Spiking LCA in a Neural Circuit with Dictionary Learning and Synaptic Normalization

Author(s):
Chavez Arana, Diego; Renner, Alpha; Sornborger, Andrew

Publication date:
2023-04-12

Permanent link:
https://doi.org/10.3929/ethz-b-000655619

Originally published in:
ACM International Conference Proceeding Series, https://doi.org/10.1145/3584954.3584968

Spiking LCA in a Neural Circuit with Dictionary Learning and Synaptic Normalization

Diego Chavez Arana
Los Alamos National Laboratory,
Los Alamos;
New Mexico State University,
Las Cruces
New Mexico, USA
saidcha@nmsu.edu

Alpha Renner
Institute of Neuroinformatics
University of Zurich
Zurich, Switzerland
alpren@ini.uzh.ch

Andrew Sornborger
Los Alamos National Laboratory
Los Alamos, New Mexico, USA
sornborg@lanl.gov

ABSTRACT

The Locally Competitive Algorithm (LCA) [17, 18] was put forward as a model of primary visual cortex [14, 17] and has been used extensively as a sparse coding algorithm for multivariate data. LCA has seen implementations on neuromorphic processors, including IBM's TrueNorth processor [10], and Intel's neuromorphic research processor, Loihi, which show that it can be very efficient with respect to the power resources it consumes [8]. When combined with dictionary learning [13], the LCA algorithm encounters synaptic instability [24], where, as a synapse's strength grows, its activity increases, further enhancing synaptic strength, leading to a runaway condition, where synapses become saturated [3, 15]. A number of approaches have been suggested to stabilize this phenomenon [1, 2, 5, 7, 12]. Previous work demonstrated that, by extending the cost function used to generate LCA updates, synaptic normalization could be achieved, eliminating synaptic runaway [7]. It was also shown that the resulting algorithm could be implemented in a firing rate model [7]. Here, we implement a probabilistic approximation to this firing rate model as a spiking LCA algorithm that includes dictionary learning and synaptic normalization. The algorithm is based on a synfire-gated synfire chain-based information control network in concert with Hebbian synapses [16, 19]. We show that this algorithm results in correct classification on numeric data taken from the MNIST dataset.

LA-UR-22-33004

KEYWORDS

Hebbian learning, synaptic stability, synaptic saturation, synfire-gated synfire chains, sparse coding

ACM Reference Format:
Diego Chavez Arana, Alpha Renner, and Andrew Sornborger. 2023. Spiking LCA in a Neural Circuit with Dictionary Learning and Synaptic Normalization. In *Neuro-Inspired Computational Elements Conference (NICE 2023), April 11–14, 2023, San Antonio, TX, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3584954.3584968

NICE 2023, April 11–14, 2023, San Antonio, TX, USA

ACM ISBN 978-1-4503-9947-0/23/04.
https://doi.org/10.1145/3584954.3584968

1 METHODS

1.1 The LCA Cost and Update Equations

The LCA is a recursive algorithm constructed from an optimization problem where updates are computed from gradients of a cost function.

$$\min_{\mathbf{a}} \left(\left\| \mathbf{s} - \sum_{m=1}^{M} a_m \boldsymbol{\phi}_m \right\|_2^2 + \lambda \left\| \mathbf{a} \right\|_1 \right) . \tag{1}$$

Here, $\mathbf{s}$ denotes a target signal, often an image, of interest. The set $\{a_i : i \in 1, \ldots, M\}$ denotes coefficients and the set $\{\boldsymbol{\phi}_i : i \in 1, \ldots, M\}$ represents dictionary elements. Dictionary elements may be fixed or learned. The first term in (1) represents the distance of a linear combination of the dictionary elements from the target signal. The second term in (1), with parameter λ, represents the L_1 norm of the vector $\mathbf{a}$. When minimized, all but a few elements of $\mathbf{a}$ are non-zero, resulting in a sparse representation of $\mathbf{s}$.

In order to implement this algorithm in a neural circuit, an internal variable, $\mathbf{u}$, is introduced. In the neural representation, $\mathbf{a} = T_\lambda(\mathbf{u})$, where T_λ is a neural activity function, which thresholds u. Thus, the a_i are represented by spikes, whereas the u_i are represented by membrane potentials in the neural circuit.

A dynamical equation to implement the minimization in (1) [17, 18] is

$$u_m^{\text{new}} = u_m^{\text{old}} + \frac{1}{\tau} \left(\langle \boldsymbol{\phi}_m, \mathbf{s}(t) \rangle - u_m(t) - \sum_{n \neq m} \langle \boldsymbol{\phi}_m, \boldsymbol{\phi}_n \rangle a_n(t) \right) , \tag{2}$$

where $\langle a, b \rangle$ denotes the inner product between vectors $\mathbf{a}$ and $\mathbf{b}$. Here, the sum in the third term on the right implements a recurrent inhibition that enforces the sparsification of the representation of $\mathbf{s}$ as a linear combination of the dictionary elements, $\{\boldsymbol{\phi}_i\}$ with coefficients a_n.

By itself, (2) implements LCA with no dictionary learning or synaptic normalization. An update rule for dictionary learning may be derived by taking the gradient of (1) with respect to the dictionary elements [25], $\boldsymbol{\phi}_i$. This results in the synaptic update

$$\Phi^{\text{new}} = \Phi^{\text{old}} + \mathbf{a} \otimes (\mathbf{s} - \Phi \mathbf{a}) . \tag{3}$$

In this update, $\otimes$ denotes the outer product and the dictionary matrix $\Phi = [\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \ldots, \boldsymbol{\phi}_M]$.

To normalize the synaptically encoded dictionary elements, we add a term to the cost that evaluates the summed distance between column vectors encoded in a dictionary matrix, Φ, with unit distance.

Explicitly,

$$C(\Phi) = \frac{1}{2} \sum_{k \in 1,\ldots,K} (\boldsymbol{\phi}_k^T \boldsymbol{\phi}_k - 1)^2 \,. \tag{4}$$

This convex cost evaluates to 0 if and only if the dictionary elements $\{\boldsymbol{\phi}_k\}$ are all of unit length.

We make use of gradient descent to minimize the cost, whose derivative with respect to the vector $\boldsymbol{\phi}_k$ is

$$\frac{\partial C}{\partial \boldsymbol{\phi}_k} = (|\boldsymbol{\phi}_k|^2 - 1)\boldsymbol{\phi}_k \,. \tag{5}$$

Thus, with update

$$\boldsymbol{\phi}_k^{\text{new}} = \boldsymbol{\phi}_k^{\text{old}} - \eta \frac{\partial C}{\partial \boldsymbol{\phi}_k} \,, \tag{6}$$

$\boldsymbol{\phi}_k$ will converge to unit length asymptotically with unit probability.

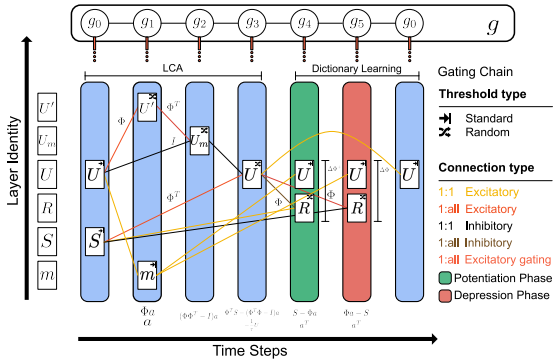

Figure 1: Functional connectivity of the LCA circuit. Neuronal population names are shown on the left, and their activity is represented at their activation time. Columns correspond to gating times during the circuit operation. The figure shows LCA with dictionary learning, where learning happens at the last phase of the circuit operation. Connections between populations can be one-to-one, where the presynaptic neuron is connected to its single homologous postsynaptic neuron. Alternatively, connections can be one-to-all, where each presynaptic neuron is connected to all of the neurons in the postsynaptic population. Synapses can be excitatory or inhibitory. A legend of connection types is indicated on the lower right side of the figure. The firing threshold type is indicated on the upper right side of the neuronal population. Synaptic update of the dictionary happens at the potentiation (green) and depression (red) phases. The gating chain causes the potentiation of the neuronal population voltages and hence firing activity at the indicated gating times; thus, information travels only if excitatory gating activates a population. The postsynaptic outcome of information depends on the connection type between populations. Below each gating column, the computed information in U and R populations is denoted mathematically.

1.2 Neural Considerations for Implementing LCA

In order to implement the updates in this algorithm in a neural circuit, we consider a spiking model of a synfire-gated synfire chain

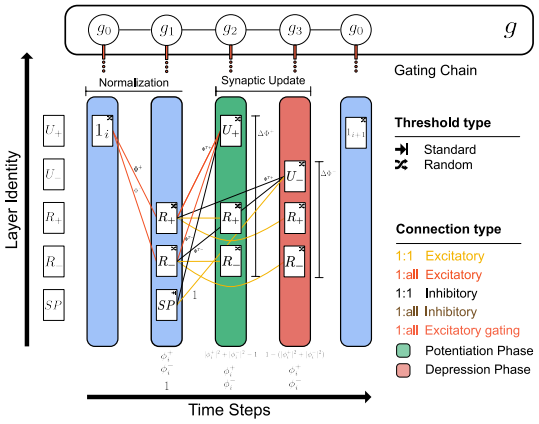

Figure 2: Functional connectivity of the normalization circuit. This diagram is depicted analogously to Fig. 1, where layer activities are shown as a function of time. See description in Fig. 1.

[16, 20–23, 27]. Here, information is gated via potentiating spikes from a presynaptic neuron to a postsynaptic neuron, implementing the synaptic transform, Φ,

$$u_{\text{post}} \xleftarrow{\text{gate}} T_\lambda(\Phi\, \mathbf{b}_{\text{pre}}) \,, \tag{7}$$

where $\mathbf{b}_{\text{pre}}$ is a vector of presynaptic spikes from a presynaptic neuron and u_{post} is a postsynaptic current.

The gate itself is implemented synaptically by connecting a neuron in the synfire chain with a postsynaptic neuron in $\mathbf{b}$. Voltages (or, similarly, thresholds) are set such that neither the gating neuron nor the presynaptic (information-carrying) neuron will cause the postsynaptic neuron to fire. However, if the spikes are concurrent, then the information-carrying spike is potentiated by the gating spike, and the information propagates downstream.

It is important to note that in some parts of our neural circuit, instead of T_λ, the usual activity function with a fixed threshold, we will use T_λ^{rand}. T_λ^{rand} is a random threshold activity function (implemented on Intel's neuromorphic research processor, Loihi, for instance). We use this activity function in order to generate spikes with a firing rate proportional to $\boldsymbol{\phi}_i$ in the following way: we gate the value 1 into an element of $\mathbf{a}_{\text{pre}}$, i.e., $\mathbf{a}_{\text{pre}} = [0, \ldots, 0, 1, 0, \ldots, 0] \equiv 1_i$. With a random threshold,

$$f_{\text{post}} = \langle T_\lambda^{\text{rand}}(\Phi\, \mathbf{b}_{\text{pre}}) \rangle \approx \boldsymbol{\phi}_i \,. \tag{8}$$

Above, $\langle\rangle_T$ indicates a time average and 1_i indicates the value 1 in the i'th element of $\mathbf{a}_{\text{pre}}$.

By using this gated transform capability, in concert with a single rank-one Hebbian update,

$$\Phi \leftarrow \Phi + \mathbf{b}_{\text{pre}} \otimes \mathbf{b}_{\text{post}} \,, \tag{9}$$

we can probabilistically implement a synaptic update,

$$\Phi \leftarrow \Phi + \langle \mathbf{b}_{\text{pre}}^t \otimes \mathbf{b}_{\text{post}}^t \rangle_T \,. \tag{10}$$

1.3 Spiking Neural Circuit Implementation of LCA with Dictionary Learning and Normalization

In Fig. 1, we depict the functional connectivity of the combined LCA and dictionary learning circuits.

At the beginning of each iteration, the image input of the LCA circuit is encoded in S, and activity, $a = T_\lambda(U)$, is gated into the circuit via gating population g_0. This activity is then gated via the g_1 gating population through the synaptic matrix, Φ, to U' via a random threshold, causing the U' neural register to spike with probability, $\Pr(U') = \Phi a$. The activity in U is simultaneously propagated to the memory population, m, via a standard threshold such that it contains a. U' is then gated via g_2 to U_m with a random threshold and U is gated with delay through one-to-one inhibitory synapses such that U_m now spikes with probability $\Pr(U_m) = (\Phi\Phi^T - I)a$. This spiking activity is then gated back to the register U, which, at this point, contains a new iterate of u (see Eq. 2).

In the next phase of the circuit, updates to the dictionary are made. The activity in U is propagated through the synaptic connectivity, Φ, and the input image is simultaneously propagated to R, giving $P(R) = S - \Phi a$ in the postsynaptic population. At the same time, the contents of the memory register, $m = a$, are gated with delay to U such that $U = a$. This causes a Hebbian update since both a and $S - \Phi a$ are simultaneously in the pre- and postsynaptic populations on either side of the synapses containing the dictionary, Φ. Similar updates are performed in the subsequent step, but the update is now to the negative weights of Φ. Additionally, the other sets of synapses encoding Φ (e.g., between S and U) in the circuit must be updated (not shown) in a similar way.

Finally, U is gated with delay to itself, where a memory register to hold the contents of U is not shown.

We note here that since we cannot encode negative numbers in spikes (spikes cannot be tagged as representing negative information), the registers in the LCA neural circuit actually represent pairs of values, with one of each pair representing 'positive' spikes, and the other member representing 'negative' spikes. In Fig. 2, since the normalization circuit is somewhat simpler, we explicitly show how this is done.

The functional connectivity and operation of the normalization circuit are depicted in Fig. 2. The circuit is iterated as many times as there are dictionary elements, i.e., separate updates are necessary to normalize each dictionary element in the synaptic connectivity, Φ. This circuit is interleaved with the LCA and dictionary learning circuits are shown in Fig. 1.

Propagation of information through the normalization circuit starts at the U_+ population, into which is gated the indicator vector 1_i. This element selects the dictionary element to be probabilistically normalized subsequently by the circuit. By gating 1_i through the synapses that encode the positive elements of the dictionary, Φ^+, into R_+, where R_+ has a random threshold, we find that $\Pr(R_+) = \phi_i^+$, and thus, the R_+ register will, on average, encode the i'th dictionary element, ϕ_i. Using the same logic, 1_i is gated through Φ_- encoding $\Pr(R_-) = \phi_i^-$ in R_-. The information in the $R_{+,-}$ register is then gated through the transposed dictionary, depositing $|\phi_i^+|^2 + |\phi_i^-|^2 - 1 = |\phi_i|^2 - 1$ in the i'th element of the U_+ register. Note that SP is a neuron that outputs a constant value (here indicated as the number 1), which is subtracted from the input to U_+, determining the vector length to which each dictionary element is normalized.

The activity in $R_{+,-}$ is gated back to these same populations (this can be done with delays or with memory registers). At this point, $|\phi_i^+|^2 + |\phi_i^-|^2 - 1$ has been gated to and probabilistically encoded in the i'th element of the U_+ register, while simultaneously the vectors ϕ_i^+ and ϕ_i^- are encoded in $R_{+,-}$. Via a Hebbian update, the synaptic weights are updated, with average increments of $(|\phi_i^+|^2 + |\phi_i^-|^2 - 1)\phi_i^+$ and $(|\phi_i^+|^2 + |\phi_i^-|^2 - 1)\phi_i^-$ between U_+ and $R_{+,-}$, respectively.

The dictionary elements between U_- and $R_{+,-}$ are also updated with connections not shown for clarity.

This normalization circuit is iterated with $i = 1, \ldots, M$, where n is the number of vectors making up the dictionary.

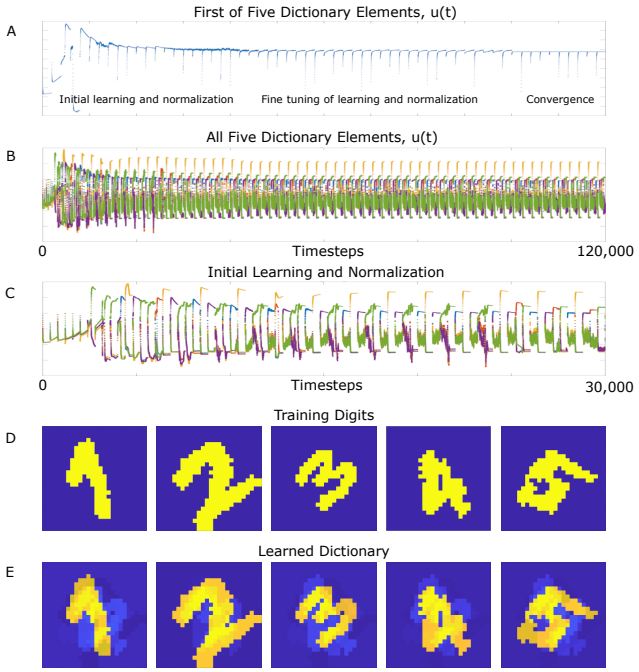

Figure 3: Unsupervised classification with LCA, dictionary learning, and dictionary normalization during training with repeated presentations of five digits. A) The internal voltage, u, as a function of the timestep, t, over 120,000 training steps. Note that only the training cycles where u responded to input were included for clarity. All timesteps are shown in B) for all five digits. In A), we note three epochs during training, an initial learning and normalization period, fine-tuning once rough dictionary elements have been learned, then convergence to stereotypical responses. C) A closeup of the initial epoch of 30,000 timesteps during which initial learning and dictionary normalization are dominant. D) The dataset on which training is performed. These digits are repeatedly sent to the neural circuit. E) The digits learned by the combined LCA, dictionary learning, and normalization algorithm.

1.4 Results

In order to test our probabilistic approach to implementing LCA (with dictionary learning and normalization) in neural circuits controlled with synfire-gated synfire chains, we trained on five numbers from the MNIST database [9]. Neural circuit simulation was performed with Matlab.

Training with this simple training set allowed us to see how the circuit response converged to stable, sparse classifications of the digits. More complex behavior would arise if we used the whole MNIST dataset and increased the dictionary size, but troubleshooting our approach would be more difficult.

In Fig. 3A, we show the time course of the membrane potential, u, of one of the sets of 5 neurons in U during unsupervised learning. At the beginning of the time course, the membrane potential rose as the dictionary element was modified due to the input data. As the digit that this neuron begins to encode was repeatedly presented, its membrane potential rose due to dictionary learning and overshot a value above which the normalization term in the cost became dominant. This caused the response to decrease and stabilize.

Over the course of the training, the response became increasingly finely tuned to its digit, completely stabilized, and this digit (a one) was recognized with high probability.

Fig. 3B shows the response of all five dictionary elements that were used for training. Note that all of these neurons stabilized at roughly the same time, and by the end of training, the responses were all stable.

Fig. 3C depicts the early learning and normalization phase of all U neurons, each responding to an individual digit.

The training data is shown in Fig. 3D, and the learned dictionary elements are shown in Fig. 3C.

2 CONCLUSIONS AND OUTLOOK

This paper presented a successful means of constructing spiking neural circuits to implement the Locally Competitive Algorithm with dictionary learning and dictionary normalization.

Our approach made use of synfire-gated synfire chains to control the propagation of information within the neural system as a whole and also to control Hebbian synaptic updates. Without this control, which allowed us to implement synaptic normalization, synaptic weights could easily encounter runaway conditions and saturate, destabilizing dictionary learning.

An important mechanism we introduced was using random thresholds to translate information encoded in synapses into firing rates [7] that could be used for Hebbian updates to synapses encoding the LCA dictionary.

Previous neuromorphic implementations of LCA have demonstrated significant power reductions with the LCA algorithm on a range of hardware substrates [4], including memristive [26], fully analog [6], CMOS [8, 11]. However, few of these [24] attack the dictionary learning problem on-chip.

Further work remains to understand how learning will perform in the spiking LCA algorithm that we present here when it is exposed to more complex datasets. Also, further studies will need to be performed to investigate how the algorithm will scale. Additionally, this spiking algorithm will need to be tested on neuromorphic hardware such as Intel's Loihi chip and memristive systems. This research is currently underway.

ACKNOWLEDGMENTS

This work was initially supported by the LANL ASC Beyond Moore's Law project (A.T.S.) and by the US Department of Energy National Nuclear Security Administration's Office of Defense Nuclear Nonproliferation Research & Development (DNN R&D) at Los Alamos National Laboratory under contract 89233218CNA000001. (D.C.A, A.T.S.). LANL approval designation: LA-UR-22-33004

REFERENCES

[1] LF Abbott. 2003. Balancing homeostasis and learning in neural circuits. *Zoology* 106, 4 (2003), 365–371.
[2] Larry F Abbott and Sacha B Nelson. 2000. Synaptic plasticity: taming the beast. *Nature neuroscience* 3, 11 (2000), 1178–1183.
[3] Christian Balkenius, Jan Morén, et al. 1998. Computational models of classical conditioning: a comparative study. (1998).
[4] Arindam Basu, Lei Deng, Charlotte Frenkel, and Xueyong Zhang. 2022. Spiking neural network integrated circuits: A review of trends and future directions. In *2022 IEEE Custom Integrated Circuits Conference (CICC)*. IEEE, 1–8.
[5] Elie L Bienenstock, Leon N Cooper, and Paul W Munro. 1982. Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex. *Journal of Neuroscience* 2, 1 (1982), 32–48.
[6] Fred N Buhler, Peter Brown, Jiabo Li, Thomas Chen, Zhengya Zhang, and Michael P Flynn. 2017. A 3.43 TOPS/W 48.9 pJ/pixel 50.1 nJ/classification 512 analog neuron sparse coding neural network with on-chip learning and classification in 40nm CMOS. In *2017 Symposium on VLSI Circuits*. IEEE, C30–C31.
[7] Diego Chavez Arana, Alpha Renner, and Andrew Sornborger. 2022. A Neuromorphic Normalization Algorithm for Stabilizing Synaptic Weights with Application to Dictionary Learning in LCA. In *Neuro-Inspired Computational Elements Conference*. 58–60.
[8] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham N. Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios D. Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, Yuyun Liao, Chit-Kwan Lin, Andrew Lines, Ruokun Liu, Deepak Mathaikutty, Steven McCoy, Arnab Paul, Jonathan Tse, Guruguhanathan Venkataramanan, Yi-Hsin Weng, Andreas Wild, Yoonseok Yang, and Hong Wang. 2018. Loihi: A Neuromorphic Manycore Processor with On-Chip Learning. *IEEE Micro* 38, 1 (2018), 82–99. https://doi.org/10.1109/MM.2018.112130359
[9] Li Deng. 2012. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine* 29, 6 (2012), 141–142.
[10] Kaitlin L Fair, Daniel R Mendat, Andreas G Andreou, Christopher J Rozell, Justin Romberg, and David V Anderson. 2019. Sparse coding using the locally competitive algorithm on the TrueNorth neurosynaptic system. *Frontiers in neuroscience* 13 (2019), 754.
[11] Charlotte Frenkel, Martin Lefebvre, Jean-Didier Legat, and David Bol. 2018. A 0.086-mm ^{2}12.7-pj/sop 64k-synapse 256-neuron online-learning digital spiking neuromorphic processor in 28-nm CMOS. *IEEE transactions on biomedical circuits and systems* 13, 1 (2018), 145–158.
[12] Erkki Oja. 1982. Simplified neuron model as a principal component analyzer. *Journal of mathematical biology* 15, 3 (1982), 267–273.
[13] Bruno A Olshausen and David J Field. 1996. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature* 381, 6583 (1996), 607–609.
[14] Bruno A Olshausen and David J Field. 2004. Sparse coding of sensory inputs. *Current opinion in neurobiology* 14, 4 (2004), 481–487.
[15] Bernd Porr and Florentin Wörgötter. 2007. Learning with "relevance": using a third factor to stabilize Hebbian learning. *Neural computation* 19, 10 (2007), 2694–2719.
[16] Alpha Renner, Forrest Sheldon, Anatoly Zlotnik, Louis Tao, and Andrew Sornborger. 2021. The Backpropagation Algorithm Implemented on Spiking Neuromorphic Hardware. *arXiv preprint arXiv:2106.07030* (2021).
[17] Christopher J Rozell, Don H Johnson, Richard G Baraniuk, and Bruno A Olshausen. 2008. Neurally plausible sparse coding via thresholding and local competition. *Neural Comput* 20, 10 (2008), 2526–2563.
[18] Christopher J Rozell, Don H Johnson, Richard G Baraniuk, and Bruno A Olshausen. 2008. Sparse coding via thresholding and local competition in neural circuits. *Neural computation* 20, 10 (2008), 2526–2563.
[19] Yuxiu Shao, Andrew T Sornborger, and Louis Tao. 2016. A pulse-gated, predictive neural circuit. In *2016 50th Asilomar Conference on Signals, Systems and Computers*. IEEE, 1051–1055.

[20] Andrew Sornborger, Louis Tao, Jordan Snyder, and Anatoly Zlotnik. 2019. A Pulse-gated, Neural Implementation of the Backpropagation Algorithm. In *Proceedings of the 7th Annual Neuro-inspired Computational Elements Workshop*. ACM, 10.

[21] A.T. Sornborger, Z. Wang, and L. Tao. 2015. A mechanism for graded, dynamically routable current propagation in pulse-gated synfire chains and implications for information coding. *J. Comput. Neurosci.* (August 2015). https://doi.org/10.1007/s10827-015-0570-8

[22] C. Wang, Z.C. Xiao, Z. Wang, A.T. Sornborger, and L. Tao. 2015. A Fokker-Planck approach to graded information propagation in pulse-gated feedforward neuronal networks. *ArXiv* 1512.00520 (Dec 2015).

[23] Z. Wang, A.T. Sornborger, and L. Tao. 2016. Graded, dynamically routable information processing with synfire-gated synfire chains. *PLoS Comp Biol* 12 (2016), 6. https://doi.org/10.1371/journal.pcbi.1004979

[24] Yijing Watkins, Edward Kim, Andrew Sornborger, and Garrett T Kenyon. 2020. Using Sinusoidally-Modulated Noise as a Surrogate for Slow-Wave Sleep to Accomplish Stable Unsupervised Dictionary Learning in a Spike-Based Sparse Coding Model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 360–361.

[25] Yijing Watkins, Austin Thresher, Peter F Schultz, Andreas Wild, Andrew Sornborger, and Garrett T Kenyon. 2019. Unsupervised dictionary learning via a spiking locally competitive algorithm. In *Proceedings of the International Conference on Neuromorphic Systems*. 1–5.

[26] Walt Woods and Christof Teuscher. 2018. Fast and accurate sparse coding of visual stimuli with a simple, ultralow-energy spiking architecture. *IEEE transactions on neural networks and learning systems* 30, 7 (2018), 2173–2187.

[27] Z.C. Xiao, B.X. Wang, A.T. Sornborger, and L. Tao. 2018. Mutual information and information gating in synfire chains. *Entropy* 20 (2018), 102. https://doi.org/10.3390/e20020102