A Physical Machine Based on a Super-Turing Computational Model

A. Steven Younger[1], Emmett Redd[1] Hava Siegelmann[2], and Conrad Bell[3]

[1] Missouri State University, Springfield, MO 65897, USA,
{steveyounger, emmettredd}missouristate.edu,
[2] University of Massachusetts-Amherst, Amherst, MA 01003, USA,
hava@cs.umass.edu,
[3] Nova Southeastern University, Fort Lauderdale, FL 33314 USA,
cb1399@nova.edu

Abstract. We present evidence that the Turing machine is too restrictive a model to sufficiently describe the computation of our analog computer and, therefore, a more comprehensive model is needed. We report on the construction of a prototype, the Optical Analog Recurrent Neural Network (OpticARNN), and experimental results showing that it performs computations which are beyond those of computers based on the Turing machine. We conclude that the behavior of OpticARNN is better described by the super-Turing computational model proposed by Siegelmann. To the best of our knowledge, this is the first application of analog recurrent neural networks realized in a physical computer based on this model.

Keywords: analog computing, recurrent neural networks, super-Turing computation, optical computing

1 Introduction

The Turing machine (TM) is one of the most powerful models in the history of science. It is the mathematical basis of digital computers and a major enabling factor for ushering in the Information Age. Indeed, the TM's success at modeling new types of computation has led to the 'widespread belief' amongst some that "models of computation more expressive than TMs are impossible" [1].

We developed a physical realization of a machine based on a computational model that subsumes the Turing model. The super-Turing theory of computation [2–4, 6] demonstrated that an analog recurrent neural network (ARNN) can represent a hierarchy of complexity classes, including the Turing machine. We show that our physical machine is capable of performing some kinds of computation that a physical realization of the Turing Machine (digital computer) cannot perform. In the following sections, the super-Turing theory of computation is presented, followed by the design and development of OpticARNN, our Optical Analog Recurrent Neural Network computer. We describe the tests performed to verify its super-Turing capability and the results obtained through experiments conducted on OpticARNN.

1.1 Theory Background: Super-Turing Analog Recurrent Neural Networks

The computational power of a machine can be classified according to the complexity class of the problems that it can solve. These complexity classes are quite abstract and have precise mathematical relationships to each other. However, for the purposes of this paper, the names of each complexity class serve as identifiers indicating relative computational strength.

Siegelmann developed a super-Turing computational model based on analog recurrent neural networks [2–4, 6]. Turing Machines can solve problems of complexity class ***P***, as can an ARNN with rational weights and discrete-time signals. ARNNs with deterministic, real-valued signals can solve problems of class ***P/poly***, which is a strict superset of class ***P***. In addition, ARNNs with rational weights and signals that have stochastic noise can solve problems of complexity class ***BPP/log**** , which is also super-Turing but of lower computational complexity than ***P/poly***. In fact, there is an infinite hierarchy of classes that can be computed efficiently between the TM and a super-Turing computer, including ***BPP/log**** [2].

In our optical ARNN (OpticARNN), stochasticity is generated by the fundamental statistical and quantum physics of its laser and analog components. Indeed, stochasticity cannot be removed from the system. The underlying physics allows that this stochasticity can be characterized by real-numbered values. The probability of the 'coin' being 'heads' can be $1/\sqrt{2}$, for example.

A long sequence of measurements allows indirect access to the real-valued quantity, facilitating its approximation to high precision [2] (p.122). As a consequence, any physically realizable super-Turing machine must be non-deterministic. Turing proposed this idea when discussing how to make a machine behave like a brain by making "its behavior depend on something like a roulette wheel or a supply of radium" [7].

The stochastic noise in the intensity of an optical signal corresponds to fluctuations of the least significant bit in a series of measurements of a physical value. One can view this "jitter" as being determined by a coin flip. Heads means that the noise bit of a particular measurement is 1; tails, the bit is 0. To be provably super-Turing, the probability of a given measurement being heads (or tails) must be a real number. This is significant because a machine that computes with deterministic real-valued signals might not be physically realizable, however an ARNN with stochastic signals can be constructed.

This indirect access to the real-numbered value is similar to Stochastic Resonance [8] which is used to increase measurement precision in areas from Experimental Physics to Neurobiology. Note that this true noise stochasticity is not the same as adding deterministic noise, such as from a Pseudo-Random Number Generator. This is because deterministic noise + digitized signal is in a discrete state space, not the continuous space of the ARNN.

The above suggests the following Conjecture: *All physically realizable Super-Turing machines are stochastic.*

1.2 Important Note

We are trying to extend Computational Modeling from discussions of abstract models toward use in the design and development of physically realized machines. Our hypothesis is that the abstract notions of computational power from theory can be reflected in physical realizations of these models. The ARNN super-Turing theory guided the design of our OpticARNN and we show it has more powerful computational abilities than a physically realized machine based on the Turing Machine model. We necessarily must compare physical implementations of Turing Machines to physical implementations of super-Turing machines. The abstract Turing Machine is unbounded in memory size, while the common physical realization of Turing Machines–digital computers–have finite size.

Science has many examples of a (sometimes quite idealized) theory resulting in prediction of actual physical phenomena. Quantum and relativistic effects cause Newtons Laws to not be an exact model of reality. However, nobody would dispute Newtonian theory's usefulness. In the area of neural networks, the Universal Approximation Theorem only holds rigorously for real-numbered weights and signals, but it is perhaps the most powerful theorem in the field[9]. Even theoretical results based (at least initially) on discrete vs. continuum arguments (as does ARNN theory) have resulted in predictions of phenomena that have been experimentally proven. The Casimir force is such an example [10].

One would probably not use a super-Turing-based computer to compute check-book balances or the trajectory of an ICBM. Digital computers are well-suited for these tasks, and many others. We do not know if classical computational modeling problems, such as the famous Halting Problem, can eventually be solved by any of these machines. However, super-Turing-based systems are expected to perform well on tasks the larger problem space is important, such as pattern recognition, modeling biological neurons, artificial neural network robustness and processing real-world data [4]. This expectation was noted by Turing himself in 1950: "The nervous system is certainly not a discrete-state machine. A small error in the information about the size of a nervous impulse impinging on a neuron, may make a large difference to the size of the outgoing impulse. It may be argued that, this being so, one cannot expect to be able to mimic the behavior of the nervous system with a discrete state system." [7].

2 OpticARNN Description of Operation

We designed and constructed an Optical Analog Recurrent Neural Network guided by the super-Turing model. We present only a summary herein (refer to [11] for more complete engineering details).

While it would have been possible to construct a purely analog system, doing so was impractical within the scope of our work. In addition, it is only necessary for the core computation to occur using analog data in order to demonstrate super-Turing capability. So, as a matter of convenience, we generated OpticARNN inputs and processed its outputs digitally.

The OpticARNN is based on the Optical Stanford Matrix-Vector Multiplier [12]. The signal flow starts in the lower center of Fig. 1, where neuron activation values sent from the Host Computer are converted to modulated electrical currents by a Field Programmable Array (FPGA). The FPGA generates laser modulation and driver signals for up to 60 source neurons, where each neuron is represented by one laser beam.

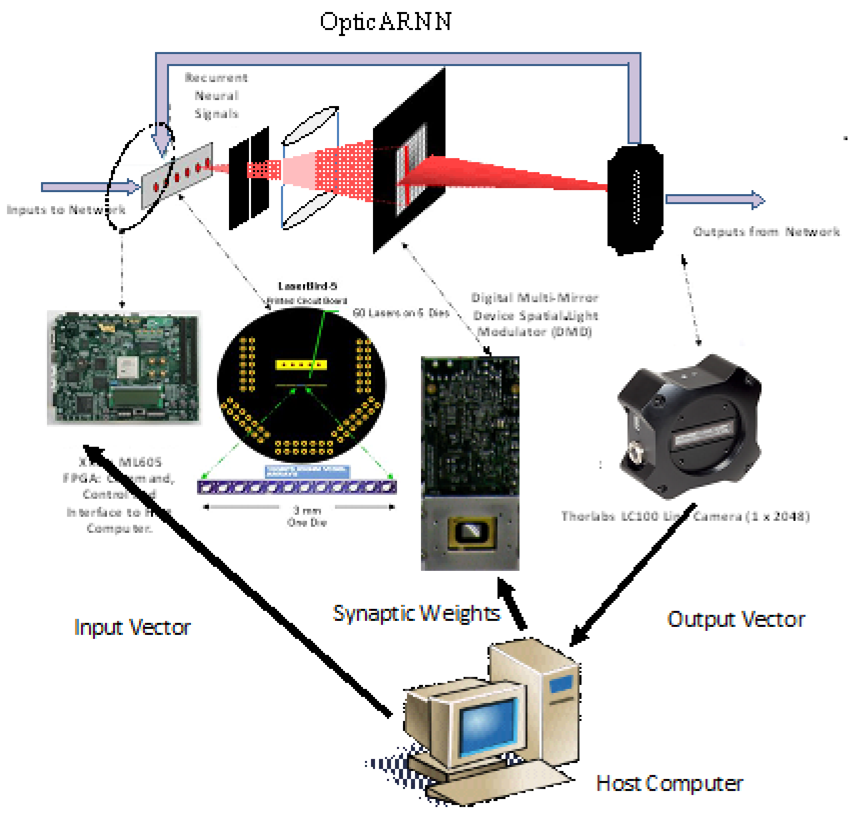

Fig. 1. Optical Analog Recurrent Neural Network

The 60 source Vertical Cavity Surface Emitting Lasers, shown in the upper left (Fig. 1), are mounted on the circular circuit board in a horizontal, linear array. Optical components project the signals onto a Spatial Light Modulator, which performs the synaptic multiplications (center top) by intensity reductions, or attenuations, of multiple portions of each vertical laser signal.

The OpticARNN uses a Digital Micro-Mirror Device (DMD) for its spatial light modulator. The DMD works in reflective mode and consists of a 1024 x 738 array of tiny mirrors. Each mirror can be independently and rapidly set to either ON (attenuation, 0) or OFF (attenuation, 1). Each synapse has its own rectangular region of interest (ROI) on the DMD. The synaptic weight of the ROI is set by adjusting the portion of ON vs. OFF mirrors.

The optics of the system causes light to focus onto a vertical linear array of 2048 CCD photodetectors (top right). The individual intensity-modulated pulses are horizontally summed by the optics, and temporally integrated by the CCD photodetectors. Both of these are analog computational processes.

Finally, the signals are digitized and sent to the host computer, where the excitatory and inhibitory signals are combined and a squashing function applied.

2.1 Hardware-in-the-Loop Training

Software digital neurons and synapses perform their computations using the same, identical, idealized behavior based on their coding. Digital hardware devices achieve nearly-idealized behavior by imposing a digital result on their outputs. Occasionally, this does cause a rounding error, but the error rate is very small in today's digital devices. Analog neurons and synapses, however, cannot be created identically. They are individual components having non-idealized behavior. This creates a difficult problem in terms of exactly replicating OpticARNN results from one run to the next.

Our solution was to train the initial synaptic weights with hardware-in-the-loop, meaning that the analog hardware does all of the forward-propagation computations. Software on the host computer performs synaptic weight updates using information, such as the activations of all of the neurons, from the hardware. Simulated annealing was then applied to reduce the mean-squared error.

The advantage of using simulated annealing for hardware-in-the-loop training is that it treated the ARNN as a black box and made no assumptions about its internal structure. However, it may not have been optimal in this case. Frye, et al. [13] found that using backpropagation of errors was faster than simulated annealing for hardware-in-the-loop training. This is a potential future enhancement.

3 Testing for Computation Beyond the Turing Limit

We next translate the abstract theoretical analysis of the ARNN models into an operational experimental test of super-Turing capability. Theoretical definitions of terms such as 'computation', 'number-system', and 'stochastic' must be interpreted in physical terms.

Some definitions of 'computation' require that the result be exactly the same for each computation. This does not allow for non-deterministic or approximate calculations, as in the use of slide rules and similar devices. Generations of scientists and engineers would disagree with this interpretation, so we relax this too-strict definition. *Computation* is an information process that transforms inputs to outputs. This physical interpretation does not require deterministic exactness and is broadly inclusive. It encompasses the activity of biological neural networks, conventional analog computers, and Turing machines.

Definition: We will call a physically realized Turing Machine a digital computer. We exclude any extra functionality that a given actual computer may have that is beyond the TM model, such as a true random number generator or real-time clock. This definition also means that the machine is deterministic and operating with bounded memory.

Definition: A physically realized super-Turing machine must be capable of:
Criterion 1: Doing any computation that a digital computer can
Criterion 2: Doing at least one computation that a digital computer cannot

Criterion 1 was proven in [2], namely that an ARNN with rational weights and signals is equivalent to a TM. Based on this proof, we assume that a physically realized ARNN will meet this criterion, at least in principle. (Our current prototype may not be large enough to do this).

Criterion 2 bounds the problem i.e. it is only necessary for a super-Turing machine to solve at least one problem, but not all problems, beyond the Turing limit.

In order to test super-Turing computation, a suitable problem must be found. In this case, the answer came from the area of chaotic systems. The dynamics of chaos are both aperiodic and defined on a continuous phase space. As such, they cannot be mimicked by a Turing Machine [2](p.155).

We examined a well-known chaotic time series, the Logistic Map:

$$y_n = rx_n(1 - x_n) \qquad ; \qquad x_{n+1} = y_n \tag{1}$$

The r parameter determines the behavior of the time series. While the generated time series is stable for a range of r values, it becomes chaotic for values between 3.9 and 4.0. We used the value of r = 3.99 throughout this study.

When generating a time series, a digital computer attempting to simulate chaos will settle into a repeating pattern i.e. every digitally generated time series eventually becomes periodic. We call this phenomenon the ‘Digital Artifact’. It was explored in detail by Blank [14] where he described it as ”a ‘localization’ process” where ”trajectories which should normally remain dense remain confined to a small number of points”. The Digital Artifact is characterized by the repeat period of a digitally-generated chaotic time series and it depends on the number of significant bits in the calculation. Increasing the number of significant bits increases the repeat period but will not remove it. The Digital Artifact is caused by the deterministic computation of the digital computer and its limited precision in performing mathematical operations.

To confirm this Digital Artifact, we generated a Hénon Map ($a = 1.31, b = 0.3$) time-series with various precision data in MATLAB. Fig. 2 shows the period of the chaotic time series for different levels of precision. For example, for 30 bits of precision the time series has a Digital Artifact period of about 10^4 data points. These simulation results were consistent with those of [14].

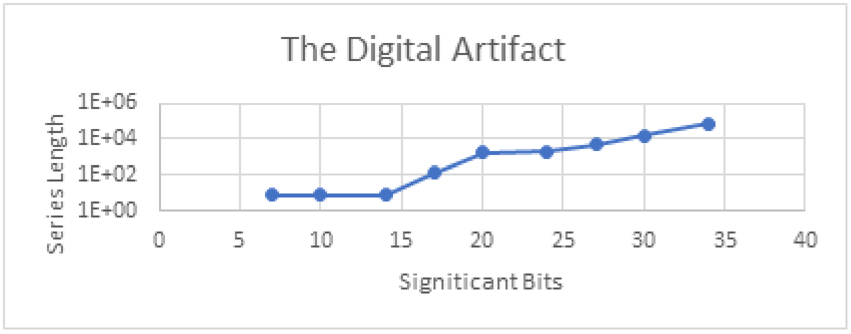

Fig. 2. The Digital Artifact.

4 Experimental Results

4.1 Data Set Generation

Table 1 shows four sets of time series data generated for use in our experiments. The Logistic Map Training data time-series was generated by MATLAB double precision based on Eq. (1). Note that for the double-precision Training data, the period of the Digital Artifact is greater than 10^6 – much too long to show up in the 20,000-point time series. All digital neural networks were trained by

Table 1. Time series data sets

Time series	***Input/Output***	***Generated by Hardware***	***Precision***
Training	Input	digital	double
DRNN9	output	digital	9 bits
DRNN18	output	digital	18 bits
OpticARNN	output	analog	7 bits

the MATLAB nntool with the GDX option enabled using the double precision Training data. In addition, they were trained in feed-forward mode to generate 20,000 (x, y) data points using Eq. (1). The OpticARNN was also trained using the Training time-series data. Simulated annealing and hardware-in-the-loop were used as previously discussed. A one-sided decision was used to always accept results which had better performance. Approximately 100 epochs were needed to reduce the training error to acceptable bounds.

All of the neural networks were configured with 2 input neurons (x and bias), five hidden neurons with the 'logsig' squashing function $f(s) = [1 + exp(-s)]^{-1}$, and one linear output neuron limited to the range [0,1].

After training, all networks were evaluated in recurrent mode. The initial input value was 0.2 for all networks. The output of the network for step n was presented as the new input for step $n + 1$, thus generating a time series. For

DRNN9, the output was rounded to 9 bits before recurrence, for DRNN18, it was rounded to 18 bits. The recurrence was executed for 20,000 iterations. The last 8184 points were evaluated in testing for chaos (8175 for DRNN18). Using the later values of the time series insured that the networks had time to get beyond any transient behavior and settle into a steady state.

Our initial precision estimate of OpticARNN was approximately 9 bits, which was used to set the DRNN9 rounding precision. Later, the OpticARNN precision was found to be between 7 and 8 bits. To be conservative, we kept the DRNN9 as 9 bits because that gave it a slight performance advantage over OpticARNN when analyzing results.

4.2 Testing for Compatibility with Chaos

There were two major difficulties involved in analyzing the neural networks for chaotic behavior.

The first was that the function mapping learned by the network was not exactly the same as the training data. Neural networks are universal approximators and learn the mapping to within a specified error. Normally, this is not a major issue - one trains the network, adjusts hidden layers, etc. until the approximation is good enough. Chaotic systems are aperiodic and very sensitive to initial conditions. Consequently, the trained OpticARNN chaotic behavior did not look exactly like the Training data. Analysis of the results, however, clearly demonstrated that OpticARNN was able to learn chaos while its digital counterparts could not. We anticipate that larger ARNNs and more effective hardware-in-the-loop training will generate a more accurate mapping.

This expectation is based on the work of Blank [14], who proposed that a small amount of true noise can cure the pathology. He also noted that even in the general case, one can verify the existence of a genuine trajectory of the system in the neighborhood of the numerical trajectory.

The second difficulty was that no test can prove that the dynamics of a system are chaotic. While finding periodicity in a time series shows that it is not chaotic, the best positive outcome that can be achieved is to show that the series is compatible with chaos[15]. We demonstrate OpticARNNs compatibility with chaos in our results.

4.3 Test Results

We ran two sets of tests for compatibility with chaos on our data sets: The Autocorrelation Test and the Largest Lyapunov Exponent (LLE) Test.

Autocorrelation Test. We used the autocorrelation test to determine periodicity in the outputs generated by the DRNNs and OpticARNN. Results are displayed in Table 2. As expected, the DRNN9 data showed a strong correlation to a periodic dynamical system. Inspection revealed that the Digital Artifact in the DRNN9 time series was exactly 22, confirming that DRNN9 is not aperiodic and, therefore, not chaotic. DRNN18 is similarly not chaotic because it contained a Digital Artifact of 109 steps.

Table 2. Autocorrelation Results

Time Series	*Periodic*	*Period*	*Compatible with Chaos?*
DRNN9	yes	22 Steps	No
DRNN18	yes	109 Steps	No
OpticARNN	no	N/A	Yes

Figure 3 shows the autocorrelation function for DRNN9 and OpticARNN. The dissimilarities between DRNN9 and OpticARNN figures highlight the differences between periodic and chaotic time series.

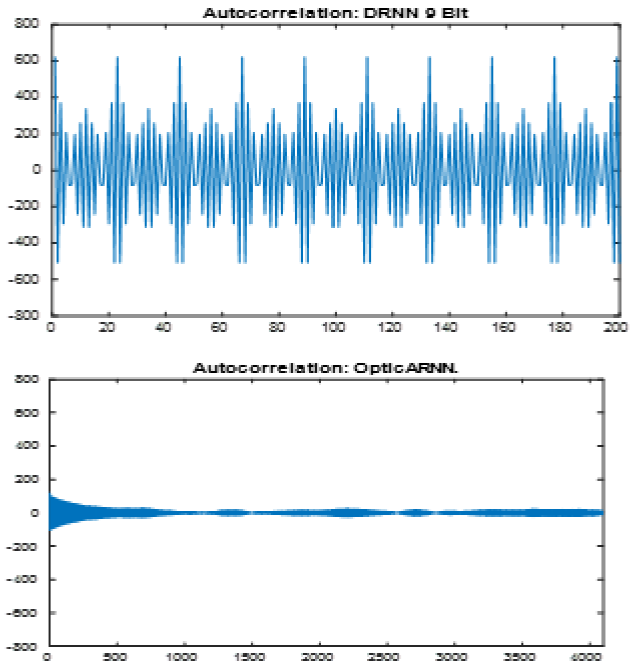

Fig. 3. Autocorrelations of DRNN9 and OpticARNN

Largest Lyapunov Exponent Test. When the Lyapunov Exponent spectrum of a system is computed, the system is compatible with chaos if it has positive exponent(s). Usually, only the largest Lyapunov exponent (λ) needs be estimated from the time series data. We implemented the Rosenstein et. al [16] method for calculating λ because it uses all available data in the calculation and is robust to noise, time series length, embedding dimension, and time lag.

The first step in calculating the largest Lyapunov exponent is to reconstruct the phase space. By Takens' theorem, the technique of translating a time series into p-dimensional space, called time-lag embedding, yields dynamics that are geometrically similar to the original series[16]. The time-lag parameter τ selects the interval between time series points making up the embedding: 1 means choose each point, 2 means choose every other point, etc. The embedding dimension parameter is p. We avoid the problem of needing to calculate several Lyapunov exponents at different time-lag and embedding dimensions because the optimal values are well known for the Logistic Map (τ = 1; p = 2)[16].

Next, for all points in the reconstructed space, the nearest neighbor of each point is determined by minimizing the Euclidean distance between it and all other points. Each pair of neighbors represent the initial conditions of two nearby trajectories. The mean rate of divergence of these trajectories is used to estimate the largest Lyapunov exponent.

The theoretical value of λ reported in Rosenstein et al. [16] for the Logistic Map with r = 4.0 is 0.693. We calculated a value of λ = 0.624 for the Training data (r = 3.99). Our experimental results are displayed in Table 3. Note that "N+" indicates that no positive Lyapunov exponent was found, consequently the dynamics of that time series is not compatible with chaos.

Table 3. Largest Lyapunov Exponent

Time Series	λ	***Compatible with Chaos?***
DRNN9	N+	No
DRNN18	N+	No
OpticARNN	1.083	Yes

5 Discussion

There has been an increased interest in analog computational devices, both electronic[17, 18] and optically based[19, 20]. There are significant energy, speed, size, and cost advantages of these devices. However, as far as we know, none of these designs have used super-Turing theoretical considerations in their design. We believe that these considerations are important to making these devices have maximum computational abilities. There is a new level of potential computational power possible, but this potential is underrecognized and underutilized.

Beggs et al. [6] suggested that systems computing in ***BPP/log**** "will not support programming, since programming in such a context turns to be a settlement of a real number" but would likely be better suited to learning tasks. This suggests that fully analog neural recurrent pathways are important to increase the power of our physical implementation because they are capable of very effective types of learning.

We also learned that the practical use of ARNNs should involve integration with conventional digital systems. Digital computers have advantages in memory, speed, flexibility and have a large number of software development tools

- compilers, integrated development systems, GUIs, GPUs, and access to the internet. Most data are in digital form.

It has also been suggested that the Digital Artifact can be removed by using larger number of significant digits in the calculations, or by introducing pseudo-random noise in the calculation. This procedure could increase the period to the point where it is not detectable in the simulation. However, the artifact will still be present. We believe that this suggestion, while true, misses the point - the OpticARNN inherently does this in its computation process, without consuming ever increasing memory and time. Moreover, the OpticARNN shows no sign of the Digital Artifact even in a region of significance where digital computer results show drastic pathology.

6 Conclusion

We presented OpticARNN, an optical analog computer which is based on a super-Turing computation theory. We devised a test for super-Turing capability based on the idea that realized Turing machines, implemented as digital computers, always produce a periodic time series when modeling chaotic systems.

We trained recurrent neural networks, both digital and analog, on the chaotic Logistic Map time series. Their outputs were subsequently tested for compatibility with chaos using two well known metrics, autocorrelation and the largest Lyapunov exponent. Both metrics confirmed that the output produced by OpticARNN was compatible with chaos. In contrast, both metrics for the digital recurrent neural networks of approximately equal precision and twice the precision of OpticARNN were shown not to be compatible with chaos.

The DRNNs output exhibited an expected periodicity that is due to the deterministic nature of TMs and the finite precision inherent in digital computers. Significantly, the OpticARNN output never displayed the Digital Artifact, even in time series output 372 times longer than was needed for the Digital Artifact to show up in DRNN9, its digital sibling. While robust systems can program their way around it, the Digital Artifact cannot be eliminated on digital computers that operate on 0s and 1s.

We have empirically shown that a real, physical computing machine can be built to solve problems beyond that of digital machines. However, we have only scratched the surface of analog computing.

7 Acknowledgments

This work is supported by U.S. National Science Foundation Grant ECCS-1201790. We thank student Meghan Sims for work on the chaotic neural network simulations.

References

1. Goldin D, and Wegner P (2008). The interactive nature of computing: Refuting the strong ChurchTuring thesis. Minds and Machines, 1-26.

2. Siegelmann, H. (1998). Neural Networks and Analog Computation Beyond the Turing Limit. Boston: Birkhauser.
3. Siegelmann, H. (1999). Stochastic Analog Networks and Computational Complexity. *Journal of Complexity*, 15(4), 451475. doi.org/10.1006/jcom.1999.0505
4. Siegelmann, H. (2013). Turing on Super-Turing and adaptivity. *Progress in biophysics and molecular biology* (Vol. 113). Elsevier Ltd. doi.org/10.1016/j.pbiomolbio.2013.03.013
5. Turing, A. M. (1950.) Computing Machinery and Intelligence, Mind, 59, pp. 433-460.
6. Beggs, E., Cortez, P., Costa, J. F., and Tucker, J. V. (2016). A Hierarchy for BPP//log Based on Counting Calls to an Oracle. In A. Adamatzky (Ed.), *Emergent Computation: A Festschrift for Selim G. Akl* (Vol. 24, pp. 3956). Springer International Publishing. doi.org/10.1007/978-3-319-46376-6_3
7. Turing, A. M.: Can Digital Computers Think? BBC Radio Program 15 May 1951
8. McDonnel, Mark D, and Derek Abbott (2009) What Is Stochastic Resonance? Definitions, Misconceptions, Debates, and Its Relevance to Biology. PLoS Comput Biol. 2009 May; 5(5): e1000348 https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2660436/
9. Wray, Jonathan, and Gary GR Green. (1995). "Neural networks, approximation theory, and finite precision computation." *Neural networks* 8, no. 1 : 31-37.
10. Casimir, Hendrick BG. (1948). "On the attraction between two perfectly conducting plates." In *Proceedings of the KNAW*, vol. 51, no. 7, pp. 793-795. 1948.
11. Younger, S., Redd, E., and Siegelmann, H. (2014). Development of Physical Super-Turing Analog Hardware. In O. H. Ibarra, L. Kari, and S. Kopecki (Eds.), *Unconventional Computation and Natural Computation 2014* (Vol. 8553, pp. 379391). Springer International Publishing. doi.org/10.1007/978-3-319-08123-6
12. Goodman, J. W. (2005). *Introduction to Fourier optics.* Englewood: Roberts and Company Publishers.
13. Frye, R. C., Rietman, E. A., and Wong, C. C. (1991). Back-propagation learning and nonidealities in analog neural network hardware. *IEEE Transactions on Neural Networks*, 2(I), 110117. doi.org/10.1109/72.80296
14. Blank, M. (1994). Pathologies generated by round-off in dynamical systems. Physica D: Nonlinear Phenomena, 78(12), 93114. doi.org/10.1016/0167-2789(94)00103-0
15. Kaplan, D., and L. Glass, (1995). *Understanding Nonlinear Dynamics.* New York: Springer-Verlag.
16. Rosenstein, M. T., Collins, J. J., and Luca, C. J. De. (1993). A practical method for calculating largest Lyapunov exponents from small data sets. *Physica D: Nonlinear Phenomena*, 65(1-2), 117-134.
17. Hasler, Jennifer. (2016). "Opportunities in physical computing driven by analog realization." In *Re-booting Computing (ICRC), IEEE International Conference on*, pp. 1-8. IEEE.
18. Modha, Dharmendra S., Introducing a Brain-inspired Computer: TrueNorth's neurons to revolutionize system architecture. http://www.research.ibm.com/articles/brain-chip.shtml
19. http://optalysys.com/
20. Saade, Alaa, Francesco Caltagirone, Igor Carron, Laurent Daudet, Anglique Drmeau, Sylvain Gigan, and Florent Krzakala. (2016). "Random Projections through multiple optical scattering: Approximating kernels at the speed of light." In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pp. 6215-6219. IEEE.