. Network-mediated cusp and Hopf bifurcations were observed as a proof-of-concept of an integrated broadcast-and-weight system[32]. Simulations of a 24 modulator neuron network performing an emulation task estimated a 294× speedup over a verified CPU benchmark. Neural network abstractions are powerful tools for bridging the gap between physical dynamics and useful application, and silicon photonic manufacturing introduces opportunities for large-scale photonic systems.

Methods

Experimental Setup. Samples shown in Fig. 1(b) were fabricated on silicon-on-insulator (SOI) wafers at the Washington Nanofabrication Facility through the SiEPIC Ebeam rapid prototyping group[10]. Silicon thickness is 220 nm, and buried oxide (BOX) thickness is 3 μm. 500 nm wide WGs were patterned by Ebeam lithography and fully etched through to the BOX[73]. After a cladding oxide (3 μm) is deposited, Ti/W and Al layers are deposited.

Ohmic heating in Ti/W filaments causes thermo-optic resonant wavelength shifts in the MRR weights. The sample is mounted on a temperature stabilized alignment stage and coupled to a 9-fiber array using focusing sub-wavelength grating couplers[74]. The reconfigurable analog network consists of 2 MRR weight banks each with four MRR weights with 10 μm radii.

Each MRR weight bank is calibrated using a multi-channel protocol described in past work[35,75]: an offline measurement procedure is performed to identify models of thermo-optic cross-talk and MRR filter edge transmission. During this calibration phase, electrical feedback connections are disabled and the set of wavelength channels carry a set of linearly seperable training signals. After calibration, the user can specify a desired weight matrix, and the control model calculates and applies the corresponding electrical currents.

Weighted network outputs are detected off-chip, and the electrical weighted sums drive fiber Mach-Zehnder modulators (MZMs). Detected signals are low-pass filtered at 10 kHz, represented by capacitor symbols in Fig. 1(c). Low-pass filtering is used to spoil time-delayed dynamics that arise when feedback delay is much greater than the state time-constant[37]. In this setup with on-chip network and off-chip modulator neurons, fiber delayed dynamics would interfere with CTRNN dynamical analysis[39]. MZMs modulate distinct wavelengths $\lambda_1 = 1549.97$ nm and $\lambda_2 = 1551.68$ nm with neuron output signals $y_1(t)$ and $y_2(t)$, respectively. The MZM electro-optic transfer function serves as the nonlinear transfer function, $y = \sigma(s)$, associated with the continuous-time neuron. A third wavelength, $\lambda_3 = 1553.46$ nm, carries an external input signal, $x(t)$, derived from a signal generator. Each laser diode source (ILX 7900B) outputs +13 dBm of power. All optical signals (u, y_1, and y_2) are wavelength multiplexed in an arrayed waveguide grating (AWG) and then coupled back into the on-chip broadcast STAR consisting of splitting Y-junctions[76] (Fig. 1(c)).

Photonic CTRNN Solver. Recently developed compilers, such as Neural ENGineering Objects (Nengo)[77], employ the Neural Engineering Framework (NEF)[40] to arrange networks of neurons to represent values and functions without relying on training. While originally developed to evaluate theories of cognition, the NEF has been appropriated to solve engineering problems[78] and has been used to program electronic neuromorphic hardware[79]. Background on the NEF compilation procedure is provided in Supplementary Section 2. Simulation state variables, $\mathbf{x}$, are encoded as linear combinations of real population states, $\mathbf{s}$. Each neuron in a population has the same tuning curve shape, σ, but differ in gain g, input encoder vector $\mathbf{e}$, and offset b. The input-output relation of neuron i – equivalent to equation (2), is thus $s_i = \sigma(g_i \mathbf{e}_i \cdot \mathbf{x} + b_i)$. In this formulation, arbitrary nonlinear functions of the simulation variables, $\mathbf{f}(\mathbf{x})$, are represented by linear combinations of the set of these tuning curves across the domain of values of $\mathbf{x}$ considered. Introducing recurrent connections in the population introduces the notion of state time-derivatives, as in equation (1). By applying the decoder transform to both sides of equation (1) and using the arbitrary function mapping technique to find $\mathbf{W}$, the neural population emulates an effective dynamical system of the form $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$. Given equations (3) stated in this form, Nengo performs the steps necessary to represent the variables, functions, and complete ODE.

Modifications were made to the standard Nengo procedure. Firstly, we specify the tuning curve shape as the sinusoidal electro-optic transfer characteristic of a MZM. Secondly, to reduce the number of MZMs required, we choose encoders to be the vertices of a unit-square $\{e\} = [1, \pm 1, \pm 1]$, while they are typically chosen randomly. Thirdly, the MZM sinusoidal transfer function provides a natural relation to the Fourier basis. Gains are chosen to correspond to the first three Fourier frequencies of the domain: $g \in s_\pi/2 \cdot \{1, 2, 3\}$, where s_π is the MZM half-period. Offsets were chosen to be $b \in \{0, s_\pi/2\}$, corresponding to sine and cosine components of each gain frequency. The total number of modulator neurons is therefore $\#e \cdot \#g \cdot \#b = 4 \cdot 3 \cdot 2 = 24$. Figure 4(e) shows the MZM states, $s(t)$, of which simulation variables, $x(t)$, are linear combinations. From this plot, it appears that some neurons are barely used. Thus, further optimizations of number of neurons could be made by pruning those neurons after compilation of the weight matrix.

The operational speed of this network would be limited by time-of-flight feedback delay. In Fig. 1(a), the longest feedback path is via the drop port of the last (pink) MRR weight of the first (yellow) neuron's bank. The path includes the perimeter of the square MRR weight matrix, plus a drop waveguide folded back along the bank. Supposing a minimum MRR pitch of 25 μm and MZM length of 500 μm, the feedback delay would then be $(6 \times 25 \times 25 + 500) \cdot n/c = 48$ ps. We model this delayed feedback in the Nengo simulation and then adjust feedback strength to find the minimum stable simulation timescale. For $\gamma_{Pho}/t_{fb} < 65$, spurious time-delayed dynamics dominate. For $\gamma_{Pho}/t_{fb} < 104$, the butterfly phase diagram in Fig. 4(b) is not reproduced accurately. $\gamma_{Pho}/t_{fb} \geq 260$ is chosen for robust reproduction of the expected dynamics.

Conventional CPU Solver. Conventional digital processors must use a discrete-time approximation to simulate continuous ODEs, the simplest of which is Euler continuation:

$$\mathbf{x}[(n+1)\Delta t] = \mathbf{x}[n\Delta t] + \Delta t \mathbf{f}(\mathbf{x}[n\Delta t]) \tag{4}$$

where Δt is the time step interval. To estimate the real-time value of Δt, we develop and validate a simple CPU model. For each time step, the CPU must compute $\mathbf{f}(\mathbf{x}[n\Delta t])$ as defined in equation (3), resulting in 9 floating-point operations (FLOPs), and 12 cache reads of the operands. The Euler update in equation (4) constitutes one multiply, one addition, and one read/write for each state variable, resulting in 6 FLOPs and 6 cache accesses. Supposing a FLOP latency of 1 clock cycle, Level 1 (L1) cache latency of 4 cycles, and 2.6 GHz clock, this model predicts a time step of $\Delta t = 33$ ns. This model is empirically validated using an Intel Core i5-4288U. The machine-optimized program randomly initializes and loops through 10^6 Euler steps of the Lorenz system, over 100 trials. CPU time was measured to be $\Delta t = 24.5 \pm 1.5$ ns. The minimum stable simulation timescale is limited by divergent errors stemming from time discretization. We performed a series of 100 trials over 100 values of $\Delta t/\gamma_{CPU}$, finding that <1% probability of divergence occurred for $\gamma_{CPU}/\Delta t \geq 150$.

Minimum Power Calculations. Static thermal power must be applied to each weight in order to track MRRs to the on-resonance condition. Supposing a bank length set by an MRR pitch of 25 μm and count of 24, the MRR network would occupy a square with 600 μm sides. Within this length, resonances can be fabricated with repeatability within ±1.3 nm[80]. Supposing a tuning efficiency of 0.25 nm/mW[81], it would take an average of 5.2 mW/weight to track resonance, for a static power dissipation of 3.0 W. On the other hand, if depletion-based tuning can be used, there would be negligible static power dissipation in the weights.

The laser bias power must be set such that a modulator neuron can drive downstream neurons with sufficient strength. A neuron fed back to itself should be able to elicit an equivalent or greater small-signal response after one round-trip. This condition is referred to as signal cascadability and can be stated as $g \geq 1$, where g is round-trip, small-signal gain. If the cascadability condition is not met, all signals will eventually attenuate out with time. In other words, in a recurrent network, the real part of system eigenvalues would not be able to exceed zero. Round-trip gain is expressed as

$$g = \frac{dP_{out}}{dP_{in}} \tag{5}$$

For a modulator-based broadcast-and-weight system, this breaks down into receiver and modulator components. Assuming a voltage-mode modulator, such as reverse-biased MRR depletion modulator,

$$\left.\frac{dP_{out}}{dV_{mod}}\right|_{max} = \frac{\pi}{2V_{\pi}} P_{pump} \tag{6}$$

$$\frac{dV_{mod}}{dP_{in}} = R_{PD}R_r \tag{7}$$

where V_π is modulator π-voltage, P_{pump} is modulator pump power, and R_{PD} is detector responsivity. Because input power generates a photocurrent, yet a depletion modulator is voltage-driven, the receiver's impedance, R_r, determines the conversion and can be set externally. As R_r increases, round-trip gain also increases, but bandwidth decreases according to $f = (2\pi R_r C_{mod})^{-1}$, where C_{mod} is PN junction capacitance of the modulator. By setting the cascadability condition: $g = 1$ and combining the above equations, we find that

$$P_{pump}(R_r) = \frac{2V_{\pi}}{\pi R_{PD} R_r} \tag{8}$$

$$P_{pump}(f) = 4(V_{\pi} C_{mod}) R_{PD}^{-1} f \tag{9}$$

The values of V_π, C_{mod}, and R_{PD} on a typical silicon photonic foundry platform have been published[41]. For an MRR depletion modulator, $V_\pi = 1.5$ V, $C_{mod} = 35$ fF. For a PD on the same platform, $R_{PD} = 0.97$ A/W. This means that the minimum pump power for a given signal bandwidth is 2.2×10^{-13} W/Hz.

In this paper, we study a 24-node CTRNN whose signal bandwidth is restricted to 1 GHz to avoid time-delay dynamics. This means that, for the cascadability condition to be met, modulator pumping must be at least 0.22 mW/neuron of optical power. Adding up 24 lasers and accounting for laser inefficiency, wall-plug system power would be 106 mW.

References

1. Keyes, R. W. Optical logic-in the light of computer technology. *Optica Acta: International Journal of Optics* **32**, 525–535 (1985).
2. Reimann, O. A. & Kosonocky, W. F. Progress in optical computer research. *IEEE Spectrum* **2**, 181–195 (1965).
3. McCormick, F. B. *et al*. Six-stage digital free-space optical switching network using symmetric self-electro-optic-effect devices. *Appl. Opt.* **32**, 5153–5171 (1993).
4. Jutamulia, S. & Yu, F. Overview of hybrid optical neural networks. *Optics & Laser Technology* **28**, 59–72 (1996).
5. Vlasov, Y. Silicon CMOS-integrated nano-photonics for computer and data communications beyond 100 G. *IEEE Commun. Mag.* **50**, s67–s72 (2012).
6. Hochberg, M. *et al*. Silicon photonics: The next fabless semiconductor industry. *IEEE Solid-State Circuits Magazine* **5**, 48–58 (2013).
7. Thomson, D. *et al*. Roadmap on silicon photonics. *Journal of Optics* **18**, 073003 (2016).
8. Lim, A.-J. *et al*. Review of silicon photonics foundry efforts. *IEEE J. Sel. Top. Quantum Electron.* **20**, 405–416 (2014).
9. Orcutt, J. S. *et al*. Open foundry platform for high-performance electronic-photonic integration. *Opt. Express* **20**, 12222–12232 (2012).
10. Chrostowski, L. & Hochberg, M. *Silicon Photonics Design: From Devices to Systems* (Cambridge University Press, 2015).
11. Sun, J. *et al*. Large-scale silicon photonic circuits for optical phased arrays. *Selected Topics in Quantum Electronics, IEEE Journal of* **20**, 264–278 (2014).
12. Beausoleil, R. G. Large-scale integrated photonics for high-performance interconnects. *J. Emerg. Technol. Comput. Syst.* **7**, 6:1–6:54 (2011).
13. Le Beux, S. *et al*. Optical ring network-on-chip (ORNoC): Architecture and design methodology. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2011*, 1–6 (2011).
14. Narayana, V. K., Sun, S., Badawy, A.-H. A., Sorger, V. J. & El-Ghazawi, T. MorphoNoC: Exploring the Design Space of a Configurable Hybrid NoC using Nanophotonics. *arXiv:1506.03264* (2017).
15. Capmany, J. *et al*. Microwave photonic signal processing. *Journal of Lightwave Technology* **31**, 571–586 (2013).
16. Farsaei, A. *et al*. A review of wireless-photonic systems: Design methodologies and topologies, constraints, challenges, and innovations in electronics and photonics. *Optics Communications* (2016).
17. Feng, N.-N. *et al*. Thermally-efficient reconfigurable narrowband RF-photonic filter. *Opt. Express* **18**, 24648–24653 (2010).
18. Zhuang, L., Roeloffzen, C. G. H., Hoekman, M., Boller, K.-J. & Lowery, A. J. Programmable photonic signal processor chip for radiofrequency applications. *Optica* **2**, 854–859 (2015).

19. Valley, G. C. Photonic analog-to-digital converters. *Opt. Express* **15**, 1955–1982 (2007).
20. Khan, M. H. *et al*. Ultrabroad-bandwidth arbitrary radiofrequency waveform generation with a silicon photonic chip-based spectral shaper. *Nature: Photonics* **4**, 117–122 (2010).
21. Chang, J., Meister, J. & Prucnal, P. R. Implementing a novel highly scalable adaptive photonic beamformer using "blind" guided accelerated random search. *Journal of Lightwave Technology* **32**, 3623–3629 (2014).
22. Ferreira de Lima, T., Tait, A. N., Nahmias, M. A., Shastri, B. J. & Prucnal, P. R. Scalable wideband principal component analysis via microwave photonics. *IEEE Photonics Journal* **8**, 1–9 (2016).
23. Merolla, P. A. *et al*. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science* **345**, 668–673 (2014).
24. Akopyan, F. *et al*. Truenorth: Design and tool flow of a 65 mW 1 million neuron programmable neurosynaptic chip. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **34**, 1537–1557 (2015).
25. Indiveri, G. & Liu, S. C. Memory and information processing in neuromorphic systems. *Proceedings of the IEEE* **103**, 1379–1397 (2015).
26. Hasler, J. & Marr, H. B. Finding a roadmap to achieve large neuromorphic hardware systems. *Front. Neurosci.* **7** (2013).
27. Wen, U.-P., Lan, K.-M. & Shih, H.-S. A review of Hopfield neural networks for solving mathematical programming problems. *European Journal of Operational Research* **198**, 675–687 (2009).
28. Lee, T. & Theunissen, F. A single microphone noise reduction algorithm based on the detection and reconstruction of spectro-temporal features. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* **471** (2015).
29. Eliasmith, C. & Anderson, C. H. *Neural engineering: Computation, representation, and dynamics in neurobiological systems* (MIT Press, 2004).
30. Donnarumma, F., Prevete, R., de Giorgio, A., Montone, G. & Pezzulo, G. Learning programs is better than learning dynamics: A programmable neural network hierarchical architecture in a multi-task scenario. *Adaptive Behavior* **24**, 27–51 (2016).
31. Diamond, A., Nowotny, T. & Schmuker, M. Comparing neuromorphic solutions in action: implementing a bio-inspired solution to a benchmark classification task on three parallel-computing platforms. *Frontiers in Neuroscience* **9** (2016).
32. Tait, A. N., Nahmias, M. A., Shastri, B. J. & Prucnal, P. R. Broadcast and weight: An integrated network for scalable photonic spike processing. *Journal of Lightwave Technology* **32**, 4029–4041 (2014).
33. Brunner, D. & Fischer, I. Reconfigurable semiconductor laser networks based on diffractive coupling. *Optics letters* **40**, 3854–3857 (2015).
34. Tait, A. N. *et al*. Microring weight banks. *IEEE Journal of Selected Topics in Quantum Electronics* **22** (2016).
35. Tait, A. N., Ferreira de Lima, T., Nahmias, M. A., Shastri, B. J. & Prucnal, P. R. Multi-channel control for microring weight banks. *Opt. Express* **24**, 8895–8906 (2016).
36. Yamada, M. A theoretical analysis of self-sustained pulsation phenomena in narrow-stripe semiconductor lasers. *IEEE Journal of Quantum Electronics* **29**, 1330–1336 (1993).
37. Romeira, B. *et al*. Broadband chaotic signals and breather oscillations in an optoelectronic oscillator incorporating a microwave photonic filter. *Lightwave Technology, Journal of* **32**, 3933–3942 (2014).
38. Beer, R. D. On the dynamics of small continuous-time recurrent neural networks. *Adaptive Behavior* **3**, 469–509 (1995).
39. Zhou, E. *et al*. Silicon photonic weight bank control of integrated analog network dynamics. In *Optical Interconnects Conference, 2016 IEEE*, TuP9 (IEEE, 2016).
40. Stewart, T. C. & Eliasmith, C. Large-scale synthesis of functional spiking neural circuits. *Proceedings of the IEEE* **102**, 881–898 (2014).
41. Khanna, A. IMEC silicon photonics platform. In *European Conference on Optical Communication* (2015).
42. Roska, T. *et al*. Simulating nonlinear waves and partial differential equations via cnn. i. basic techniques. *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on* **42**, 807–815 (1995).
43. Ratier, N. Analog computing of partial differential equations. In *Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), 2012 6th International Conference on*, 275–282 (2012).
44. Vogelstein, R. J., Tenore, F. V. G., Guevremont, L., Etienne-Cummings, R. & Mushahwar, V. K. A silicon central pattern generator controls locomotion *in vivo*. *IEEE Transactions on Biomedical Circuits and Systems* **2**, 212–222 (2008).
45. Arena, P., Fortuna, L., Frasca, M. & Patane, L. A cnn-based chip for robot locomotion control. *IEEE Transactions on Circuits and Systems I: Regular Papers* **52**, 1862–1871 (2005).
46. Barron-Zambrano, J. H. & Torres-Huitzil, C. {FPGA} implementation of a configurable neuromorphic cpg-based locomotion controller. *Neural Networks* **45**, 50–61 Neuromorphic Engineering: From Neural Systems to Brain-Like Engineered Systems (2013).
47. Friedmann, S., Fr'emaux, N., Schemmel, J., Gerstner, W. & Meier, K. Reward-based learning under hardware constraints - using a RISC processor embedded in a neuromorphic substrate. *Front. Neurosci.* **7** (2013).
48. Benjamin, B. *et al*. Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations. *Proceedings of the IEEE* **102**, 699–716 (2014).
49. Pfeil, T. *et al*. Is a 4-bit synaptic weight resolution enough? – constraints on enabling spike-timing dependent plasticity in neuromorphic hardware. *Frontiers in Neuroscience* **6**, 90 (2012).
50. Binas, J., Neil, D., Indiveri, G., Liu, S.-C. & Pfeiffer, M. Precise deep neural network computation on imprecise low-power analog hardware. *arXiv preprint arXiv:1606.07786* (2016).
51. Shen, Y. *et al*. Deep learning with coherent nanophotonic circuits. *arXiv:1610.02365* (2016).
52. Shainline, J. M., Buckley, S. M., Mirin, R. P. & Sae Woo, N. Superconducting optoelectronic circuits for neuromorphic computing. *arXiv preprint arXiv:1610.00053* (2016).
53. Nahmias, M. A., Shastri, B. J., Tait, A. N. & Prucnal, P. R. A leaky integrate-and-fire laser neuron for ultrafast cognitive computing. *IEEE J. Sel. Top. Quantum Electron.* **19**, 1–12 (2013).
54. Prucnal, P. R., Shastri, B. J., Ferreira de Lima, T., Nahmias, M. A. & Tait, A. N. Recent progress in semiconductor excitable lasers for photonic spike processing. *Adv. Opt. Photon.* **8**, 228–299 (2016).
55. Selmi, F. *et al*. Relative refractory period in an excitable semiconductor laser. *Phys. Rev. Lett.* **112**, 183902 (2014).
56. Romeira, B., Av'o, R., Figueiredo, J. M. L., Barland, S. & Javaloyes, J. Regenerative memory in time-delayed neuromorphic photonic resonators. *Scientific Reports* **6**, 19510 EP – (2016).
57. Nahmias, M. A. *et al*. An integrated analog O/E/O link for multi-channel laser neurons. *Applied Physics Letters* **108** (2016).
58. Vaerenbergh, T. V. *et al*. Cascadable excitability in microrings. *Opt. Express* **20**, 20292–20308 (2012).
59. Shastri, B. J. *et al*. Spike processing with a graphene excitable laser. *Sci. Rep.* **5**, 19126 (2015).
60. Zhang, H. *et al*. Z-scan measurement of the nonlinear refractive index of graphene. *Opt. Lett.* **37**, 1856–1858 (2012).
61. Hill, M., Frietman, E. E. E., de Waardt, H., Khoe, G.-D. & Dorren, H. All fiber-optic neural network using coupled soa based ring lasers. *IEEE Trans. Neural Networks* **13**, 1504–1513 (2002).
62. Brunner, D., Soriano, M. C., Mirasso, C. R. & Fischer, I. Parallel photonic information processing at gigabyte per second data rates using transient states. *Nat Commun* **4**, 1364 (2013).
63. Vandoorne, K. *et al*. Experimental demonstration of reservoir computing on a silicon photonics chip. *Nat Commun* **5** (2014).
64. Soriano, M. C., Brunner, D., Escalona-Mor'an, M., Mirasso, C. R. & Fischer, I. Minimal approach to neuro-inspired information processing. *Frontiers in Computational Neuroscience* **9**, 68 (2015).

65. Duport, F., Smerieri, A., Akrout, A., Haelterman, M. & Massar, S. Fully analogue photonic reservoir computer. *Scientific Reports* **6**, 22381 EP – (2016).
66. Vandoorne, K. *et al*. Toward optical signal processing using photonic reservoir computing. *Opt. Express* **16**, 11182–11192 (2008).
67. Mesaritakis, C., Papataxiarhis, V. & Syvridis, D. Micro ring resonators as building blocks for an all-optical high-speed reservoir-computing bit-pattern-recognition system. *J. Opt. Soc. Am. B* **30**, 3048–3055 (2013).
68. Hopfield, J. J. & Tank, D. W. "Neural" computation of decisions in optimization problems. *Biological Cybernetics* **52**, 141–152 (1985).
69. Tumuluru, V. K., Wang, P. & Niyato, D. A neural network based spectrum prediction scheme for cognitive radio. In *Communications (ICC), 2010 IEEE International Conference on*, 1–5 (2010).
70. Mitra, U. & Poor, H. V. Neural network techniques for adaptive multiuser demodulation. *IEEE Journal on Selected Areas in Communications* **12**, 1460–1470 (1994).
71. Du, K.-L., Lai, A., Cheng, K. & Swamy, M. Neural methods for antenna array signal processing: a review. *Signal Processing* **82**, 547–561 (2002).
72. Tait, A. *et al*. Silicon microring weight banks for multivariate RF photonics. In *CLEO: 2017* (IEEE, 2017 (accepted)).
73. Bojko, R. J. *et al*. Electron beam lithography writing strategies for low loss, high confinement silicon optical waveguides. *J. Vac. Sci. Technol., B* **29** (2011).
74. Wang, Y. *et al*. Focusing sub-wavelength grating couplers with low back reflections for rapid prototyping of silicon photonic circuits. *Opt. Express* **22**, 20652–20662 (2014).
75. Tait, A., F de Lima, T., Nahmias, M., Shastri, B. & Prucnal, P. Continuous calibration of microring weights for analog optical networks. *Photonics Technol. Lett.* **28**, 887–890 (2016).
76. Zhang, Y. *et al*. A compact and low loss Y-junction for submicron silicon waveguide. *Opt. Express* **21**, 1310–1316 (2013).
77. Bekolay, T. *et al*. Nengo: a Python tool for building large-scale functional brain models. *Frontiers in Neuroinformatics* **7**, 48 (2013).
78. Friedl, K. E., Voelker, A. R., Peer, A. & Eliasmith, C. Human-inspired neurorobotic system for classifying surface textures by touch. *IEEE Robotics and Automation Letters* **1**, 516–523 (2016).
79. Mundy, A., Knight, J., Stewart, T. & Furber, S. An efficient SpiNNaker implementation of the neural engineering framework. In *Neural Networks (IJCNN), 2015 International Joint Conference on*, 1–8 (2015).
80. Chrostowski, L. *et al*. Impact of fabrication non-uniformity on chip-scale silicon photonic integrated circuits. In *Optical Fiber Communication Conference*, Th2A.37 (Optical Society of America, 2014).
81. Jayatilleka, H. *et al*. Wavelength tuning and stabilization of microring-based filters using silicon in-resonator photoconductive heaters. *Opt. Express* **23**, 25084–25097 (2015).

Acknowledgements

This work is supported by National Science Foundation (NSF) Enhancing Access to the Radio Spectrum (EARS) program (Award 1642991). Fabrication support was provided via the Natural Sciences and Engineering Research Council of Canada (NSERC) Silicon Electronic-Photonic Integrated Circuits (SiEPIC) Program. Devices were fabricated by Richard Bojko at the University of Washington Washington Nanofabrication Facility, part of the NSF National Nanotechnology Infrastructure Network (NNIN).

Author Contributions

A.N.T. conceived of the methods and experiments. A.N.T., E.Z., and A.X.W. conducted the experiments and analyzed the results. T.FdL developed methods and simulations of the photonic CTRNN emulator. All authors reviewed the manuscript.

Additional Information

Supplementary information accompanies this paper at doi:10.1038/s41598-017-07754-z

Competing Interests: The authors declare that they have no competing interests.

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

Neuromorphic photonic networks using silicon photonic weight banks

Supplementary information

Alexander N. Tait, Thomas Ferreira de Lima, Ellen Zhou, Allie X. Wu, Mitchell A. Nahmias,
Bhavin J. Shastri, and Paul R. Prucnal

1 Derivation of Theoretical Bifurcations

The CTRNN model of interest is described by equations (1-2). Mathematical analysis of dynamical systems begins by examining fixed states (where $\dot{\vec{s}} = 0$) and the effects of parameters on their behavior. Bifurcations occur when the number or stability of fixed-points change as a result of parameter variation. Here, we derive simple bifurcations of small networks[38]. Predictions of amplitude and frequency are then compared to experiment in Fig. 2 and Fig. 3. For the sake of analysis, the neuron transfer function, whether sigmoidal (in the case of a perceptron) or sinusoidal (in the case of a MZM neuron), is approximated to the first nonlinear term: $\sigma(s) \approx \alpha s - \kappa s^3$, where α and κ are positive coefficients.

Cusp bifurcations describe transitions from monostability to bistability. A cusp can be observed in the simplest case of equations (1-2), which is a single node with self-feedback weight, W_F.

$$0 = W_F \sigma(s^*) - \frac{s^*}{\tau} + w_{in} u^* = \kappa W_F s^{*3} - \left(\alpha W_F - \tau^{-1}\right) s^* - w_{in} u^* \tag{S1.1}$$

where s^* and u^* are the steady-state scalar values of the neuron state and input, respectively.

When the input is zero, the steady-state solutions have the form:

$$s^*_{(1)} = 0; \quad s^*_{(2,3)} = \pm\sqrt{\frac{\alpha}{\kappa}\frac{W_F - W_B}{W_F}} \tag{S1.2}$$

where $W_B = (\alpha\tau)^{-1}$ is the bifurcation weight, and subscripts index the three solution branches. Below W_B, solution branches (2) and (3) are imaginary and therefore do not physically exist. This expression, plotted on the W_F-y axis of Fig. 2(b), exhibits the standard form of a pitchfork bifurcation, in which two stable solution branches arise out of one stable branch.

Returning to the general expression, the inputs, u^*, that yield steady-state solutions, s^*, take the form

$$u^* = \frac{\kappa W_F}{w_{in}}\left(s^{*3} - \frac{\alpha}{\kappa}\frac{W_F - W_B}{W_F} s^*\right) \tag{S1.3}$$

The resulting, familiar S-shaped bistable curve is plotted on the u-y axis of Fig. 2(b). Three roots of s^* exist when feedback weight is fixed above the pitchfork bifurcation value. The edges of this bistable regime are referred to as saddle-node points because the unstable middle saddle and one of the stable nodes annihilate one another. The saddle-node points (s^*_{SN}, u^*_{SN}) are found where the derivative of u^* in equation (S1.3) is zero with respect to s^*.

$$s^*_{SN} = \pm\sqrt{\frac{\alpha}{3\kappa}\frac{W_F - W_B}{W_F}}; \quad u^*_{SN} = \pm\frac{2}{3}\frac{\alpha^{3/2}}{w_{in}\sqrt{3\kappa W_F}}\left(W_F - W_B\right)^{3/2} \tag{S1.4}$$

These parametric equations define a cusp, which is projected onto the W_F-u axis of Fig. 2(b). The cusp bifurcation is more informative than either the pitchfork or saddle-node bifurcations because it is described only in reference to two parameters, while the other bifurcations can occur in systems of one parameter.

Hopf bifurcations are characterized by a transition from stable to oscillating dynamics. In experiments and this analysis, we fix the off-diagonal weights asymmetrically such that $-w_{12} = w_{21} = 1$ and parameterize the diagonals such that $w_{11} = w_{22} = W_F$. Under this formulation, there is always one and only one steady-state at $\vec{s} = 0$. To examine its stability, we linearize the system around this point to yield the Jacobian matrix whose eigenvalues indicate fixed-point stability.

$$\mathbf{J} = \frac{d}{ds}\left(\frac{ds}{dt}\right) = \alpha\begin{bmatrix} W_F - (\alpha\tau)^{-1} & -1 \\ 1 & W_F - (\alpha\tau)^{-1} \end{bmatrix}; \quad \text{eigenvalues: } \lambda = W_F - (\alpha\tau)^{-1} \pm i \tag{S1.5}$$

The imaginary part of the eigenvalue pair is indicative of oscillating behavior. The real part of the eigenvalue switches sign at the bifurcation weight $W_B = (\alpha\tau)^{-1}$. In this case, when the only fixed-point solution becomes unstable, a stable limit cycle arises instead of new stable states. Near threshold, we can assume a circular form of the limit cycle in order to model its expected amplitude, A, and frequency, ω.

$$s_1(t) = A\sin(\omega t); \quad s_2(t) = A\cos(\omega t) \tag{S1.6}$$

At points where ωt is a multiple of 2π, the time-derivative of s_2 is zero. Examining the $\dot{s}_2$ equation from equations (1-2),

$$\left.\frac{ds_2}{dt}\right|_{\omega t = 2\pi m} = 0 = \sigma(0) + W_F\sigma(A) - \tau^{-1}A \tag{S1.7}$$

$$A = \sqrt{\frac{\alpha}{\kappa}\frac{W_F - W_B}{W_F}} \tag{S1.8}$$

where m is an integer. The amplitude follows a form similar to that of the pitchfork bifurcation in equation (S1.2) and is plotted as a red curve on the s_1-W_F axis of Fig. 3(d). The equation for $\dot{s_1}$ at this same point can be used to find the angular frequency.

$$\left.\frac{ds_1}{dt}\right|_{\omega t=2\pi m} = -A\omega = W_F\sigma(0) - \sigma(A) \tag{S1.9}$$

$$\omega = \frac{\tau^{-1}}{W_F} \tag{S1.10}$$

The expected limit cycle frequency is therefore finite at the Hopf point and inversely proportional above, as shown in Fig. 3(e).

2 NEF Compilation Procedure

2.1 Solving ODEs with Photonic Modulator Neurons

The complete jupyter notebook used to generate plots in this section and in Fig. 4 is available in: https://github.com/lightwave-lab/Neuromorphic_Silicon_Photonics

Modifications to the nengo project

Nengo is based exclusively on monotonic, non-negative output neuron models. However, its encoding-decoding algorithms should work with other kinds of neuron models. Here, we use the following `FourierSinusoid` class of neurons included in our fork of the nengo project.

The Lorenz chaotic attractor

In this simulation, we chose to construct a neural network using the neurons defined above to solve a classical chaotic dynamical system named "Lorenz attractor".

The equations are:

$$\dot{x}_0 = \nu(x_1 - x_0) \qquad \dot{x}_1 = x_0(\rho - x_2) - x_1 \qquad \dot{x}_2 = x_0 x_1 - \beta x_2$$

Since x_2 is centered around approximately ρ, and since NEF ensembles are usually optimized to represent values within a certain radius of the origin, we substitute $x_2' = x_2 - \rho$, giving these equations:

$$\dot{x}_0 = \nu(x_1 - x_0) \qquad \dot{x}_1 = -x_0 x_2' - x_1 \qquad \dot{x}_2' = x_0 x_1 - \beta(x_2' + \rho)$$

Refer to the standard example of the Lorenz attractor solver with 2000 neurons in a nengo example. *Note that the last equation for x_2' is typically shown with an error in that example and in other articles from Prof. Eliasmith's group.

2.2 Encoding strategy

From here onwards, we will refer the Lorenz system in its reduced form as as $\vec{x} = f(\vec{x})$, with:

$$\vec{x} = [x_0, x_1, x_2']^T \quad \text{and} \quad f(\vec{x}) = \begin{bmatrix} \nu(x_1 - x_0) \\ -x_0 x_2' - x_1 \\ x_0 x_1 - \beta(x_2' + \rho) \end{bmatrix}$$

In the following sections, we briefly explain the details on how nengo can be used to inform us on how to configure a photonic neural network to emulate an accelerated ODE, having the Lorenz attractor as an example.

Neuron model

Using nengo, we instantiate a population of N neurons that are all-to-all interconnected. These neurons are responsible of *representing* the vector $\vec{x}$ at any time t. We consider the state of each neuron as $\vec{s} = [s_i]$ for neuron i. The ODE that models each neuron, in this case, is:

$$\tau s_i + s_i = u_i$$

where u_i represents the post-synaptic input of the neuron and $y_i = \sigma(s_i)$ its output.

Nengo encoding strategy

In order to *encode* a vector $\vec{x}$ in the population N, nengo performs the following linear transformation (it has to be linear for the method to work):

$$s_i = g_i \vec{e}_i \cdot \vec{x} + b_i$$

where g_i is a gain term, $\vec{e}_i$ is an encoder vector, and b_i is a bias term. This is called the *encoding strategy*.

Nonlinear operations are effectively performed by linear combinations of the neural nonlinearities $\sigma(s_i)$. Therefore, it is the encoder's mission to generate as much entropy about the variables $\vec{x}$ as possible. This can be done by generating a diverse set of $(g, \vec{e}, b)$ parameters. Below, we do this by using $\vec{e}_i = [1, \pm 1, \pm 1]$, mixing all components of $\vec{x}$ together. Note: this can be optimized even further by noticing that the ODE does not contain $x_0 x_2$ terms.

Because we know that σ is a sinusoid, we create a set of (g, b) values to span a Fourier-like basis of functions across the domain $\vec{e}_i \cdot \vec{x} \in [-1, 1]$. (See tuning curves).

```python
# Intercept, in this case, corresponds to where the tuning curve intercepts
# zero. Range of [-.5, .5] corresponds to [-pi, pi]
ints = [0, 1/4]
# This number represents how many periods do we want between -1 and 1
# (see tuning curves below)
rats = s_pi * np.arange(1, 4)/2
# Encoder multipliers
enst = [-1,1]

num_intercepts = len(ints)
num_max_rates = len(rats)
num_encoders = len(enst) ** 2

j = 0
encoders = np.zeros(shape=(num_neurons, 3))
intercepts = np.zeros(num_neurons)
max_rates = np.zeros_like(intercepts)
for ir in range(num_max_rates):
    for ii in range(num_intercepts):
        for ie0 in range(len(enst)):
            if ie0 is 0:
                continue
            for ie1 in range(len(enst)):
                for ie2 in range(len(enst)):
                    vertex = np.array([enst[ie0], enst[ie1], enst[ie2]])
                    if not np.all(vertex == 0):
                        encoders[j,:] = vertex
                        intercepts[j] = ints[ii]
                        max_rates[j] = rats[ir]
                        j += 1
```

Neuron	Encoder	Intercept	Max_rates
1	[1. -1. -1.]	0.0	0.05
2	[1. -1. 1.]	0.0	0.05
3	[1. 1. -1.]	0.0	0.05
4	[1. 1. 1.]	0.0	0.05
5	[1. -1. -1.]	0.25	0.05
6	[1. -1. 1.]	0.25	0.05
7	[1. 1. -1.]	0.25	0.05
8	[1. 1. 1.]	0.25	0.05
9	[1. -1. -1.]	0.0	0.1
10	[1. -1. 1.]	0.0	0.1
11	[1. 1. -1.]	0.0	0.1
12	[1. 1. 1.]	0.0	0.1
13	[1. -1. -1.]	0.25	0.1
14	[1. -1. 1.]	0.25	0.1
15	[1. 1. -1.]	0.25	0.1
16	[1. 1. 1.]	0.25	0.1
17	[1. -1. -1.]	0.0	0.15
18	[1. -1. 1.]	0.0	0.15
19	[1. 1. -1.]	0.0	0.15
20	[1. 1. 1.]	0.0	0.15
21	[1. -1. -1.]	0.25	0.15
22	[1. -1. 1.]	0.25	0.15
23	[1. 1. -1.]	0.25	0.15
24	[1. 1. 1.]	0.25	0.15

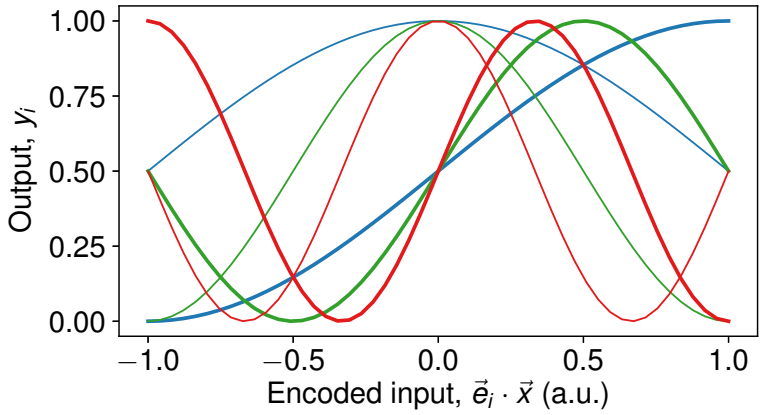

Figure S2.1. Selected tuning curves corresponding to sinusoidal Mach-Zehnder transfer functions. Output is transmission and input is voltage normalized to V_π. The encoded input is derived from the dot product of the encoding weight vector and the input signal vector.

Tuning curves in Fourier basis

Here, assuming that the neuron states are $s_i = g_i \vec{e}_i \cdot \vec{x} + b_i$, we plot the functions $\sigma(s_i)$ for neurons with different g_i, b_i values according to the previous table.

Timing

In order to account for the true-time delay between the output of a neuron and the feedback latency of the photonic waveguides, we instantiate a delay node in nengo, which essentially offsetstime signals in time.

```
# Round-trip feedback delay in ns
delayTime = .048
# gamma is a characteristic time scale in real time units
# The coefficient gamma/delayTime determines the stability
# In paper, coefficient was 65 (spurious), 104 (inaccurate), 260 (looks good)
gamma = 260 * delayTime

# We'll make a simple object to implement the delayed feedback
class Delay(object):
    def __init__(self, dimensions, timesteps=50):
        timesteps = max(timesteps,1)
        self.history = np.zeros((timesteps, dimensions))
    def step(self, t, x):
        self.history = np.roll(self.history, -1, axis=0)
        self.history[-1] = x
        return self.history[0]
delay = Delay(3, timesteps=int(delayTime / dt))
```

2.3 Nengo Implementation

After having the encoding strategy laid out, we are then ready to extract any function from the population. Here, we set the feedback function to be $\tau f(\vec{x}) + \vec{x}$, for reasons that are explained in the following section.

```
# the ODE to emulate
# The default values for sigma, beta and rho originally used by Lorenz.
# Cf. https://en.wikipedia.org/wiki/Lorenz_system#Analysis
nu = 10
beta = 8.0/3
```

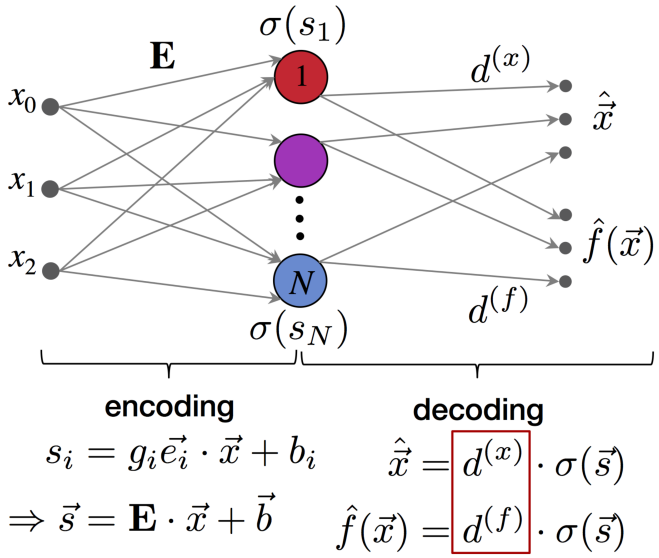

Figure S2.2. Encoding and decoding a variable with a population of neurons in the Neural Engineering Framework.

```
rho = 28
def feedback(x):
    dx0 = (-nu * x[0] + nu * x[1]) / gamma
    dx1 = (-x[0] * x[2] - x[1]) / gamma
    dx2 = (x[0] * x[1] - beta * (x[2] + rho)) / gamma

    return [dx0 * tau + x[0],
            dx1 * tau + x[1],
            dx2 * tau + x[2]]
```

In the following code snippet, we show how we can instantiate an ensemble of neurons and set the feedback connections to emulate the Lorenz attractor.

```
# The main ensemble
state = nengo.Ensemble(num_neurons, dimensions=3,
    intercepts=intercepts,
    neuron_type=nengo.neurons.FourierSinusoid(max_overall_rate=max_transmission,
                                              s_pi=s_pi),
    max_rates=max_rates,
    encoders=encoders, radius=60.)

# This special node calls a function every timestep,
# in this case a class method of delay
delay_node = nengo.Node(delay.step, size_in=3, size_out=3)

# Connections from state to delay and back
cdel = nengo.Connection(state, delay_node,
                        function=feedback, synapse=tau)
conn = nengo.Connection(delay_node, state)
```

2.4 Decoding strategy: calculating weight matrix

As mentioned, nengo decodes a function $h(\vec{x})$ from the population of neurons by a linear decoding strategy, i.e. a matrix $d^{(h)}$ resulting in an estimator $\hat{h}(\vec{x})$:

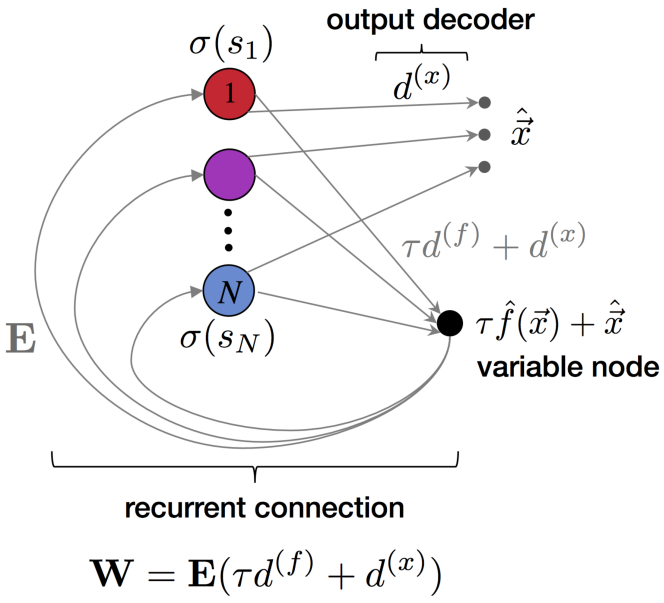

Figure S2.3. Recurrent connection for dynamical system emulation in the Neural Engineering Framework.

$$\hat{h}(\vec{x}) = d^{(h)}\vec{y}$$

where $y_i = \sigma(s_i) = \sigma(g_i\vec{e}_i \cdot \vec{x} + b_i)$.

This matrix $d^{(h)}$ is uniquely dependent on the encoder strategy, the neuron's transfer function σ and the function h. As a result, it can be pre-computed before any real-time simulation. Namely, it attempts to minimize the following objective function:

$$J = \int \left\| d^{(h)}\vec{y} - h(\vec{x}) \right\| \mathrm{d}\vec{x}$$

where the integral is over the desired range of values of $\vec{x}$.

The minimum can be calculated via the Moore-Penrose pseudoinverse method (Stewart et al. Front Neuroinform. 3 (2009)):

$$\Gamma_{ij} = \int y_i y_j \mathrm{d}\vec{x}$$

$$\Upsilon_i = \int y_i h(\vec{x}) \mathrm{d}\vec{x}$$

$$d^{(h)} = \Gamma^{-1} \cdot \Upsilon$$

Weight matrix

If we add an all-to-all recurrent connection to the neural population, their collective dynamics is described by the following ODE system:

$$\tau\dot{\vec{s}} + \vec{s} = \overline{\overline{W}}\sigma(\vec{s}) + \vec{I}$$

where $\overline{\overline{W}}$ is the weight matrix and $\vec{I}$ a bias vector.

Nengo sets $\overline{\overline{W}} = \overline{\overline{E}}(d^{(x)} + \tau d^{(f)})$ and $\vec{I} = \vec{b}$, where $\overline{\overline{E}}_{ij} = (\vec{e}_i)_j$. When applied to the ODE above, it is easy to see that one can recover the Lorenz system:

$$\overline{\overline{E}}(\tau\dot{\vec{x}} + \vec{x}) = \overline{\overline{E}}(\hat{\vec{x}} + \tau\hat{f}(\vec{x}))$$

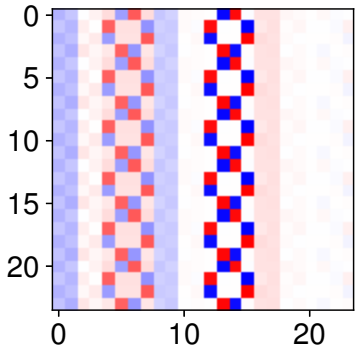

Figure S2.4. Recurrent weight matrix returned by Nengo compiler procedure. Red (blue) correspond to weights that are more positive (negative). X and Y axes correspond to neuron index, from 1 to 24.

$$\implies \dot{\vec{x}} = f(\vec{x}) + \varepsilon(\vec{x})$$

where $\varepsilon(\vec{x}) = (1/\tau)(\hat{\vec{x}} - \vec{x}) + \hat{f}(\vec{x}) - f(\vec{x})$.

Below, we show the computed weight matrix $\overline{\overline{W}}$ for this system.