Programming Molecular Systems To Emulate a Learning Spiking Neuron

Jakub Fil,* Neil Dalchau, and Dominique Chu

ABSTRACT: Hebbian theory seeks to explain how the neurons in the brain adapt to stimuli to enable learning. An interesting feature of Hebbian learning is that it is an unsupervised method and, as such, does not require feedback, making it suitable in contexts where systems have to learn autonomously. This paper explores how molecular systems can be designed to show such protointelligent behaviors and proposes the first chemical reaction network (CRN) that can exhibit autonomous Hebbian learning across arbitrarily many input channels. The system emulates a spiking neuron, and we demonstrate that it can learn statistical biases of incoming inputs. The basic CRN is a minimal, thermodynamically plausible set of microreversible chemical equations that can be analyzed with respect to their energy requirements. However, to explore how such chemical systems might be engineered de novo, we also propose an extended version based on enzyme-driven compartmentalized reactions. Finally, we show how a purely DNA system, built upon the paradigm of DNA strand displacement, can realize neuronal dynamics. Our analysis provides a compelling blueprint for exploring autonomous learning in biological settings, bringing us closer to realizing real synthetic biological intelligence.

KEYWORDS: Hebbian learning, spiking neurons, DNA strand displacement, autonomous learning, biochemical intelligence

INTRODUCTION

While intelligent behaviors are usually associated with higher organisms that have a nervous system, adaptive and protointelligent behaviors are well documented in unicellular organisms. Examples include sensing,1–3 chemotaxis,4,5 or diauxic growth.6–8 This begs the question whether it is possible to rationally build molecular systems that show protointelligent behaviors and can be used as machines to monitor or control their chemical environment at a microscopic scale. Systems of this type could find applications in areas such as drug delivery, bioprocessing, or biofabrication.

As a step in this direction, we will probe how artificial intelligence can be realized in molecular systems. More specifically, we will show how to realize artificial neurons, as they are widely used in computer science as components of neural networks.9 Individual artificial neurons are simple machines but nevertheless show a remarkable ability to learn from observation. For the purpose of this article, we will consider a particular type of neuron, a spiking neuron (SN). SNs are widely used in machine learning,10,11 and it is well known that they have significant learning capabilities12,13 including principal component analysis,14 recognition of handwriting,15 or classification of fighter planes.16 There are a number of different models of SNs in the literature. Commonly a SN has an internal state, usually represented by a positive real number. The internal state may decay, which means that it reduces over time with some rate. The internal state variable increases when the SN receives a stimulus (an input spike) via one of its N input channels. Importantly, these input channels are weighted. The higher the weight, the more the internal state variable increases following an input spike through this channel. This weighting is crucial for the behaviors of the neuron. Consequently, “learning”, in the context of neural networks, normally means adjusting the weights.

There have been numerous attempts to build neurons in chemical systems. The earliest dates back to the 1980s by Okamoto and collaborators,17 who showed that certain biochemical systems implement the McCulloch–Pitts neuronal equations. Later, a mathematical description of a neuron was proposed,18 but this system had no ability to learn. Banda et al.19 used artificial chemistry to emulate an artificial neuron and a fully fledged feed-forward neural network20 which could solve the XOR problem. Their model requires regular interventions by outside operators, however. Besides these simulation studies, there have also been attempts to implement learning in vivo,21–24 but again, these systems are not autonomous: they rely on iterative measurement and manipulation protocols, which limit their practical deployment as computing machines within a molecular environment.

Received: December 13, 2021
Published: May 27, 2022
An attractive concept of learning that avoids the need to monitor the molecular neurons is Hebbian learning. This concept originated from neuroscience but is now widely used in artificial intelligence to train neural networks. The basic idea of Hebbian learning is that the connection between neurons that fire at the same time is strengthened. This update scheme is attractive because unlike many other learning algorithms, it does not require evaluating an objective function, which would be difficult to achieve in general with chemical networks.

To illustrate the basic idea of Hebbian learning—or associative learning as it is often called when there are only two input channels—consider a neuron with two inputs $A_1$ and $A_2$. Let the weights associated with the inputs be set such that (an output firing of) the neuron is triggered whenever $A_1$ fires but not when $A_2$ fires, and assume now that $A_2$ fires usually at the same time as $A_1$. Then, its weights will be strengthened by the Hebbian rule because of the coincidence of $A_1$ and $A_2$. Eventually, the weights of the second channel will have increased sufficiently such that firing of $A_2$ on its own will be sufficient to trigger an output.

Molecular models of Hebbian learning have been proposed before. A biochemical model of associative learning was proposed by Fernando and co-workers. Their model is fully autonomous, but it is also inflexible. Association is learned after just a single coincidence, and hence, the model is unable to detect statistical correlations robustly. Moreover, the system cannot forget the association between the inputs. McGregor et al. introduced an improved design with systems that were found by evolutionary processes. A biochemically more plausible system was proposed by Solé and co-workers, but this system is also limited to learning two coinciding inputs and relies on an explicit operator manipulation in order to forget past associations.

In this article we will propose a fully autonomous chemical artificial neuron, henceforth referred to as CN, Table 1, that goes beyond the state of the art in that it can learn statistical relations between an arbitrary number of inputs. The CN is also able to forget learned associations and as such can adapt to new observations without any intervention by an external observer. Via each of its input channels the CN can accept boli, which is the injection of a certain amount of chemical species, representing the input spikes of simulated neurons. The CN will “learn” the statistical biases of the input boli in the sense that the abundance of some of its constituent species, which play an analogous role to neuronal weights, reflect statistical biases of the boli. In particular, we consider two types of biases. (i) Frequency biases (FB): one or more input channels of the CN receive boli at different rates. (ii) Time correlations (TC): two or more input channels are correlated in time. The TC task can be understood as a direct generalization of associative learning with an arbitrary number of input channels.

We will propose three different versions of the CN. The first (basic) version will be the CN itself, which is a minimal set of chemical reactions. It is also thermodynamically consistent in that it comprises only microreversible reactions with mass-action kinetics. This first version, while compact, assumes a high degree of enzymatic multiplicity which is unlikely to be realizable. Therefore, we shall propose a second version of the model which is not thermodynamically explicit but biologically plausible in the sense that it can be formulated in terms of known biochemical motifs. The main difference between this and the previous system is that the former is compartmentalized. Henceforth, this compartmentalized system will be referred to as c-CN.

We also propose d-CN, a version of the CN that is formulated using DNA strand displacement (DSD), a type of DNA-based computing. DSD is a molecular computing paradigm based entirely on interactions of DNA strands and Watson–Crick complementarity and is biocompatible. By this we mean that DSD computers can, in principle, be injected into organisms and interact with their biochemistry and therefore have potential to be used to control molecular systems. It has been shown that DSD systems are capable of universal computation and indeed that any chemical reaction network can be emulated in DSD. From a practical point of view, it is relatively easy to experimentally realize DSD systems, and their behavior can also be accurately predicted using simulation software such as Visual DSD or Peppercorn. There is now also a wealth of computational methods and tools for designing DNA-based circuits.

Given these properties, there have been a number of attempts to build intelligent DSD systems. Examples include linear-threshold circuits, logic gates, switches, oscillators, and consensus algorithms. There were also some attempts to emulate neural networks in DSD: Qian et al. proposed a Hopfield network which has the ability to complete partially shown patterns. However, because the weights connecting individual neurons were hard coded into the system, the system was unable to learn. Networks of perceptron-like neurons with competitive winner-take-all architectures have also been proposed and show how to use DSD reaction networks to classify patterns, such as MNIST handwritten digits. However, learning is external to these systems; weights have to be determined before building the DNA circuit and are then hard coded into the design.

Supervised learning in DSD was proposed by Lakin and collaborators. They used a two-concentration multiplier circuit motif in order to model the gradient descent weight update rule. However, this approach requires an external observer to provide constant feedback. From the perspective of implementing artificial protointelligence in biochemistry, none of the above approaches can be used as a fully autonomous component of a molecular learning system in the sense that they can operate independently of constant external maintenance.

## RESULTS

In the first part of this section, we describe the microreversible chemical reactions that constitute the CN. Next, we demonstrate that the system of reactions behaves like a spiking neuron, and we analyze the key parameters that determine the performance of the system. In the subsequent section, we describe c-CN, which lends itself more easily to

---

**Table 1. List of Acronyms**

| acronym | definition |
|---------|------------|
| CN      | chemical neuron |
| c-CN    | compartmentalized chemical neuron |
| d-CN    | DNA chemical neuron |
| FB      | frequency biases |
| TC      | time correlations |
| DSD     | DNA strand displacement |

---

2056

https://doi.org/10.1021/acssynbio.1c00625

ACS Synth. Biol. 2022, 11, 2055–2069
experimental implementation. Finally, we discuss how DNA strand displacement can be used to construct the d-CN.

**Chemical Neuron—Minimal Model. Overview.** We model the CN as a set of microreversible elementary chemical reactions obeying mass-action kinetics (Table 2, Figure 1).

Table 2. List of Chemical Reactions Constituting the CN

| function | reaction(s) |
|----------|-------------|
| input    | \( k_{sA} \gamma_A \) \( A_i \rightarrow k_{sA} \gamma_A \) \( A_i \rightarrow B \) |
| activation function | \( B + E \frac{k_{sE}}{k_{sE}} \rightarrow E \) \( \frac{1}{k_{sE}} \) \( i < m - 1 \) \( B + E_{m-1} \rightarrow E \) \( k_{sE} \frac{k_{sE}}{k_{sE}} \) \( \frac{1}{k_{sE}} \) \( H_n + E \) |
| learning  | \( A_i + E \frac{k_{sE}}{k_{sE}} \rightarrow E \) \( \frac{1}{k_{sE}} \) \( \frac{k_{sE}}{k_{sE}} \) \( H_n \rightarrow E \) \( k_{sE} \rightarrow B + H_n \) |
| leak      | \( H_n \rightarrow k_{sE} \) \( \phi \) |

Figure 1. Graphical representation of the minimal model of the CN.

Molecular reversibility makes the model thermodynamically consistent. The system is best understood by thinking of each molecular species \( A_i \) as an input to the system via channel \( i \). The inputs are provided in a form of bolus, which is defined as a fixed amount of molecules introduced into the system at the time of the input. The weight equivalent of the \( i \)th input channel of the CN is the abundance of the species \( H_s \). The species \( E \) is the activated form of \( E \) and plays a double role. It is (i) the learning signal, which indicates that a weight update should take place, and (ii) the output of the system, which could be coupled to further neurons downstream. The internal state of the CN, which acts as a memory for the system, is represented by the abundance of the molecular species \( B \). We now proceed by discussing each reaction in Table 2 in turn.

**Input.** We assume here that the CN has \( N \) different species of input molecules \( A_1, \ldots, A_N \). These represent the \( N \) input channels, each of which is associated with a corresponding weight \( H_1, \ldots, H_N \). The weight molecules are the interpretable output of the neuron in the sense that the abundance of the \( H_i \) molecules will reflect statistical biases in the input. The input is always provided as an exponentially decaying bolus at a particular time \( t_i \) where \( s \) is a label for individual spikes. Concretely, this means that at time \( t = t_i \) the CN is brought into contact with a reservoir consisting of \( \beta \) (unmodeled) precursor molecules \( I_i \) that then decay into \( A_i \) molecules with a rate constant \( k > 0 \). A particular consequence of this is that the \( A_i \) are not added instantaneously but will enter the system over a certain time. This particular procedure is a model choice that has been made for convenience. Different choices are possible and would not impact on the results to be presented. The important point is that the input signal to channel \( i \) is a bolus of quantity \( A_i \) and occurs at a particular time \( t_i \). This enables the system to reach a steady state provided that the input is stationary.

The basic idea of the CN is that input boli \( A_i \) are converted into internal state molecules \( B \). This reaction takes a catalyzed as well as an uncatalyzed form. The uncatalyzed reaction \( A_i \rightarrow B \) is necessary in order to allow the system to learn to react in response to new stimulus, even when the weight associated with a given channel decayed to 0. In the case of the catalyzed reaction, the channel-specific \( H_s \) molecules play the role of the catalyst. Thus, the speed of conversion depends on the amount of weight \( H_s \). If at any one time there is enough of \( B \) in the system then the learning signal \( E \) is created by activating \( E \) molecules. Once the learning signal is present, some of the \( A_i \) are converted into weight molecules, such that the weight of the particular input channel increases. This realizes Hebbian learning in the sense that the coincidence of inputs \( A_i \) and output \( E \) activates weight increases following the well-known Hebbian tenet “What fires together, wires together”.

**Activation Function.** The link between the internal state molecules \( B \) and the learning signal is often called the activation function. In spiking neurons, as they are used in artificial intelligence, this activation is usually a threshold function. The neuron triggers an output if the internal state crosses a threshold value. In chemical realizations, such a threshold function is difficult to realize. Throughout this contribution, our systems are parametrized such that the dynamics of the system is dominated by noise. Molecular abundances are therefore noisy. As a consequence, the activation function has to be seen as the probability to observe the activated form \( E \) as a function of the abundance of \( B \).

An ideal activation function would be a step function, but physical realization will necessarily need to approximate the step function by a continuous function, for example, a sigmoid. In the CN, this is realized as follows. Each of the \( E \) molecules has \( m \) binding sites for the internal state molecules \( B \). Once all \( m \) binding sites are occupied, \( E \) is converted into its active form \( E \). We make the simplifying assumption that the conversion from \( E \) to \( E \) is instantaneous once the last \( B \) binds. Similarly, if a \( B \) molecule unbinds then the \( E \) changes immediately to \( E \). In this model, the balance between \( E \) and \( E \) molecules depends on the binding and unbinding rates of \( B \). We assume that there is a cooperative interaction between the \( B \) molecules such that unbinding of \( B \) from \( E \) is much slower than unbinding from \( E \).

With an appropriate choice of rate constants, this system is known to display ultrasensitivity, i.e., the probability for the fully occupied form of the ligand chain \( (E) \) to exist transitions rapidly from close to 0 to close to 1 as the concentration of ligands approaches a threshold value \( \theta \approx k_i / k_r \). The dynamics of such systems is often approximated by the so-called Hill kinetics. It can be shown that the maximal Hill exponent that can be achieved by such a system is \( m \). This means that the chain-length \( m \), which we henceforth shall refer to as the “nonlinearity”, controls the steepness of the activation function of \( E \). In the limiting case of \( m \rightarrow \infty \), this will be a step function, whereby the probability to observe \( E \) is 0 if the abundance of \( B \) is below a threshold and 1 otherwise. We are limited here to
finite values of \( m > 0 \). In this case, the function is sigmoidal or a saturating function in the case of \( m = 1 \). The parameter \( m \) and hence the steepness of the activation function will turn out to be crucial factors determining the computational properties of the CN.

**Learning.** In neural networks, “learning” is usually associated with the update of weights. Accordingly, in the case of the CN, learning is the change of abundances \( H_i \). The abundance can only increase if two conditions are fulfilled: (i) the learning signal \( E \) is present and (ii) there are still input molecules \( A_i \) in the system. In short, learning can only happen if input and output coincide, which is precisely the idea of Hebbian learning. For an illustrative example of how Hebbian learning works in the CN, see Figure 2.

**Leak.** Finally, we assume that the weight molecules \( H_i \) and the internal state molecules \( B \) decay, albeit at different rates. This is so that the weight abundances can reach a steady state; in addition, it enables the CN to forget past inputs and to adapt when the statistics of the input changes. We will assume that the decay of \( H_i \) is slow compared to the typical rate of input bolii.

Throughout this paper we will assume that the dynamics of \( A, B, \) and \( E \) are fast compared to the change in concentration of \( H \). This is a crucial assumption to allow the weights to capture long-term statistics of inputs; in particular, the weights should not be influenced by high-frequency noise present in the system. Furthermore, we also assume that the lifetime of \( E \) is short. For details of the parameters used, see Table S1.

**Associative Learning.** We first demonstrate that the CN is capable of associative learning (Figure 3). To do this, we generate a CN with \( N = 2 \) input channels. Then, we initialize the CN with a high weight for the first channel \( (H_1 = 100) \) and

Figure 2. (a) Example of three inputs of uniform size received from 3 different channels. Each input shown in the second graph has a different weight associated it: \( H_{green} = 250, H_{blue} = 50, \) and \( H_{red} = 0 \). \( H \) molecules act as a catalyst in the \( A \rightarrow B \) reaction, hence the change in the function of \( B \) molecules over time for each of the inputs. The higher the amount of \( H \), the higher is the peak of \( B \) molecules caused by a particular input. Moreover, with the increase in weights, the function of inputs also changes. The higher the amount of \( H \), the quicker its corresponding \( A \) dissipates.

(b) Example simulation showing the core idea of the CN dynamics. Graphs show the internal state \( B \), learning signal \( E \), and weight \( H \) for a single channel. We assume a bolus provided at time \( t = 0.015 \). This causes the internal state to go up and reach the threshold. Learning signal is triggered at around \( t = 0.03 \), and consequently, the weight is increased by (in this case) 15 molecules of \( H \).

Figure 3. Associative learning in CN. First two graphs show inputs \( A_1 \) and \( A_2 \). Clearly, a single \( A_2 \) does not lead to a sufficient increase of the internal state \( B \), such that no learning signal is triggered. After a few coincidences of \( A_1 \) and \( A_2 \), weights \( H_2 \) (last graph) have increased sufficiently for \( A_2 \) to trigger a signal in its own at time \( t = 0.8 \). Note the increase in weights for the second channel after each coincidence.
a low weight for the second channel ($H_2 = 0$). Furthermore, we set the parameters of the model such that a bolus of $A_1$ is sufficient to trigger an output but a bolus of $A_2$, corresponding to stimulating the second channel, is not. This also means that presenting simultaneously both $A_1$ and $A_2$ triggers a learning signal and increases $H_1$ and $H_2$. If $A_1$ and $A_2$ coincide a few times then the weights of $A_2$ have increased sufficiently so that a bolus of $A_2$ can push the internal state of the system over the threshold on its own. This demonstrates associative learning. Note that unlike some previous molecular models of associative learning (e.g., ref 25), the CN requires several coincidences before it learns the association. It is thus robust against noise.

This means that the CN can also readily unlearn the correlation if input patterns change (see Figures S4 and S5). There are two mechanisms in the system that ensure that the neuron is able to continuously learn new input statistics. These are (i) the decay of the weights, which ensures a rate of forgetting, and (ii) the uncatalyzed reaction $A_j$ to $B$, which allows the system to learn to react in response to new stimulus, even when the weight associated with a given channel decayed to 0.

**Full Hebbian Learning.** We now show that the ability of the CN to learn extends to full Hebbian learning with an arbitrary number of $N$ input channels. First, we consider the FB task, where the CN should detect input channels that fire at a higher frequency than others. To do this, we provide random boli to each of the $N$ input channels. Random here means that the waiting time between two successive boli of $A_i$ is distributed according to an exponential distribution with parameter $1/f_i$, where $f_i$ is the frequency of the input boli to channel $i$. The CN should then detect the difference in frequencies $f_i$ between input channels. We consider the FB task as solved if (after a transient period) the ordering of the abundances of weights reflects the input frequencies, i.e., the number of $H_i$ should be higher than the number of $H_j$ if $f_i > f_j$. Below we will show, using a number of example simulations, that the CN is indeed able to show the desired behavior. Later, we will probe in more detail how the response of the system depends on its parametrization and the strength of the input signal.

In order to test a CN with multiple inputs ($N = 5, m = 1$), we consider 3 variants of the FB task. First, we assume that boli to the first two input channels come at a frequency of 4 Hz, whereas channels 3, 4, and 5 fire at a frequency of 2 Hz; we call this variant FB 2. Similarly, for FB 3 and FB 4, the first 3 and 4 channels, respectively, fire at the higher frequency. Figure 4 shows the steady state weights for each of the three tasks. As expected, in each of the experiments, the weights of the high-frequency inputs are higher when compared to the low-frequency inputs. We conclude that the CN can work as a frequency detector at least for some parametrizations.

The other scenario that we will investigate is the TC task, which is the direct generalization of the associative learning task to an arbitrary number of input channels. For this problem we assume that all input frequencies are the same, i.e., $f_i = f_j$ for all $i, j \leq N$. Instead of differences in frequency, we allow temporal correlations between input boli of some channels. If $A_1$ and $A_2$ are temporally correlated then each bolus of $A_1$ is followed by a bolus of $A_2$ after a time period of $\delta + \xi$, with $\delta$ being a fixed number and $\xi$ a random variable drawn from a normal distribution with $\mu = 0$ and $\sigma^2 = 0.0001$ for each bolus. In all simulations, the input frequency of all channels is set to 2 Hz.

The CN can solve the TC task in the sense that, after a transient period, the weights indicate which channels are correlated. They also indicate the temporal order implied by the correlation, i.e., if $A_i$ tends to precede $A_j$ then the abundance of weight $H_i$ should be lower than the abundance of $H_j$. Furthermore, if $A_i$ is correlated with some other channel $k$ but $A_j$ is not then the abundance of $H_i$ must be greater than that of $H_j$.

In order to test whether the system is indeed able to detect TC biases, we again simulated a CN with $N = 5$ input channels and all weight molecules initialized to $H_i = 0$. We then determined the steady state weights in four different scenarios: there are correlations between (i) $A_1$ and $A_2$ (TC 2), (ii) $A_i$, $A_j$, and $A_3$ (TC 3), and (iii) $A_i$, $A_j$, and $A_4$ (TC 4). The temporal order is always in ascending order of the index, such that in the last example, $A_1$ occurs before $A_2$, which in turn occurs before $A_3$. We find that the behavior of the CN is as expected (Figure 4). At steady state the weights reflect the correlation between input channels, including the temporal ordering, thus allowing us to conclude that, at least for some parametrizations, the CN successfully identify temporal correlations.

**Analysis of Activation Function Nonlinearity.** The ability of the CN to perform in the TC task depends on its ability to detect coincidences. In this section, we will now analyze in more detail how this coincidence detection depends on the nonlinearity of the activation function, i.e., the parameter $m$.
To do this, we consider two extreme cases: First, the case of minimal nonlinearity (i.e., $m = 1$), and second, the limiting (and hypothetical) case of maximal nonlinearity (i.e., $m = \infty$). This latter case would correspond to an activation function that is a step function. While a chemical neuron cannot realize a pure step function, considering the limiting case provides valuable insight.

We consider first this latter scenario with a CN with two inputs $A_1$ and $A_2$. In this case, there will be a learning signal $E$ in the CN if the abundance of $B$ crosses the threshold $\theta$. Let us now assume that the parameters are set such that a single bolus of either $A_1$ or $A_2$ is not sufficient to push the abundance of $B$ over the threshold but a coincidence of both is. In this scenario then we have the following.

- A single bolus of $A_1$ will not lead to a threshold crossing. No learning signal is generated, and weights are not increased.
- If a bolus of $A_1$ coincides with a bolus of $A_2$ then this may lead to a crossing of the threshold of the internal state. A learning signal is generated. Weights for both input channels 1 and 2 are increased (although typically not by equal amounts).

Next, consider an activation function tuned to the opposite extreme, i.e., $m = 1$. It will still be true that both $A_1$ and $A_2$ are required to push the abundance of $B$ across the threshold. However, the learning behavior of the CN will be different.

- A single bolus of $A_1$ will not lead to a threshold crossing. A learning signal may still be generated even below the threshold because the activation function is not a strict step function. The weight $H_1$ will increase by some amount, depending on the bolus size.
- If a bolus of $A_1$ coincides with a bolus of $A_2$ then this will lead to more learning signal being generated than in the case of $A_1$ only. As a result, the weights for both input channels 1 and 2 are increased by more than if they had occurred separately.
These two extreme cases illustrate how the CN integrates over input. In the case of low nonlinearity, the weights of a channel will be a weighted sum over all input events of this channel. The weights will be higher for channels whose bolus coincide often. On the other hand, a step-like activation function will integrate only over those events where the threshold was crossed, thus specifically detect coincidences. From this we can derive two conjectures.

- The higher the nonlinearity, the better the CN at detecting coincidences. Low nonlinearity still allows coincidence detection but in a much weaker form.
- As the bolus size increases, the CN will lose its ability to detect coincidences, especially when the bolus size is so large that a single bolus is sufficient to push the abundance of $B$ over the threshold. In this case, a single input spike can saturate the activation function, thus undermining the ability of the system to detect coincidences effectively.

In order to check these conjectures, we simulated a version of the CN with 3 inputs, where $A_1$ and $A_2$ are correlated and $A_3$ fires at twice the frequency of $A_1$ and $A_2$. We considered the minimally nonlinear case ($m = 1$) and a moderate nonlinearity ($m = 4$), which shows the weights as a function of the bolus size (Figure 5). The minimal nonlinear CN detects both coincidences and frequency differences but loses its ability to detect coincidences as the bolus size increases. This is consistent with the above formulated hypothesis. In contrast, for the nonlinear CN and moderately low bolus sizes, the weights indicate the coincidences strongly (i.e., the weights $H_2$ are highest) and less so the FB. As the bolus size increases, the nonlinear CN loses its ability to detect coincidences and becomes a frequency detector, as conjectured.

Next, we check how the coincidence detection depends on the time delay between the correlated signals. To do this, we created a scenario where we provided two bolus to the system. The first bolus $A_1$ comes at a fixed time and the second one a fixed time period $\delta$ thereafter. We then vary the length of $\delta$ and record the accumulation of weights $H_2$ as a fraction of the total weight accumulation. Figure 6 shows the average weight accumulation per spike event. It confirms that the CN with low nonlinearity is less sensitive to short coincidences than the CN with higher $m$. However, it can detect coincidences over a wider range of lag durations. This means that for higher nonlinearities, the differential weight update becomes more specific but also more limited in its ability to detect coincidences that are far apart. In the particular case of $\delta > 0.1$, the CN with $m = 1$ does not detect any coincidences any more whereas the case of $m = 1$ shows some differential weight update throughout.

Next, we tested the conjecture that the TC can be solved more effectively by the CN when the nonlinearity is higher. To do this, we generated a CN with $N = 5$ input channels on the TC 2 task. We then trained the CN for nonlinearities $m = 1, \ldots, 10$. As a measure of the ability of the system to distinguish the weights, we used the index of dispersion, i.e., the standard deviation divided by the mean of the weights. A higher index of dispersion indicates more heterogeneity of the weights and hence a better ability of the system to discriminate between the biased and the unbiased input channels.

Consistent with our hypothesis, we found that the ability to distinguish temporarily correlated inputs increases with the nonlinearity. However, it does so only up to a point (the optimal nonlinearity), beyond which the index of dispersion reduces again (Figure 7). Increasing the bolus size, i.e., increasing the number of $A_i$ that are contained within a single bolus, shifts the optimal nonlinearity to the right. This suggests that the decline in the performance of the CN for higher chain lengths is due to a resource starvation. The realization of the sigmoidal function, i.e., the thresholding reactions in Table 2, withdraws $m$ molecules of $B$ from the system. As a consequence, the CN is no longer able to represent its internal state efficiently and the activation function is distorted. If the total abundance of $B$ is high compared to $E$ then this effect is negligible. We conclude that there is a resource cost associated with computing nonlinearity. The higher $m$, the higher the bolus size required to faithfully realize the activation function. As an aside, we note that other designs for the system are also possible. For example, $B$ molecules could be used catalytically. Nevertheless, such systems would also face different trade-offs. The system presented here was one of many designs that we tested and provided the most desirable properties for learning temporal patterns.

While the TC task requires nonlinearity, the FB task does not. This can be understood acknowledging that the FB task is fundamentally about integrating over input, which can be done naturally in chemical systems. Indeed, it can be done by systems that are much simpler than the CN. For example, the minimal system to detect FB bias is $A_i \xrightarrow{d} \phi$. For appropriately chosen values of $d$, the steady state value of $A_i$ would then reflect the input frequency. To understand this, note that the
input frequency determines the rate of increase of $A_i$. This rate divided by the decay rate constant $d$ then determines the steady state abundance of $A_i$ such that $A_i$ trivially records its own frequency. This system is the minimal and ideal frequency detector.

The CN itself is not an ideal frequency detector because all weight updates are mediated by the internal state $B$. Hence, the weights are always convolutions over all inputs. The weights thus reflect both frequency bias and temporal correlations. In many applications this may be desired, but sometimes it may not be. We now consider the conditions necessary to turn the CN into a pure frequency detector, i.e., a system that indicates only FB but not TC. One possibility is to set the parameters such that the CN approximates the minimal system. This could be achieved by setting $k_B^{AB} \ll k_B^{BA}$ and all other rate constants very high in comparison to $k_B^{BA}$. The second possibility is to tune the CN such that a single bolus saturates the threshold. In this case, the strength of the learning signal does not depend on the number of boli that are active at any one time. A single bolus will trigger the maximal learning signal. This is confirmed by Figure 5, which shows that as the bolus size increases, the system becomes increasingly unable to detect temporal correlations but remains sensitive to frequency differences.

**c-CN: CN with Compartments.** The CN, as presented in Table 2, is thermodynamically plausible and has the benefit of being easy to simulate and analyze. However, it is biologically implausible. As written in Table 2, the molecular species $A_i$, $H_j$, and $B$ would have to be interpreted as conformations of the same molecule with different energy levels. In addition, we require that these different conformations have specific enzymatic properties. Molecules with the required properties are not known currently, and it is unlikely that they will be discovered or engineered in the near future.

As we will show now, it is possible to reinterpret the reaction network that constitutes the CN (Table 2) so as to get a model whose elements are easily recognizable as common biochemical motifs. This requires only relatively minor adjustments of the reactions themselves but a fundamental reinterpretation of what the reactions mean.

The main difference we introduce is that the new model is compartmentalized (Figure 8). While in the basic model the indices of $A_i$ and $H_j$ referred to different species that exist in the same volume, it should now be interpreted as the same species but living in different compartments. This means that $A_i$ and $A_j$ are the same type of molecule but located in compartments $i$ and $j$, respectively. Similarly, $H_i$ and $H_j$ are the same species. All compartments $i$ and $j$ are themselves enveloped in a further compartment (the “extracellular space”). The internal state species $B$ is the same as $A_i$ but located in the extracellular space. From here on, we will refer to this reinterpreted model as the c-CN. It is formally described by the reactions in Table 3.

Input to channel $i$ is provided by boli of the molecular species $A$ into the compartment $i$. A novelty of c-CN when compared to CN is that it has an activated form of $A_i$, denoted by $A_i^*$. The conversion from $A$ to $A_i^*$ is catalyzed by the learning signal $E$. Also new is that each compartment contains a gene $h$ that codes for the molecule $H$ (we suppress the index indicating the compartment). Expression of the gene is activated by $A_i^*$ binding to the promoter site of $h$. We also allow a low leak expression by the inactivated gene (denoted as $h_0$ in Table 3). Gene activation of this type is frequently modeled using Michaelis–Menten kinetics, thus reproducing in good approximation the corresponding enzyme kinetics in the CN. The molecules of type $H$ are now transporters for $A$. We then interpret the conversion of $A_i$ to $B$ as export of $A$ from compartment $i$ to the extracellular space. The rate of export of $A$ is specific to each compartment in that it depends on the abundance of $H$ in this compartment. Finally, we interpret the $E$ molecules as transmembrane proteins that are embedded in the membrane of each compartment. Their extracellular part has $m$ binding sites for $B$ molecules which bind cooperatively. When all sites are occupied, the intracellular part is activated, i.e., becomes $E$. In its activated form it can convert $A$ to $A_i^*$. Another difference between the two versions of the models is that the molecule $E$ is now specific to each membrane. The minimum number of copies of $E$ is thus $N_i$, whereas in the basic model a single copy of $E$ at time $t = 0$ could be sufficient. This

![Graphical representation of a c-CN.](https://doi.org/10.1021/acssynbio.1c00625)
Table 3. List of Chemical Reactions Constituting the c-CN*β

| function             | reaction(s)                                                                 |
|----------------------|-----------------------------------------------------------------------------|
| input                | \( k_{in} \underset{h_{in}}{\rightarrow} A \)                          |
|                      | \( A + H \underset{h_{in}}{\rightarrow} AH \underset{h_{in}}{\rightarrow} B + H \) |
| activation function  | \( B + E \underset{h_{in}}{\rightarrow} E_{in}, \ i \leq m \) − 1          |
| weight accumulation  | \( E + A \underset{h_{in}}{\rightarrow} EA \underset{h_{in}}{\rightarrow} E + A^* \) |
|                      | \( A^* \underset{h_{in}}{\rightarrow} A \)                              |
|                      | \( h_{in} \underset{h_{in}}{\rightarrow} H_{in} + h_{in} \)            |
| leak                 | \( H \underset{h_{in}}{\rightarrow} \phi \)                            |
|                      | \( B \underset{h_{in}}{\rightarrow} \phi \)                           |

*Molecular species \( A, E, E_i, h_{in}, h, \) and \( H \) are compartmentalized. Each compartment has a gene \( h \), which when activated by \( A^* \) can express a transporter \( H \).

has two consequences. First, at any particular time the number of occupied binding sites will typically be different across the different \( N \) compartments. This is a source of additional variability. Moreover, since the number of copies of \( E \) is higher than that in CN, the c-CN is more susceptible to starvation of \( B \) as a result of the extracellular binding sites withdrawing molecules from the outer compartment. Both of these potential problems can be overcome by tuning the model such that the abundance of \( B \) molecules is high in comparison to \( E \) molecules.

This highlights that the differences between the basic CN and c-CN are deeper than the list of reaction suggests. Our simulations, however, confirm that the c-CN supports associative learning (Figure 9) and full Hebbian learning (Figure 10) just as the basic CN provided that the parameters are set appropriately.

**d-CN: Chemical Neuron in DNA.** We now show how to emulate the chemical reaction network of Table 2 using DNA strand displacement (DSD).28 This is interesting because the experimental realization of DSD systems is straightforward and predictable when compared to biochemical reaction networks.

The basic idea of DNA-based computation is that double-stranded DNA molecules with an overhang on one strand—often called the toehold—can interact with single-stranded DNA that contains the Watson–Crick complement of the toehold via partial or total displacement of the existing complement. DNA-based systems are typically analyzed on two levels: the sequence level and domain level. The former involves the study of interactions between individual nucleotide pairs, while the latter focuses on the interactions between domains. Here, domains are sequences of nucleotides of varied length. There are two types of domains which are differentiated by their length. Short domains or toeholds are between 4 and 10 nucleotides and are assumed to be able to bind and unbind from complementary strands. Long domains, or recognition domains, are at least 20 nucleotides in length and assumed to bind irreversibly. DSD is a domain-level mechanism for performing computational tasks with DNA via two basic operations: toehold-mediated branch migration and strand displacement.

**Implementing the d-CN Using Two-Domain DSD.** In order to emulate the chemical neuron in DNA, we will focus here on two-domain strand displacement.32,36 where each molecular species comprises a toehold and a long domain only. These species can interact with double-stranded gates which facilitate the computation. Restricting computation to two-domain strands helps to protect against unexpected interactions between single-stranded species, which can occur with more complex molecules. Also, as all double-stranded structures are stable and can only change once a single-stranded component has bound, there is no possibility for gate complexes to polymerize and interact with each other.

Here, we will be using the standard syntax of the Visual DSD programming language28 to describe the species present in our system. We denote double-stranded molecules as \([r]x\), where its upper strand \( <r \) is connected to a complementary lower strand \( ^x \). Each of the reactants and products in our system is an upper single-stranded molecule composed of a short toehold domain (annotated with a prefix \( t \) and an identifier \( ^t \)) and a corresponding long domain \( ^<t \) \( x \). We will refer to a short domain of a two-domain DSD strand \( A_n \) as \( t a \) and its corresponding long domain as \( a n \), where \( n \) is a channel index. Note that the toehold is not specific to the species index \( n \), and therefore, the recognition of each input and weight strand is dependent on their long domains rather than their toeholds. We will use the same convention for all other channel-specific

![Figure 9](https://doi.org/10.1021/acssynbio.1c00625)

Same as Figure 3 but for c-CN. For the parameters used, see Table S2. For this experiment, we approximated the ligand kinetics by a Hill function in order to speed up the simulations.
two-domain species. For a detailed description of the nucleotide structure and binding rates, see Tables S3 and S4 in the SI. The main two-domain strands that enable communication between different modules of the d-CN are shown in Table 5.

While there is a theoretical guarantee that any chemical reaction network can be mimicked by a DSD circuit, it is often difficult to find circuits. However, there are now a number of general design motifs with known behaviors in the literature. Here, we will make extensive use of the two-domain scheme, which introduces a Join–Fork motif to mimic a chemical reaction. While the abstract chemical system remains broadly similar to the CN model, there are some crucial differences (see Table 4). The general strategy we take to convert the CN to DSD is to translate each of the catalytic reactions in Table 4 into a Join–Fork gate.32–36 Subsequently, we will simulate the gates acting in concert.

We first explain how we use the Join–Fork gates. For each reaction, a Join gate is able to bind the reactants and produces a translator strand. Then, the translator activates a Fork gate, which in turn releases the reaction products. Additional energy must be supplied to completely release all products from Fork gates, as the translator strand will only displace the first product. Appropriately designed helper strands are therefore placed in the solution to release subsequent products. After the first product has unbound, an exposed toehold is left, which can lead to unwanted side effects. To address this, we follow32 and extend the original design from ref 36 by incorporating an additional long domain on the left-hand side of the Fork gate, which upon binding an appropriate auxiliary molecule seals the gate to prevent rebinding of its outputs. Here, we extend all Join gates in an equivalent way to prevent rebinding of the translator strand. This addition allows us to avoid interactions of the double-stranded complexes with waste molecules.

In our design, binding of the translator immediately releases an $A_\text{r}(<\text{ta}^\text{an}>)$ strand, the first of the reaction products. The second product, $B(<\text{tb}^\text{b}>)$, is released upon binding of a Fork helper strand $<\text{tb}^\text{an}>$. Finally, the Fork$_{AB}$ gate is sealed upon binding of the Fork seal strand $<\text{i tb}^\text{an}>$. The pair of Join and Fork gates together consume 1 molecule for each of the reactants and produce 1 molecule for each of the products, ensuring equivalent stoichiometry to the abstract reaction.

In order to illustrate the mapping from the CN to DSD, we describe now in detail the reaction $F_{si_\text{A}} = A_\text{r} \rightarrow A_\text{r} + B$ (Figure 11a), which serves as a representative of all 3 catalytic reactions in the d-CN. A Join$_{Fsi_\text{A}}$ gate is defined by a structure that enables the binding of $F_{si_\text{A}}$ and $A_\text{r}$; the gate is only active if both input species are present. First, $F_{si_\text{A}}$ binds and displaces the incumbent bound $<\text{in ta}^\text{an}>$ molecule, exposing the $\text{ta}^\text{an}$ toehold. This enables the binding of $A_\text{r}(<\text{ta}^\text{an}>)$, which then displaces the $<\text{an tisi}^\text{an}>$ translator strand, signaling that the reactants have been received and that the overall reaction can fire. The Join$_{Fsi_\text{A}}$ gate is then sealed by the binding of $<\text{tisi}^\text{i} >$, preventing rebinding of the translator and producing a further waste molecule $<\text{i}$. The Fork$_{AB}$ gate is designed in such a way that upon triggering by the translator strand of the corresponding Join gate it is able to release both product molecules.

Controlling the Activation Function Nonlinearity with Extended Polymers. The only reaction which takes a different form than a combination of Join and Fork gates is the activation function. We first describe the simplest case of an activation function with minimal nonlinearity, i.e., $m = 1$. In this case it takes the form $\{\text{tb}^\text{b}\} = [\text{tb}^\text{b}] <\text{b}>$ or graphically: $\frac{1}{m} \frac{m - 1}{m}$ $\frac{m - 2}{m}$ $\frac{m - 3}{m}$. B molecules can bind to this compound; in doing so they expose the $\text{tb}^\text{b}$ short domain which allows for binding of $E_\text{b}$. When $E_\text{b}$ binds to the complex, it displaces a long domain $<\text{b}>$ and releases the learning signal $E_\text{c}$, which in the case of $m = 1$ is represented by three-domain species $<\text{b tel}^\text{b}>$.

This system can now be generalized to arbitrary integer values of $m$ by extending the polymer with additional segments to accommodate for binding of more $B$ and $E_\text{b}$ molecules (Figure 12). We use segments of the form $[\text{b tel}^\text{k}]$, where $k$ is the index of the $k$th extra segment in the complex. Each new segment should be added before the last

Table 4. List of Reactions That Constitute the d-CN

| function               | reaction                  |
|------------------------|----------------------------|
| signal integration     | $F_{si_\text{A}} = A_\text{r} + B$ |
| weight accumulation    | $A_\text{r} + E = E + H_\text{r}$ |
| signal modulation      | $A_\text{r} + H_\text{r} \equiv H_\text{r} + B$ |
| activation function    | $B + E_\text{b} \equiv E_\text{i}$ |
|                        | $E + E_{i-1} \equiv E$ |

Table 5. List of Key DNA Strands Which Facilitate Learning

| name            | signal          | DSD species |
|-----------------|-----------------|-------------|
| input           | $A_\text{r}$    | $<\text{ta}^\text{an}>$ |
| weights         | $H_\text{r}$    | $<\text{th}^\text{hn}>$ |
| internal state  | $B$             | $<\text{tb}^\text{b}>$ |
| learning signal | $E$             | $<\text{tb}^\text{an}>$ |
| signal integration fuel | $F_{si_\text{A}}$ | $<\text{tisi}^\text{fsin}>$ |

Figure 10. Same as Figure 4 but for c-CN. Experiments approximated the ligand dynamics by a Hill function in order to speed up the simulations.
fragment which contains $E$: \([b \text{ tel}^1] <b>\). In the case of \(m = 2\), the activation function then is \((tb^* b) [b \text{ te}0^*] : [b \text{ te}^*] : [b \text{ te}2^*] <b>\) or graphically: \([b \text{ te}0^*] : [b \text{ te}^*] : [b \text{ te}2^*] <b>\).

The weight accumulation function is distinguished from standard gates in that the first reactant of the Join gate, i.e., \(<b \text{ te}2^* b>\) representing the learning signal \(E\) and the first product of the Fork gate are both three-domain species. The initial form of the Fork gate complex has a long domain \(b\) branching out of the double-stranded structure (Figure 13). This modification is necessary in order to allow for \(E\) to catalyze the reaction.

An alternative way to implement this mechanism could be the use of a multistep cascade of gates. This approach, however, would necessitate the use of additional toehold definitions, thus limiting the number of input channels that could be simulated.

**Computational Complexity.** Extending the d-CN to accommodate additional input channels requires the user to define a single new toehold domain definition \(\text{tiwan}\), which is responsible for weight accumulation in each of the \(N\) channels. Moreover, there are six toehold domains that remain the same regardless of the number of input channels (\(ta, th, tb, tfsi, tism, tisi\)). Therefore, the system with \(N = 3\) input channels requires 9 toehold definitions (6 + \(N\)). In addition, depending on the length of the polymer which facilitates the activation function there are at least two additional toehold domains: \(\text{te}0\) and \(\text{te}1\). We base the recognition of the inputs as well as other two-domain strands in the system on the long domains. There are two long domains which remain the same regardless of the number of channels (\(h, l\)) and three which need to be defined when adding another input channel (\(an, hn, fsin\)). Therefore, the system with \(N = 3\) input channels requires 11 long domain definitions (2 + 3\(N\)).

**Simulating the d-CN.** When simulating the d-CN, we initialize the system with different amounts of gate complexes and helper strands needed for the computation by both Join and Fork gates depending on their function. Signal modulation fuel molecules are initiated at 25 000 \(\mu M\), signal integration at

### Figure 11
Mapping the CRN neuron to a DNA neuron. We use a two-domain Join—Fork gate to emulate each of the catalytic reactions in the CN (Table 4). In each case, a Join gate binds the two reactants in sequence, first displacing a waste molecule and second displacing a translator molecule, which triggers the corresponding Fork gate to release strands representing the reaction products. Translator replaces the first product, and then a Fork helper displaces the second product. Both Join and Fork gates can be sealed upon binding of an appropriate auxiliary strand (labeled Join seal and Fork seal), which displaces the final incumbent bound \(<i>\) strand.

### Figure 12
Activation function for \(m = 2\) is modeled as a long polymer which accommodates for binding of \(B\) and subsequent \(E\) molecules to its surface. These two species can bind to the polymer in an alternate manner. First, the binding of \(B\) frees up a \(\text{te}0\) toehold; next, the binding of \(E\) frees up a \(tb\) toehold, etc. Altogether, this process consumes \(B\) molecules. At the end of the process, a three-domain learning signal molecule \(E\) is produced. In the case of \(m = 2\), this molecule takes the following form: \(<b \text{ te}2^* b>\). This mechanism can also run backward to produce \(B\) molecules.

### Figure 13
Weight accumulation \((m = 2): A_n + E \rightarrow E + H_n\). Catalytic reaction that realizes the weight accumulation function is a Join gate and a modified variant of the Fork gate. In this variation, the first reactant of the Join gate and the first product of the Fork gate are a three-domain species \(E\), which represents the learning signal. Initial Fork gate complex now has a long domain \(b\) branching out of the double-stranded structure. This modification is needed to ensure complementarity with the tunable activation function.
50 000 \mu M, and weight accumulation at 10 000 \mu M. We also initialize the fuel molecules necessary for the signal integration mechanism $F_{si}$ with 50 000 \mu M. Lastly, in all of the experiments, we choose to set the bolus size, i.e., the amount of $A_n$ species injected to the system at each spike, to $\beta = 10 \mu M$. In order to model decay of $H_n$ species, we introduce garbage collection molecules $\{\text{th}^*\}[hn]$, which sequester and inactivate the molecular species $H_n$. We inject 12 and 0.1 \mu M of these species to the system periodically every 1000 s.

We have been careful to use strand displacement reaction rates that are within the range that has been measured experimentally.\(^{47}\) In order to reproduce the desired dynamical behaviors, the binding rates associated with the ta, th toeholds have been set to lower values than the other toeholds; see Table S3 for details on the parameters.

To determine whether the d-CN is capable of learning, we carried out a range of simulations using Visual DSD, Figure 14. We found that both in finite and detailed mode compilation could produce the intended dynamical behaviors. Similarly, we

![Figure 14](https://pubs.acs.org/doi/10.1021/acssynbio.1c00625)  
**Figure 14.** Examples of learning episodes in d-CN for (a) frequency bias task and (b) temporal correlation task. For statistical data about the weight distributions obtained over multiple runs, see Figure S2.

![Figure 15](https://pubs.acs.org/doi/10.1021/acssynbio.1c00625)  
**Figure 15.** Normalized steady state weights as a function of the length of the activation function polymer. As the polymer is extended, the activation function becomes steeper and therefore requires a correlation of at least two signals to trigger learning. Therefore, the DNA neuron becomes better at recognizing temporal correlations when $m$ is high.

![Figure 16](https://pubs.acs.org/doi/10.1021/acssynbio.1c00625)  
**Figure 16.** Same as Figure 4 but for the d-CN. Data was only collected after the weights reached the steady state (after 800 000 time units).
found that these behaviors could be produced in both simulations at low copy numbers (using Gillespie’s stochastic simulation algorithm) and in the fluid limit (deterministic rate equations). Accordingly, we show infinite mode deterministic simulations in the main article and other simulations in the S1 (Figure S3).

To check whether the d-CN behaves as expected, we test its ability to distinguish the two types of biases on tasks where 2 is temporally correlated with 3 and further analyze how this depends on the nonlinearity/polymer length (Figure 15). First, we consider a scenario where 3 is both uncorrelated with 3/ 3 and has a spiking frequency twice as high as the other input channels (0.0002 Hz; Figure 15a). Consistent with the CN, the d-CN is sensitive to frequency bias when the nonlinearity is low, corresponding to the weights of 3 being high for m = 1. Vice versa, in the case of high nonlinearity, the d-CN recognizes the temporal correlations, corresponding to the weights of 3 and 3 being high. When removing the frequency bias of 3, the system still differentiates between uncorrelated and correlated inputs but the ability to distinguish the two types of signals increases with m (see Figure 15b).

We also compared the ability of the d-CN directly with the CN. We found that the d-CN is able to detect both FB and TC biases (Figure 16). However, in the TC task the indication of the temporal order of the input signals is subtle in the sense that the steady state weights of the correlated channels are almost the same with only a slight difference indicating temporal order.

**DISCUSSION**

To the best of our knowledge, the CN is the first fully autonomous chemical model of a Hebbian spiking neuron. While it is unlikely that the basic model can be engineered as is, it has some features that make the system interesting from a fundamental point of view.

One of the attractive features of the (basic) CN neuron is that it is micoreversible and therefore thermodynamically plausible. This makes it a useful theoretical tool to probe the thermodynamics of learning. While a thorough analysis of the energy requirements of the system is beyond the scope of this article, we note that the physical plausibility of the model has highlighted resource requirements of computation. In particular, we found that increasing the nonlinearity comes at an additional cost in resources. The CN suffers from starvation of B molecules as m increases. For a sufficiently high number of m, this leads to a breakdown of the mechanisms and the system loses its ability to detect coincidences, as illustrated in Figure 7. This “starvation” effect can be alleviated by increasing the bolus size (while keeping the threshold fixed; Figure 7). In a biological context, the increase of the bolus size comes at a direct synthesis cost if the molecules that make up the bolus need to be made by the cell. Yet, even if we assume that the particles are, somehow, pre-existing, injecting a bolus requires chemical work, which is proportional to the number of particles, i.e., the bolus size. Hence, there is a fundamental thermodynamic cost involved in computing the nonlinearity. We are not aware of any formal proofs that show that computing nonlinearities necessarily requires an increased energy requirement. It therefore remains an open question whether or not this is a feature of the particular model choices or the manifestation of a deeper constraint. Interestingly, the FB task, which does not rely on nonlinearities, can be solved with much simpler and thermodynamically cheaper designs, e.g., a simple decaying particle.

While the basic CN does not lend itself to a direct implementation in biochemistry, we presented a compartmentalized interpretation of the system that is biologically more plausible. It interprets different input species and indeed the internal state molecule B as one and the same species but contained in different compartments. This makes the system feasible, in principle. Although creating many compartments with the required dynamics may remain challenging, significant progress has been made in recent years toward programming molecular systems in protocells.

Interestingly, there are structural similarities between the c-CN and the lac system in E. coli. The essence of the lac system is that it only switches on the lactose metabolism (the equivalent to the weight molecules in the compartment) when it is stimulated by lactose in the environment (i.e., B). The principle of operation of the lac system is similar to that of the c-CN, except that E. coli does of course not export lactose to the environment. Taking this analogy seriously, it would be interesting to consider whether catabolite repression, which is a moderately complex decision process, can be mapped to a simple neural network.

Among the three versions of the chemical neuron that we presented, we found that all could reproduce the same qualitative behaviors (Figures 3 and S9). However, given that all three of them are different designs, each version required its own parametrization, which had to be found by manual exploration in each case. It is thus not possible to reproduce the behavior of one model with another one exactly. Qualitatively, however, we found the same behaviors in all models. The only major difference was on the TC task. Unlike the other two versions of the chemical neuron, the d-CN did not clearly highlight the temporal order of input signals (Figure 16). While the d-CN indicates a strong difference between the correlated and the noncorrelated species, the weight difference between the correlated channels which should indicate the temporal order is marginal. Whether this can be improved with a better parametrization or whether this points to a fundamental limitation of the model must remain an open question.

From an engineering perspective, the d-CN is certainly the easiest to realize experimentally. DNA circuits are much less prone to crosstalk than more standard biochemical reaction networks. Synthesizing DNA molecules is now a routine procedure. There are, however, several elements of the d-CN design that will require careful consideration before an implementation can be done. For any practical use, one would need to interface the DNA computer with the in vivo target systems. How to do this in a general way remains an open question, but there have been a number of previous systems that indicate possible pathways.

More specifically, for the d-CN, there are a number of experimental challenges that need to be addressed. In order to ensure that the kinetics of the d-CN are conserved throughout the learning and testing phases, we require the activation species B to decay. To remove the B species, we employ simple helper complexes, which are periodically replenished during the simulation. These complexes are capable of making B and H, species unreactive, thereby removing them from the system. In order to achieve better reproducibility of the results, the experimental realization of this would necessitate a relatively frequent or continuous supply of these DNA complexes. While
difficult to achieve experimentally, there are known techniques to overcome the need for frequent replenishment, including the use of buffered gates\textsuperscript{40} or timer circuits.\textsuperscript{35}

Scaling the system to more input channels requires additional short domain sequences per new channel. Primafac the scaling up of the d-CN is therefore limited by the availability of orthogonal short domain sequences. A redesign based on localized design principles could be a feasible solution if the number of toeholds becomes a problem. Here, instead of using a different set of long and short domains, distinct channels could be implemented through physical separation of the species.\textsuperscript{35}

Describing the model as a neuron encourages the question of building networks capable of complex computational tasks. A major impediment for building networks of d-CN could be the immediate injection of A species to the neurons in the next layers of the network. This would necessitate inclusion of a different activation function or a mechanism which would allow for signal propagation. Incorporating a buffered gate design\textsuperscript{40} could allow for a programmed release of a certain number of input species once the activation signal is produced. Nevertheless, we leave the question of constructing functional neural networks in DNA for future research.

## ASSOCIATED CONTENT

### Supporting Information
The Supporting Information is available free of charge at https://pubs.acs.org/10.1021/acssynbio.1c00625.

Detailed description of the reaction rate constants necessary to realize CN and c-CN\textsubscript{2}; lists of nucleotide sequences and binding rates for d-CN\textsubscript{2}; weight distributions for FB 2 and TC 2 tasks as a function of m; statistical data about the weight distributions for d-CN\textsubscript{2}; examples of d-CN training in other simulation modes and relearning of input statistics; study of signal modulation mechanism in the d-CN; examination of strategies for garbage collection and stability of the learnt solutions as a function of bolus size and abundance of gate molecules; analysis of performance of the CN and d-CN models on the FB and TC task, and Visual DSD code for d-CN (PDF)

## AUTHOR INFORMATION

### Corresponding Author

Jakub Fil — APT Group, School of Computer Science, The University of Manchester, Manchester M13 9PL, United Kingdom; orcid.org/0000-0002-4044-8390; Email: jakub.fil@manchester.ac.uk

### Authors

Neil Dalchau — Microsoft Research, Cambridge CB1 2FB, United Kingdom; orcid.org/0000-0002-4872-6914
Dominique Chu — CEMS, School of Computing, University of Kent, Canterbury CT2 7NF, United Kingdom

Complete contact information is available at: https://pubs.acs.org/10.1021/acssynbio.1c00625

### Author Contributions

Conceived the research: D.C., N.D., J.F. Conducted the research: J.F. Wrote the paper: D.C., N.D., J.F.

### Notes
The authors declare no competing financial interest.

## REFERENCES

1. Govern, C.; ten Wolde, P. Energy dissipation and noise correlations in biochemical sensing. Phys. Rev. Lett. 2014, 113, 258102.
2. Govern, C.; ten Wolde, P. Optimal resource allocation in cellular sensing systems. Proc. Natl. Acad. Sci. U.S.A. 2014, 111, 17486−17491.
3. Alon, U. An Introduction to Systems Biology: Design Principles of Biological Circuits; Chapman & Hall/CRC Computational Biology Series; CRC Press LLC., 2019.
4. Yi, T.; Huang, Y.; Simon, M.; Doyle, J. Robust perfect adaptation in bacterial chemotaxis through integral feedback control. Proc. Natl. Acad. Sci. U. S. A. 2000, 97, 4649−4653.
5. Hoffer, S. M.; Westerhoff, H. V.; Hellingwerf, K. J.; Postma, P. W.; Tommassen, J. Autoamplification of a two-component regulatory system results in “learning” behavior. Journal of bacteriology 2001, 183, 4914−4917.
6. Chu, D. Performance limits and trade-offs in entropy-driven biochemical computers. J. Theor. Biol. 2018, 445, 1−9.
7. Chu, D. Limited by sensing - A minimal stochastic model of the lag-phase during diauxic growth. J. Theor. Biol. 2017, 414, 137−146.
8. Chu, D.; Barnes, D. The lag-phase during diauxic growth is a trade-off between fast adaptation and high growth rate. Sci. Rep. 2016, 6, 25191.
9. Aggarwal, C. Neural networks and deep learning: a textbook; Springer: Cham, Switzerland, 2018.
10. Sengupta, A.; Ye, Y.; Wang, R.; Liu, C.; Roy, K. Going Deeper in Spiking Neural Networks: VGG and Residual Architectures. Front. Neurosci. 2019, 13, 95.
11. Afshar, S.; Ralph, N.; Xu, Y.; Schai, J.; Cohen, A. v.; Tapson, G. Event-Based Feature Extraction Using Adaptive Selection Thresholds. Sensors 2022, 20 (6), 1600.
12. Fil, J.; Chu, D. Minimal Spiking Neuron for Solving Multilabel Classification Tasks. Neural Computation 2020, 32, 1408−1429.
13. Gütt, R. Spiking neurons can discover predictive features by aggregate-label learning. Science 2016, 351, aa4113.
14. Oja, E. Simplified neuron model as a principal component analyzer. Journal of Mathematical Biology 1982, 15, 267−273.
15. Diehl, P.; Cook, M. Unsupervised learning of digit recognition using spike-timing-dependent plasticity. Front. Comput. Neurosci. 2015, 9, 99.
16. Afshar, S.; Hamilton, R. J.; Tapson, J.; van Schaik, A.; Cohen, G. Investigation of Event-Based Surfaces for High-Speed Detection, Unsupervised Feature Extraction, and Object Recognition. Frontiers in Neuroscience 2019, 12, 1047.
17. Okamoto, M.; Sakai, T.; Hayashi, K. Biochemical Switching Device Realizing McCulloch-Pitts Type Equation. Biol. Cybern. 1988, 58, 295−299.
18. Hjelmfelt, A.; Weinberger, E. D.; Ross, J. Chemical implementation of neural networks and Turing machines. Proc. Natl. Acad. Sci. U. S. A. 1991, 88, 10983−10987.
19. Banda, P.; Teuscher, C.; Stefanovic, D. Training an asymmetric signal perceptron through reinforcement in an artificial chemistry. Journal of The Royal Society Interface 2014, 11, 20131100.
20. Blount, D.; Banda, P.; Teuscher, C.; Stefanovic, D. Feedforward Chemical Neural Network: An In Silico Chemical System That Learns xor. Artificial Life 2017, 23, 295−317.
21. Shirakawa, T.; Sato, H. Construction of a Molecular Learning Network. Journal of Advanced Computational Intelligence and Intelligent Informatics 2013, 17, 913−918.
22. Nesbath, D.; Zaikin, A.; Sakai, Y.; Romano, M.; Giuraniuc, C.; Kanakov, O.; Laptveva, T. Synthetic biology routes to bio-artificial intelligence. Essays in Biochemistry 2016, 60, 381−391.
23. Chen, M.; Xu, J. Construction of a genetic conditional learning system in Escherichia coli. Sci. China Inf. Sci. 2015, 58, 1−6.
24. Racovita, A.; Jaramillo, A. Reinforcement learning in synthetic gene circuits. Biochem. Soc. Trans. 2020, 48, 1637−1643.
25. Fernando, C.; Liekens, A.; Bingle, L.; Beck, C.; Lenser, T.; Stelk, D.; Rowe, J. Molecular circuits for associative learning in
single-celled organisms. Journal of The Royal Society Interface 2009, 6, 463–469.

(26) McGregor, S.; Vasas, V.; Husbands, P.; Fernando, C. Evolution of Associative Learning in Chemical Networks. PLOS Computational Biology 2012, 8, e1002739.

(27) Macia, J.; Vidiella, B.; Solé, R. Synthetic associative learning in engineered multicellular consortia. Journal of The Royal Society Interface 2017, 14, 20170158.

(28) Lakin, M.; Youssef, S.; Polo, F.; Emmott, S.; Phillips, A. Visual DSD: a design and analysis tool for DNA strand displacement systems. Bioinformatics (Oxford, England) 2011, 27, 3211–3213.

(29) Amir, Y.; Ben-Ishay, E.; Levner, D.; Ittah, S.; Abu-Horowitz, A.; Bachelet, I. Universal computing by DNA origami robots in a living animal. Nature Nanotechnol. 2014, 9, 353–357.

(30) Seelig, G.; Soloveichik, D.; Zhang, D.; Winfree, E. Enzyme-Free Nucleic AcidLogic Circuits. Science 2006, 314, 1585–1588.

(31) Soloveichik, D.; Seelig, G.; Winfree, E. DNA as a universal substrate for chemical kinetics. Proc. Natl. Acad. Sci. U. S. A. 2010, 107, 5393–5398.

(32) Chen, Y.; Dalchau, N.; Srinivas, N.; Phillips, A.; Cardelli, L.; Soloveichik, D.; Seelig, G. Programmable chemical controllers made from DNA. Nat. Nanotechnol. 2013, 8, 755–762.

(33) Yurke, B.; Turberfield, A.; Mills, A.; Simmel, F.; Neumann, J. A DNA-fuelled molecular machine made of DNA. Nature 2000, 406, 605–608.

(34) Fontana, W. Pulling Strings. Science 2006, 314, 1552–1553.

(35) Badelt, S.; Grun, C.; Sarma, K. V.; Wolfe, B.; Shin, S. W.; Winfree, E. A domain-level DNA strand displacement reaction enumerator allowing arbitrary non-pseudoknotted secondary structures. Journal of The Royal Society Interface 2020, 17, 20190866.

(36) Cardelli, L. Two-Domain DNA Strand Displacement. Proceedings Sixth Workshop on Developments in Computational Models: Causality, Computation, and Physics, Edinburgh, Scotland, September 10–10 July 2010; DCM, 2010; pp 47–61.

(37) Mullor Ruiz, I.; Arbona, J.-M.; Lad, A.; Mendoza, O.; Aime, J.-P.; Elezgaray, J. Connecting localized DNA strand displacement reactions. Nanoscale 2015, 7, 12970–12978.

(38) Qian, L.; Winfree, E. Scaling up digital circuit computation with DNA strand displacement cascades. Science 2011, 332, 1196–1201.

(39) Dalchau, N.; Szép, G.; Hernansaz-Ballesteros, R.; Barnes, C. P.; Cardelli, L.; Phillips, A.; Csikasz-Nagy, A. Computing with biological switches and clocks. Natural Computing 2018, 17, 761–779.

(40) Lakin, M.; Youssef, S.; Cardelli, L.; Phillips, A. Abstractions for DNA circuit design. Journal of The Royal Society Interface 2012, 9, 470–486.

(41) Qian, L.; Winfree, E.; Bruck, J. Neural network computation with DNA strand displacement cascades. Nature 2011, 475, 368–372.

(42) Genot, A.; Fujii, T.; Rondelez, Y. Scaling down DNA circuits with competitive neural networks. J. R. Soc. Interface 2013, 10, 20130212.

(43) Cherry, K. M.; Qian, L. Scaling up molecular pattern recognition with DNA-based winner-take-all neural networks. Nature 2018, 559, 370–376.

(44) LeCun, Y.; Cortes, C. MNIST handwritten digit database, 2010; http://yann.lecun.com/exdb/mnist/.

(45) Lakin, M.; Stefanovic, D. Supervised Learning in Adaptive DNA Strand Displacement Networks. ACS Synth. Biol. 2016, 5, 885–897.

(46) Chu, D.; Zabet, N.; Mitavskiy, B. Models of transcription factor binding: Sensitivity of activation functions to model assumptions. J. Theor. Biol. 2009, 257, 419–429.

(47) Zhang, D.; Winfree, E. Control of DNA Strand Displacement Kinetics Using Toehold Exchange. J. Am. Chem. Soc. 2009, 131, 17303–17314.

(48) Lyu, Y.; Peng, R.; Liu, H.; Kuai, H.; Mo, L.; Han, D.; Li, J.; Tan, W. Protocells programmed through artificial reaction networks. Chemical Science 2020, 11, 631–642.

(49) Chu, D.; Barnes, T. The lag-phase during diauxic growth is a trade-off between fast adaptation and high growth rate. Sci. Rep. 2016, 6, 25191.

(50) Groves, B.; Chen, Y.; Zurla, C.; Pochekaiylov, S.; Kirschman, J.; Santangelo, P.; Seelig, G. Computing in mammalian cells with nucleic acid strand exchange. Nat. Nanotechnol. 2016, 11, 287–294.

(51) Oesinghaus, L.; Simmel, F. Switching the activity of Cas12a using guide RNA strand displacement circuits. Nat. Commun. 2019, 10, 2092.

(52) Jung, C.; Ellington, A. D. Diagnostic Applications of Nucleic Acid Circuits. Acc. Chem. Res. 2014, 47, 1825–1835.

(53) Douglas, S. M.; Bachelet, I.; Church, G. M. A Logic-Gated Nanorobot for Targeted Transport of Molecular Payloads. Science 2012, 335, 831–834.

(54) Fern, J.; Scalice, D.; Cangialosi, A.; Howie, D.; Potters, L.; Schultman, R. DNA Strand-Displacement Timer Circuits. ACS Synthetic. ACS Synth. Biol. 2017, 6, 190–193.

(55) Chatterjee, G.; Dalchau, N.; Muscat, R.; Phillips, A.; Seelig, G. A spatially localized architecture for fast and modular. Nat. Nanotechnol. 2017, 12, 920.