**PAPER • OPEN ACCESS**

# Associative memories using complex-valued Hopfield networks based on spin-torque oscillator arrays

To cite this article: Nitin Prasad *et al* 2022 *Neuromorph. Comput. Eng.* **2** 034003

View the article online for updates and enhancements.

# NEUROMORPHIC
## Computing and Engineering

**PAPER**

# Associative memories using complex-valued Hopfield networks based on spin-torque oscillator arrays

Nitin Prasad[1,2,*] ⓘ, Prashansa Mukim[1,2] ⓘ, Advait Madhavan[1,3] ⓘ and Mark D Stiles[4] ⓘ

1   Associate, Physical Measurement Laboratory, National Institute of Standards and Technology, Gaithersburg, MD, United States of America
2   Department of Chemistry and Biochemistry, University of Maryland, College Park, MD, United States of America
3   The Institute for Research in Electronics and Applied Physics, University of Maryland, College Park, MD, United States of America
4   Physical Measurement Laboratory, National Institute of Standards and Technology, Gaithersburg, MD, United States of America
*   Author to whom any correspondence should be addressed.

**E-mail:** nitin.prasad@nist.gov

## Abstract

Simulations of complex-valued Hopfield networks based on spin-torque oscillators can recover phase-encoded images. Sequences of memristor-augmented inverters provide tunable delay elements that implement complex weights by phase shifting the oscillatory output of the oscillators. Pseudo-inverse training suffices to store at least 12 images in a set of 192 oscillators, representing $16 \times 12$ pixel images. The energy required to recover an image depends on the desired error level. For the oscillators and circuitry considered here, 5% root mean square deviations from the ideal image require approximately 5 $\mu$s and consume roughly 130 nJ. Simulations show that the network functions well when the resonant frequency of the oscillators can be tuned to have a fractional spread less than $10^{-3}$, depending on the strength of the feedback.

## 1. Introduction

The need to realize energy-efficient, high-density, and high-speed computing is driven by the explosive emergence of big data [1]. One problem with scaling traditional processors to meet this need is that they require shuttling data between memory and the computational processors, a problem referred to as the von Neumann bottleneck. Inspired by biological neural networks, in-memory computing has advanced as an approach to minimize this bottleneck in the last decade [2, 3]. Novel nanoscale devices offer unique pathways to complement existing complimentary metal-oxide-semiconductor (CMOS) technologies in creating artificial neural networks [4–6] in which memory is embedded in the computational engines.

Neural activity in the brain has inspired a variety of ways to encode information to make processing it more efficient. Mimicking spiking behavior has led to spike-based neural networks that encode and process information based on the strength, rate, and timing of spiking signals [7, 8]. Similarly, spontaneous neuronal oscillations observed in collections of biological neurons have inspired oscillatory neural computers that encode and process information as relative frequencies and phases [9, 10]. The parallels between associative learning observed in the brain [11] and spontaneous synchronization observed in weakly coupled oscillators [12] has led to proposals for artificial associative memories based on synchronized oscillators [9, 13].

One approach to synchronized-oscillator-based artificial associative memories is to physically realize binary Hopfield networks [9, 14]. Traditional Hopfield networks are *auto-associative* because they can retrieve learned data when presented with a defective or partial piece of data. These networks are recurrent, all-to-all connected, artificial neural networks in which the neurons are binary threshold units [15]. The neuron outputs are determined by applying thresholds to the sum of the incoming weighted feedback connections from all other neurons. Information is encoded in the real-valued weights forming these feedback connections. When presented with an incomplete or a noisy version of one of the stored vectors encoded in these weights, the Hopfield network retrieves the corresponding stored vector by performing energy minimization on the functional defined by the network training. Such networks address the von Neumann bottleneck when the weights

are stored in proximity to the neurons. In oscillator-based binary Hopfield networks, the information to be processed is encoded as the relative phases of the oscillators (in-phase or out-of-phase) and the memory used in the processing is stored in the synaptic weights.

There have been several advances in the computational power of Hopfield networks since their original proposal [15]. These include advances in training algorithms [16] and energy functionals [17, 18] that allow for greater information storage [19]. Binary Hopfield networks have also been extended to store and retrieve continuous- or quasi-continuous-valued information, such as grayscale images [20–22]. This is achieved by using multilevel real activations [20, 21] or by using complex-valued signum activations [22]. As we demonstrate in this work, the latter approach can be mapped to a network of weakly coupled oscillators.

Physically realizing oscillators can be done with CMOS technology alone [23], but such oscillators are not particularly compact. Alternative oscillators for associative memories can be fabricated from emerging nanoscale devices such as spintronic oscillators [13], vanadium-dioxide oscillators [24, 25] and opto-electronic oscillators [26, 27]. In this work, we focus on auto-associative memories built using particular spintronic oscillators, spin-torque oscillators realized with magnetic tunnel junctions, although many of the arguments presented here are general and can easily be extended to other oscillator-based systems.

Spin-torque oscillators based on magnetic tunnel junctions (MTJs) consist of a stack of two ferromagnetic layers separated by a thin tunnel barrier insulator [28]. Two effects make MTJs useful as the foundation for making oscillators. The first is that due to the tunneling magnetoresistance effect, the device resistance depends on the relative orientation of the magnetizations in the two magnetic layers so that as the magnetization precesses, the resistance and hence either current through or voltage across the MTJ oscillates. The resistance varies roughly as $R(\theta) = R_{\mathrm{P}} + (R_{\mathrm{AP}} - R_{\mathrm{P}})(1 - \cos\theta)/2$, where $\theta$ is the angle between the magnetization directions, $R_{\mathrm{P}}$ is the low resistance found for the parallel alignment, and $R_{\mathrm{AP}}$ is the larger value for antiparallel alignment. The second effect that makes MTJs useful as oscillators is that the magnetization can be dynamically excited by passing a current through the tunnel junction. The carriers tunneling through the insulator are spin-polarized by one layer and perturb the magnetic orientation in the other layer through a process called spin-transfer torque [29]. In most applications, the magnetization of one of the ferromagnetic layers is *pinned* and that of the other, *free* layer responds to the current passing through the MTJ. In magnetic random-access memory [30], spin transfer torques due to current pulses are used to switch the magnetization. In spintronic oscillators [31], a DC current excites gigahertz dynamical oscillations that can be detected by the resulting time-varying resistance, which for MTJs is due to the tunneling magnetoresistance.

For memory applications, tunnel junctions are formed so that the parallel and antiparallel configurations of the free layer are energetically favored. Any perturbations of these states give rise to oscillations in the magnetization of the free layer, which decay after the external perturbations are removed [32]. A spin-torque oscillator can be realized by tailoring the geometry of an MTJ in a way that sustained microwave-frequency oscillations can be induced in the MTJ when the damping force is balanced by the bias current-induced spin-transfer torque [33]. When external small current perturbations are applied alongside the bias currents, the spin-torque oscillators can lock to these perturbations, provided that the driving frequencies are sufficiently close to the natural frequencies of the spin-torque oscillators [34, 35]. In this paper, we consider the well-studied vortex-based spin-torque oscillator [36]. Such oscillators have been used in other neuromorphic computing implementations such as reservoir computing [6], synchronizing an array to external sources for speech formant recognition [37], and synchronizing an array to external sources to implement synaptic weights [38]. An array of related oscillators based on metallic nanocontacts have been synchronized for norm computation [39].

Computing with oscillators typically involves coupling them together with controllable interactions. There is significant research into physically coupling spintronic oscillators through magnetostatic or exchange interactions [31, 40, 41], which could be compact and energy efficient. However, experiments are just beginning to demonstrate control [42] over the oscillators and their coupling. Applications of coupled spin-torque oscillators so far have used electrical coupling [37, 39], taking advantage of tunneling magnetoresistance and spin-transfer torques. The goal of the present work is to present an application based on electrically coupled oscillators and energy-efficient electrical circuits that enable this approach. These circuits allow control of both the amplitude and the phase of the interaction.

In addition to the spin-torque oscillator, efficient implementations of Hopfield networks require local memory. We implement such memory with memristors, which are programmable two-terminal nonvolatile resistors with history-dependent resistances [43]. Memristors have been made from several classes of materials including transition metal oxides like $VO_x$ [44], $TiO_x$ [45], and $HfO_x$ [46]; perovskites [47] and chalcogenides [48]. Different mechanisms are responsible for resistive switching behavior in different materials and are described in [43]. Their appeal is in their potential ability to address the von Neumann bottleneck by providing dense non-volatile local memory. When used as digital cross-point memories, memristors have been integrated

in the back-end-of-the-line in various fabrication facilities [49, 50]. In an analog context, memristors are commonly used in a crossbar array where the resistances encode the elements of a matrix to enable matrix-vector multiplication [51, 52], the workhorse operation for machine learning acceleration. Here, local memory is used to store the control currents for oscillators and the delay and scale parameters in the implementation of the synapses.

We model a continuous-time Hopfield network using specific models for these two emerging technologies. We implement the neurons with spin-torque oscillators based on CoFeB/MgO/CoFeB MTJs [36] and use generic resistive memristor models [53] to program the delays and scaling in the feedback network. In section 2, we describe a schematic structure that implements such a network with spin-torque oscillators functioning as neurons and memristors providing the local control of the delay and scale circuitry that operate as weights in the feedback network. We simulate a network with $N = 192$ neurons with $192 \times 191$ complex weights (self-feedback is omitted), which are used to store twelve $16 \times 12$-pixel images. The simulations reported in section 3 demonstrate retrieval of these images from distorted versions in 5 $\mu$s while consuming 130 nJ of energy. Section 4 details the CMOS circuitry that implements this network for a particular spin-torque oscillator model. Memristors, in which the resistance can be varied, store the delay times and the scaling of the complex weights. For the system size considered here, the implementation of the complex weights consumes comparable energy to the spin-torque oscillators. For larger systems, we expect the energy implementing the complex weights to dominate because the number of weights scales as $N^2$ relative to the oscillators and the amplifiers, which scale as $N$. The details of the model used for the oscillators in the network simulations is described in appendix A. In section 5, we discuss some of the limits of our zero-temperature model for the spin-torque oscillators, the idealized memristor models, and the progress that is needed in device fabrication to fully realize this implementation of a Hopfield network.
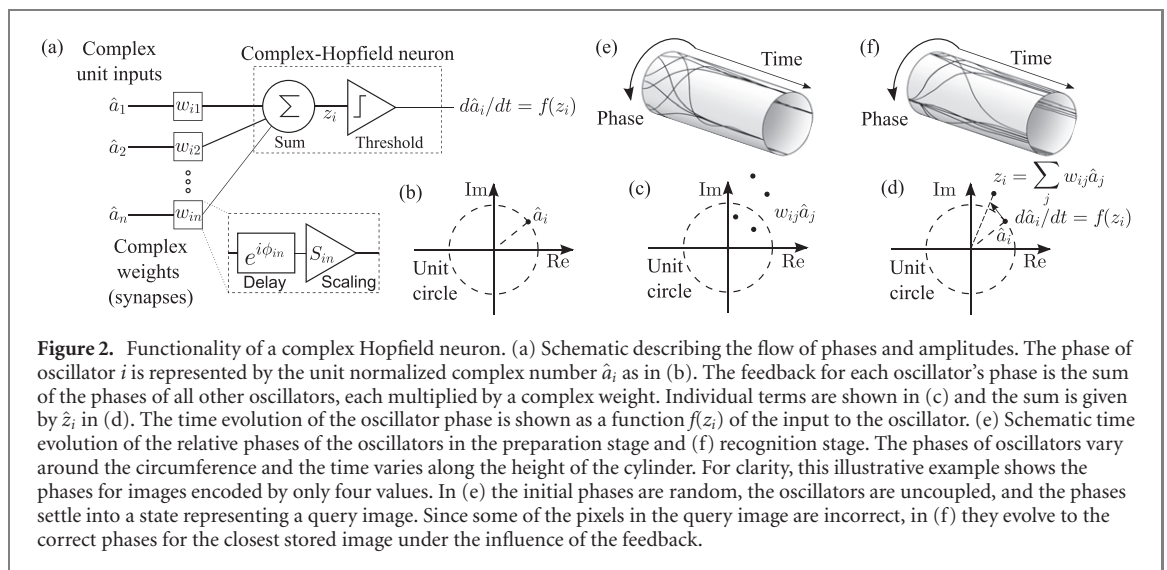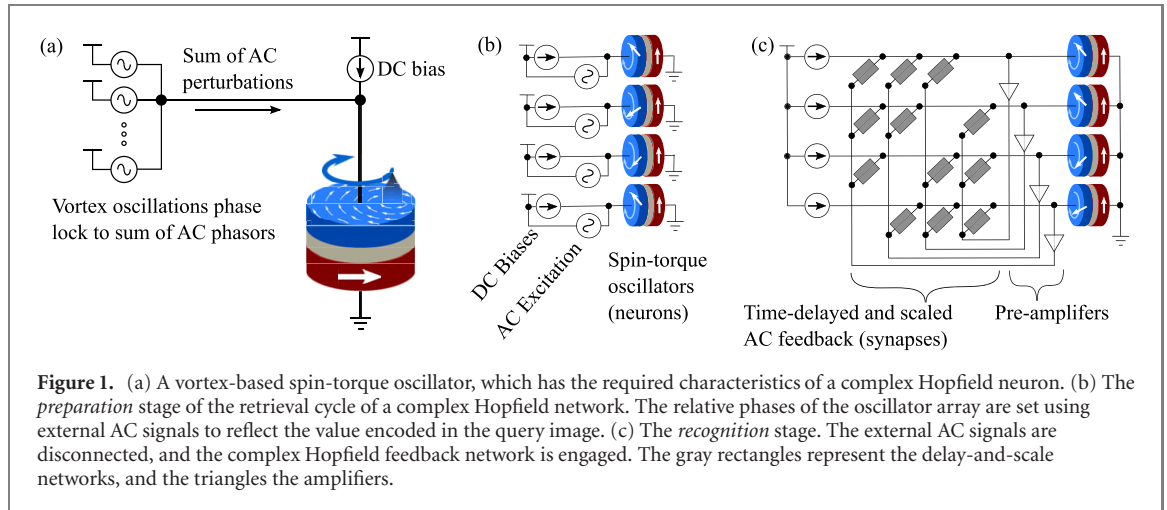
## 2. Spin-torque-oscillator-based complex Hopfield networks

The Hopfield network proposed in reference [22] achieves multivalued information storage by replacing the binary state of each neuron and real-valued weights used in the original Hopfield network [15] with continuous unit-circle complex states for each neuron and complex-valued weights. In this complex Hopfield network [22], time evolution occurs asynchronously and in discrete time steps, so that the state of each node is maintained on the complex unit circle by choosing the activation functions to be complex-valued signum functions, $\text{sgn}(z) = z/|z|$.

Here, we implement a complex Hopfield network with a physical model of frequency-synchronized coupled oscillators, specifically vortex-based spin-torque oscillators, as depicted in figure 1(a). These oscillators are well characterized [36] and have been used in several experimental implementations of neuromorphic computing [6, 37]. As in the approach of reference [22], the frequencies of the oscillators are tuned to be close enough that the information is encoded in the relative phases of the oscillators. The absolute phase does not play a role. However, unlike reference [22], both the phase and time evolution are continuous because we are modeling the physical behavior of the oscillators and the electronic circuits. That is, the evolution of the network is described by differential equations rather than discrete steps in time. During the time evolution of the oscillators, their phases advance or retard in a continuous manner due to coupling to the other oscillators. Their instantaneous outputs are passed through the complex weights which scale and introduce additional delays to the oscillator outputs. The outputs of the complex weights are then fed back through small AC currents, which together with the nonlinearity of the oscillators determine the time evolution of their phases. The phases evolve because the feedback causes small changes in amplitude that shift the frequency due to the nonlinearity, in turn shifting the phase. We neglect the small changes in the amplitudes except for their effect on the frequencies and phases. Eventually, each oscillator synchronizes to the AC input from the other oscillators with a modified phase.

Training a coupled-oscillator-based complex Hopfield network to store a set of vectors requires setting the complex weights in the feedback loop. Then, the trained network is used to recover stored images from distorted versions of those images. We start by describing the retrieval of stored images using an already trained complex Hopfield oscillator network and return to a discussion of training at the end of this section. The retrieval procedure consists of two stages, a *preparation* stage, which uses the circuit schematically illustrated in figure 1(b), and a *recognition* stage, which uses the circuit schematically illustrated in figure 1(c). In the preparation stage, the oscillators are uncoupled and driven by external AC signals that individually set the phase of each oscillator. The relative phases of these input signals encode the distorted query vector to be presented to the network. In the recognition phase, the outputs of the oscillators are fed back to the other oscillators to adjust their relative phases to one of the stored images.
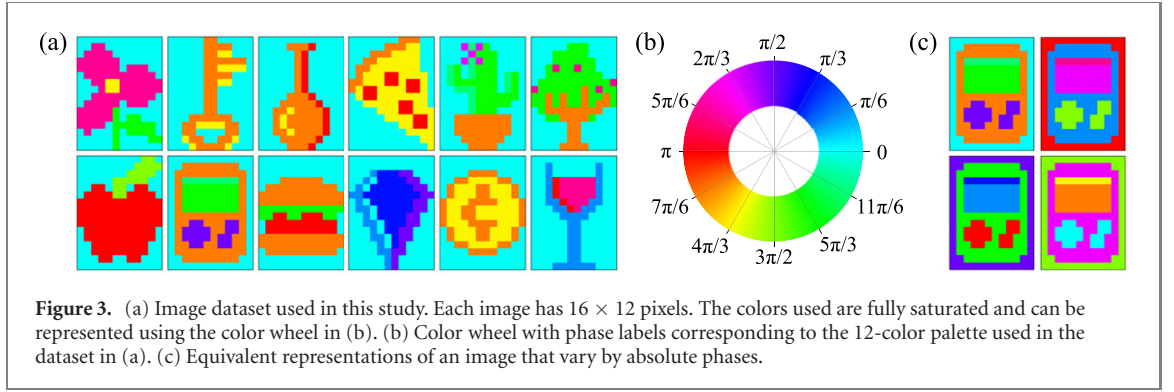
Ignoring small changes in the amplitude of the oscillations, the voltage oscillations of the $i$th oscillator can be parameterized by a unit complex number $\hat{a}_i$, as shown in figure 2(b). The phase of the AC voltage oscillation corresponds to the angle of $\hat{a}_i$ with $|\hat{a}_i| = 1$ set by the oscillation amplitude. A schematic of the time evolution

**Figure 1.** (a) A vortex-based spin-torque oscillator, which has the required characteristics of a complex Hopfield neuron. (b) The *preparation* stage of the retrieval cycle of a complex Hopfield network. The relative phases of the oscillator array are set using external AC signals to reflect the value encoded in the query image. (c) The *recognition* stage. The external AC signals are disconnected, and the complex Hopfield feedback network is engaged. The gray rectangles represent the delay-and-scale networks, and the triangles the amplifiers.



**Figure 2.** Functionality of a complex Hopfield neuron. (a) Schematic describing the flow of phases and amplitudes. The phase of oscillator $i$ is represented by the unit normalized complex number $\hat{a}_i$ as in (b). The feedback for each oscillator's phase is the sum of the phases of all other oscillators, each multiplied by a complex weight. Individual terms are shown in (c) and the sum is given by $\hat{z}_i$ in (d). The time evolution of the oscillator phase is shown as a function $f(z_i)$ of the input to the oscillator. (e) Schematic time evolution of the relative phases of the oscillators in the preparation stage and (f) recognition stage. The phases of oscillators vary around the circumference and the time varies along the height of the cylinder. For clarity, this illustrative example shows the phases for images encoded by only four values. In (e) the initial phases are random, the oscillators are uncoupled, and the phases settle into a state representing a query image. Since some of the pixels in the query image are incorrect, in (f) they evolve to the correct phases for the closest stored image under the influence of the feedback.

of the relative phases of the spin-torque oscillators as they phase lock to the input signals during the preparation stage is depicted in figure 2(e). The phases are plotted on a cylindrical manifold with the angle on the circular axis representing the phase of an oscillator and the dimension along the length of the cylinder representing time. Each solid black line on the cylinders in figure 2(e) represents the time evolution of the phase of one of the oscillators as it starts from a random phase and stabilizes into one of the four states in the query image. For illustrative purposes in this example, the nominal information is encoded by relative phases of 0, $\pi/2$, $\pi$ and $3\pi/2$. The stored images in this example have these discrete phases in each of the pixels. The noisy query image also has these values but some of them are incorrect with respect to the stored image.

After each oscillator individually phase locks to its input signal with a phase encoding the trial image, the external AC drives are turned off while the feedback circuit is turned on to start the recognition stage. The feedback circuit used in the recognition phase, implemented by delay-and-scaling networks, is schematically illustrated in figure 1(c) and conceptually in figure 2(a). The feedback to each oscillator from the other oscillators is an AC current determined by the phases of the AC voltage outputs of the other oscillators and the appropriate pairwise weights. When a mixture of AC perturbations is applied to an oscillator close to its natural frequency, the oscillator output evolves toward phase-locking to the sum of these AC perturbations. The output voltage of the $j$th oscillator in the network, represented by $\hat{a}_j$, is multiplied by a set of complex-valued synaptic weights $w_{ij}$, which connect the output of the $j$th oscillator to the input of all other oscillators. The resulting products for all $j \neq i$, represented by dots in figure 2(c), are fed into the $i$th complex Hopfield neuron which first adds up all the incoming weighted inputs to compute $z_i = \sum_j w_{ij}\hat{a}_j$. Information is encoded only in the phases of the oscillators and not in their amplitudes. However, as is the case in any Hopfield network, both phases and amplitudes of the weights in the feedback network are important as they capture the interference between the many stored vectors.

**Figure 3.** (a) Image dataset used in this study. Each image has $16 \times 12$ pixels. The colors used are fully saturated and can be represented using the color wheel in (b). (b) Color wheel with phase labels corresponding to the 12-color palette used in the dataset in (a). (c) Equivalent representations of an image that vary by absolute phases.

The feedback causes the phases of the oscillators to adjust relative to each other and to settle into the relative phases corresponding to the stored image that is closest to the noisy input image. The oscillator state $\hat{a}_i$ evolves continuously towards the instantaneous values of $\mathrm{sgn}(z_i)$ with a rate $\mathrm{d}\hat{a}_i/\mathrm{d}t$ dependent on $\hat{a}_i$ and $z_i$, as depicted in figure 2(d). The evolution of $\hat{a}_i$ does not have a simple functional form as discussed in appendix A but is determined by the equation of motion of the oscillator. Finally, in steady state, $\hat{a}_i = \mathrm{sgn}(z_i)$ for all $i$. A schematic illustration of this evolution is given in figure 2(f). Here, the phases each start from the chosen discrete value in the query image and evolve toward the closest stored image. After allowing sufficient time for the network to stabilize, the relative phases settle to a steady state that encodes the recalled vector. Pixels that start with the wrong value change significantly as they approach the correct values and pixels that start with the correct values are pulled slightly away from those values before they settle back into the correct values.

The feedback circuit used in the recognition phase is shown schematically in figure 1(c). The feedback path consists of a preamplifier, used to boost the weak microwave output voltages of the spin-torque oscillators, and complex-valued weight elements. The DC biases across each oscillator have been tuned using memristors so that each of the oscillators has the same frequency. Because all the oscillators in the network run at the same frequency, the complex weight elements can be implemented using a time-delay network followed by an amplifier network as in figure 2(a). Both the delays and scalings are programmed by tuning the resistances of memristors, see section 4. The outputs of these weight elements inject small-signal AC currents (small relative to the DC current) into the oscillators that get added to the constant DC bias currents.

We demonstrate the two-stage retrieval process by simulating an array of $N = 192$ vortex-based spin-torque oscillators. The complex weight elements encode $K = 12$ images, each with $16 \times 12$ pixels, shown in figure 3(a). These images are fully saturated in color and the color of each pixel corresponds to one of the twelve discretized phases that are equivalent to twelve particular states of the color wheel in figure 3(b). The chosen discrete phases are indicated on the color wheel. The circular nature of the color wheel allows us to naturally map the colors to the periodic phases of the oscillator. Although such a dataset with periodic pixel values is naturally suited to be coded on a complex Hopfield network, demonstrations of linear grey-scale image learning and retrieval have also been demonstrated on software-based complex Hopfield networks [22, 54]. In such linear scale mappings, however, an additional error may be introduced during the retrieval process if the extreme values of the linear scale are mapped to adjacent discretized phases. Note that in the phase encoding used here, only the relative phases of the oscillators encode information. An absolute phase added to all oscillators does not affect the dynamics. For example, all four images in figure 3(c), which differ from each other by an absolute phase, represent the same image.

The complex-valued weights $w_{ij}$ are set so that when the network of oscillators is connected in the feedback configuration, the original vectors stored on the oscillator array correspond to fixed points of the oscillator dynamics. A wide selection of learning rules is available. Non-iterative rules such as Hebbian [15] and pseudo-inverse [55] learning rules provide single-shot offline training methods when provided with the entire dataset. On the other hand, iterative training rules such as contrastive divergence [56] and Storkey [16] learning rules provide iterative methods of learning starting from an initial guess solution. Here, we first use a pseudo-inverse learning rule for offline training, which can store correlated vectors more efficiently than the Hebbian rule [55] and explore iterative training rules at the end of section 3. The pseudo-inverse weights $w_{ij}$ are given by

$$w_{ij} = \frac{1}{N}\sum_{k,l=1}^{K} \hat{x}_i^k [C^{-1}]^{kl} \left(\hat{x}_j^l\right)^* \quad \text{and} \quad C^{kl} = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{x}_i^k\right)^* \hat{x}_i^l. \tag{1}$$

Here, $\hat{x}_i^k$ are unit complex-numbers encoding the $i$th pixel of the $k$th image to be stored on the oscillator array. Reference [55] showed that altering the values of the diagonal elements of the weight matrix $w_{ii}$ does not affect the fixed points of the oscillator dynamics. Reducing the magnitude [57] of the diagonal weights, $w_{ii}$, or setting

them to zero [55] have been proposed as ways to improve convergence towards fixed points. We choose to set $w_{ii} = 0$ and avoid self-feedback.

When the oscillator network is connected in the feedback configuration using the weights, $w_{ij}$, the phases, described by $\hat{a}_i$, continuously evolve toward the steady-state phase-locked condition, as shown in figure 2(d). As they evolve, the network stabilizes into a local minimum of the energy $E = -\sum_{i,j=1}^{N} \hat{a}_i^* w_{ij} \hat{a}_j$.

The complex-valued weights can be implemented using a delay-and-scale network. The details of a complex weight implementation using CMOS circuits augmented with memristors to store the weights locally are described in section 4. In section 3, we assume that these are ideal delay and scale elements. We capture the non-linear dynamics of a vortex-based spin torque oscillator using the model described in appendix A. Using this oscillator model along with the ideal delay elements, we perform system-level simulations, described in section 3, to study the two stages of the inference cycle in an offline-trained network that stores the images in figure 3(a).

## 3. Network simulations

We simulate the performance of the network described in section 2 assuming ideal performance of the feedback network for the synapses and the behavior of the spin-torque oscillators described in appendix A. The total current injected into each oscillator $j$ is a sum of the DC bias current $I_j^{\mathrm{DC}}$ and the AC synaptic feedback current $I_j^{\mathrm{AC}}$. $I_j^{\mathrm{DC}}$ sets the nominal oscillation frequency of the gyrotropic mode of the oscillator, chosen here to be 248 MHz by setting $I_j^{\mathrm{DC}} = 80\ \mu$A. On the other hand, $I_j^{\mathrm{AC}}$ is a sum of synaptic currents $I_{ij}^{\mathrm{Syn}}$ from the feedback network that feed into the $j$th oscillator. The total AC feedback into the neuron $j$ is given by

$$I_j^{\mathrm{AC}} = \sum_{i=1}^{N} I_{ij}^{\mathrm{Syn}} = \sum_{i=1}^{N} \kappa |w_{ij}| \sin\left[2\pi f t + \arg(\hat{a}_i) + \arg(w_{ij})\right]. \tag{2}$$

Here $\kappa$ is a coupling constant that controls the strength of the injected feedback current.

Figures 4(a) and (b) show representative snapshots at various labeled time intervals during the two stages of the retrieval cycle, preparation and recognition. In figure 4(a), the oscillators are prepared to represent the query image. The color of each pixel represents the phase of the oscillator representing that pixel. The query image stored on the oscillator array is distorted using a discrete Gaussian noise generator, which distorts each pixel by one state on average from its nominal value. In the recognition phase of figure 4(b), the prepared image state evolves in the presence of the feedback network. The oscillator network relaxes to the stored image of figure 3(a) that the noisy version of the image most resembles.

Similarly, when presented with an incomplete image, the network can retrieve the complete image. The recognition stage of such a retrieval cycle is shown in figure 4(c). The oscillator array is prepared to represent an image with half the pixels representing an original image from the dataset in figure 3(a), while the other half of the pixels are independently assigned one of the twelve allowed values from a uniform distribution of the color wheel. While the coupling to the disordered pixels pulls some of the correct pixels away from the correct value, taken as a whole, the set of pixels continuously evolves toward the correct image.

We characterize the error between a retrieved image and the $k$th stored image as

$$\Delta^k = \min_{\phi} \sqrt{\frac{1}{N}\sum_{j=1}^{N} \left(\arg\left\{\exp[\mathrm{i}(\phi_j^k - \phi_j + \phi)]\right\}\right)^2}, \tag{3}$$

where $\phi_j$ is the phase of the $j$th pixel in the retrieved image and $\phi_j^k$ is the corresponding phase in the stored image. This complicated expression is simply describing the root mean square difference in the phases between the stored and retrieved images. The complications arise from the facts that the relative phases must be between $-\pi$ and $\pi$ and that the overall relative phase does not matter as shown in figure 3(c). The argument of the exponential function of a complex argument keeps the phase difference in the appropriate range and the minimization over $\phi$ shifts the relative phases to get the best agreement.

Figure 4(d) shows the evolution in the error between a set of distorted images and the correct images averaged over 30 distorted versions of each of the twelve images shown in figure 3(a). At time $t = 0\ \mu$s, the oscillators are prepared to be in one of the distorted images using a Gaussian noise generator on each image of the dataset that distorts each pixel on average by one discrete phase level, $2\pi/12$, from its nominal value. The average error decreases as the oscillator dynamics brings the oscillators into one of the fixed points representing the correct stored image. Each color represents a different coupling constant $\kappa$. Stronger coupling constants lead to a quicker initial decrease in the error, but this improved rate of convergence largely saturates for values of $\kappa$ greater than 10 nA. We note that the value of $\kappa$ is the upper bound on the feedback associated with
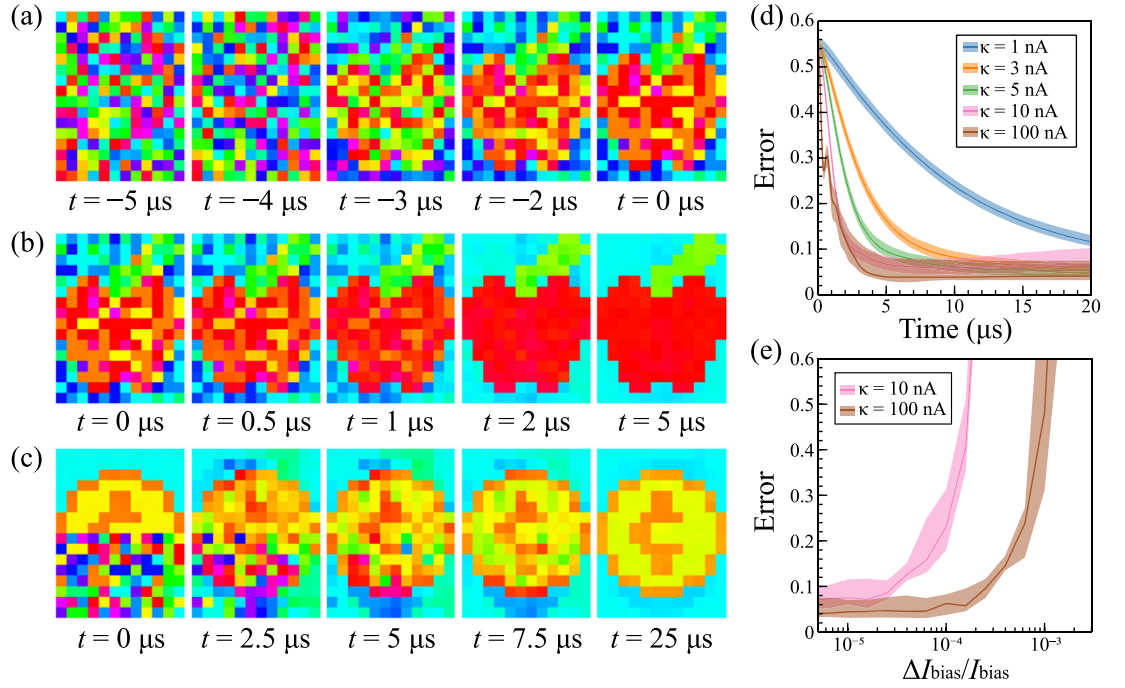
**Figure 4.** Representative snapshots of the image represented on the oscillator array (a) in the preparation stage and (b) in the retrieval stage. The entire query image is distorted using random, discrete noise that gives a root mean square distortion of one state along the color wheel. (c) Retrieval process of an incomplete query image. The upper-half the query image is intact, and the lower-half has randomized pixel values. (d) Error evolution as a function of time for 30 Gaussian distorted images of each of the 12 images, as illustrated in one distortion of one image in (b). (e) Average error at the end of 20 $\mu$s, for 30 distorted images of each of the 12 images, plotted as a function of relative standard deviation of the Gaussian DC bias spread of the oscillators. In (d) and (e), the solid line corresponds to the average error and the translucent fills around each solid line are the quartile bounds. The different curves are for different strengths of the feedback current, $\kappa$, as described in equation (2). In (b) and (c), $\kappa = 10$ nA.

each pair of neurons. When summed over all other neurons, typical values of the total feedback current for $\kappa = 10$ nA are around 150 nA, ranging up to close to 1 $\mu$A for some neurons.

The primary assumption used in the analysis thus far is that all oscillators run at a set fixed frequency. While it is possible to achieve phase and frequency locking of oscillators in the presence of small frequency variations between the oscillators, large variations cause desynchronization resulting in a failed retrieval process. Figure 4(e) shows the average error after 5 $\mu$s of relaxation for 30 distorted versions of each of the 12 images in the dataset, as described for figure 4(d). These simulations are performed for two coupling constants $\kappa$. The oscillator frequencies are varied by varying the DC biases. The final error is plotted against the fractional variations of the DC biases, with each oscillator bias varied independently using a Gaussian distribution. When the oscillator array desynchronizes, the final state error is high. Higher coupling values result in a larger threshold for desynchronization because the locking range increases linearly with the amplitude of the AC feedback current for values near or below 1 $\mu$A. A bias tolerance of 0.01% or lower is required for successful retrieval for a coupling of $\kappa = 10$ nA. For oscillators with a distribution of parameters, this tolerance indicates the degree to which the DC bias currents need to be controlled to keep the oscillators synchronized. While the sensitivity to variation in the locking frequencies could be improved by increasing the AC feedback amplitude, the improvement comes at the cost of increasing the energy consumed in the calculation.

The simulations thus far assume perfect implementation of offline-trained weights. In the presence of non-ideal writing, layout-related delay differences between synapses, and device-to-device variations, the synapses may produce a different value of delay and scaling than intended. These errors can be iteratively corrected to ensure proper storage of images on the oscillator array.

A typical iterative procedure to correct the implemented weights is shown in figure 5(a). With the initial weights implemented into the synaptic feedback matrix, the oscillator array is prepared to represent one of the $K$ original images with pixel values $\hat{x}_i^k$. Starting from this stored image state, the network is left to evolve in the recognition stage configuration. In the presence of imperfections, the network relaxes into an incorrect image with pixel values $\hat{a}_i^k$. This incorrect image retrieval process starting from ideal images is repeated for the entire dataset while accumulating the error and the weight update corrections. Here, we use a contrastive divergence
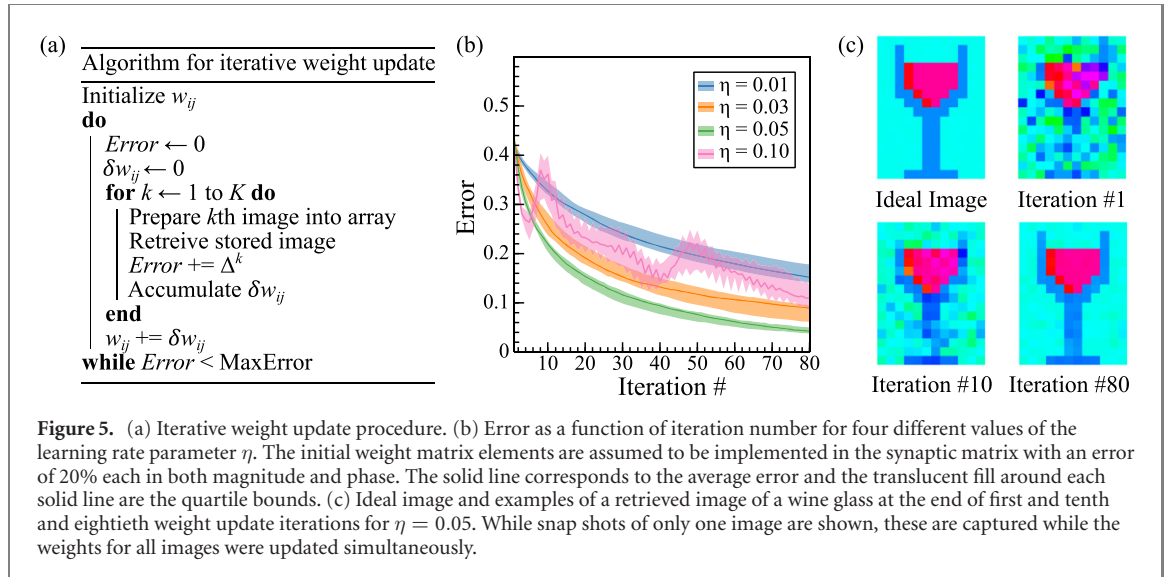
**Figure 5.** (a) Iterative weight update procedure. (b) Error as a function of iteration number for four different values of the learning rate parameter $\eta$. The initial weight matrix elements are assumed to be implemented in the synaptic matrix with an error of 20% each in both magnitude and phase. The solid line corresponds to the average error and the translucent fill around each solid line are the quartile bounds. (c) Ideal image and examples of a retrieved image of a wine glass at the end of first and tenth and eightieth weight update iterations for $\eta = 0.05$. While snap shots of only one image are shown, these are captured while the weights for all images were updated simultaneously.

learning rule, which gives the weight update $\delta w_{ij}$ correction at the end of each iteration as

$$\delta w_{ij} = \frac{\eta}{N} \sum_{k=1}^{K} [\hat{x}_i^k (\hat{x}_j^k)^* - \hat{a}_i^k (\hat{a}_j^k)^*], \qquad (4)$$
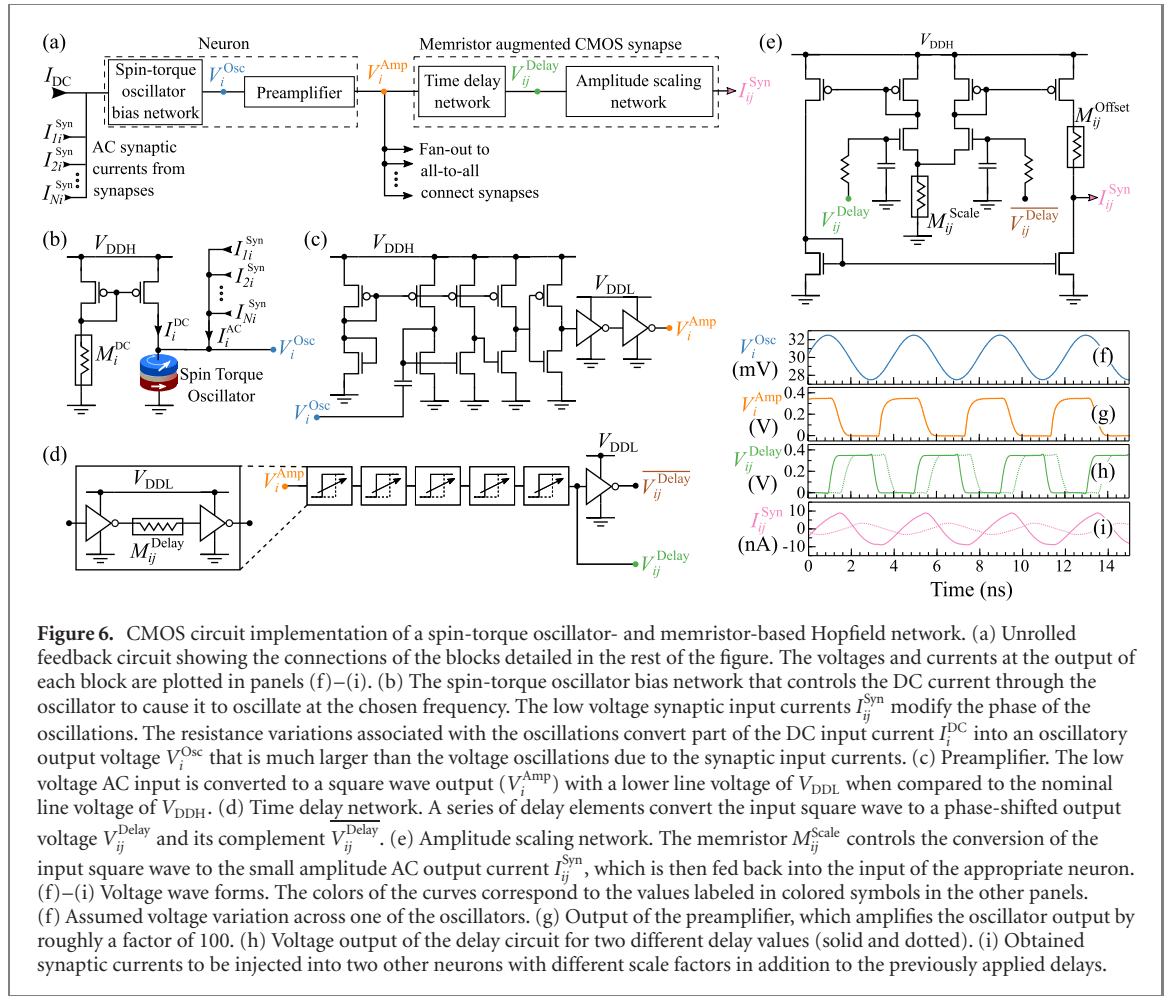
where $\eta$ is the learning rate. The weight update is carried out at the end of each iteration provided the total accumulated error is greater than the acceptable tolerance.

An example of the reduction of error with every weight update iteration is shown in figure 5(b) for four different learning rates $\eta$. The weight matrix elements are prepared to be in a state with a normal error of 20% each in their magnitude and phase. Weight updates are deterministic and are performed using the weight update rule in (4) at each iteration. Notice that the error reduces with the increase in the number of iterations, with the rate of reduction of error controlled by $\eta$. The rate of reduction of error increases concomitantly with $\eta$ until a critical value is reached. Beyond this critical value the corrections become too large to iteratively correct towards the right solution, as is the case with $\eta = 0.10$. Representative snapshots of retrieved images for $\eta = 0.05$ at the end of first, tenth, and eightieth iterations are shown in figure 5(c) along with an ideal image for comparison.

## 4. Circuits for complex synaptic weights

This section describes the detailed circuitry required to realize a complex Hopfield network using spin torque oscillators. The network is composed of neuron and synapse circuits which are recurrently connected as shown in figure 1(c). Unrolled versions of the blocks involved are shown in figure 6(a), each of which is discussed in more detail below. The neuron circuit consists of a bias network, which sets the current flowing through the spin-torque oscillator, a summation circuit, which sums the synaptic currents from the various branches, and a preamplifier circuit, which generates a square wave from the output voltage of the spin torque oscillator. This square wave passes through the synapse circuitry, which takes this square wave output and passes it through a programmable time delay network that performs phase shifting, and then a programmable amplitude scaling network that generates a scaled synaptic current output. These currents are fed back into the appropriate neurons, as the recurrent loop is completed. The output of each of the $N$ neurons feeds the input of a synapse connecting to each of the other $N - 1$ neurons, then each neuron sums the input from the other $N - 1$ neurons.

Detailed versions of the neuron circuits are shown in figures 6(b) and (c). The bias network consists of a memristor, $M_i^{\mathrm{DC}}$ in figure 6(b), in series with a diode-connected p-channel metal-oxide semiconductor field effect transistor (MOSFET). That pair is connected in parallel with another p-channel MOSFET in series with the spin-torque oscillator. Since both of those p-channel MOSFETs see the same gate–source voltage and are the dominant resistances in the two lines, the memristor-controlled current from the first branch is mirrored into the spin-torque oscillator. If the spin-torque oscillators could be fabricated with sufficient uniformity, see figure 4(e), a single programmable current branch could be shared amongst them to minimize power consumption. In the likely event that such uniformity is too difficult to fabricate, the memristors for each oscillator need to be tuned to give the same oscillation frequency for all of the oscillators to within 0.1% depending on the feedback current (see figure 4(e)). The AC synaptic currents from the various synapses are summed into

**Figure 6.** CMOS circuit implementation of a spin-torque oscillator- and memristor-based Hopfield network. (a) Unrolled feedback circuit showing the connections of the blocks detailed in the rest of the figure. The voltages and currents at the output of each block are plotted in panels (f)–(i). (b) The spin-torque oscillator bias network that controls the DC current through the oscillator to cause it to oscillate at the chosen frequency. The low voltage synaptic input currents $I_{ij}^{\text{Syn}}$ modify the phase of the oscillations. The resistance variations associated with the oscillations convert part of the DC input current $I_i^{\text{DC}}$ into an oscillatory output voltage $V_i^{\text{Osc}}$ that is much larger than the voltage oscillations due to the synaptic input currents. (c) Preamplifier. The low voltage AC input is converted to a square wave output ($V_i^{\text{Amp}}$) with a lower line voltage of $V_{\text{DDL}}$ when compared to the nominal line voltage of $V_{\text{DDH}}$. (d) Time delay network. A series of delay elements convert the input square wave to a phase-shifted output voltage $V_{ij}^{\text{Delay}}$ and its complement $\overline{V_{ij}^{\text{Delay}}}$. (e) Amplitude scaling network. The memristor $M_{ij}^{\text{Scale}}$ controls the conversion of the input square wave to the small amplitude AC output current $I_{ij}^{\text{Syn}}$, which is then fed back into the input of the appropriate neuron. (f)–(i) Voltage wave forms. The colors of the curves correspond to the values labeled in colored symbols in the other panels. (f) Assumed voltage variation across one of the oscillators. (g) Output of the preamplifier, which amplifies the oscillator output by roughly a factor of 100. (h) Voltage output of the delay circuit for two different delay values (solid and dotted). (i) Obtained synaptic currents to be injected into two other neurons with different scale factors in addition to the previously applied delays.

each oscillator by wiring them together. The oscillators respond to this summed current by adjusting their phase. Since the amplitude of the oscillation does not vary significantly, the DC current through the spin-torque oscillator produces an output voltage, $V_i^{\text{Osc}} \approx 5$ mV as shown in figure 6(f). This output signal is much smaller than $V_{\text{DDH}} = 0.7$ V.

The output signal is converted into a square wave by a preamplifier circuit shown in figure 6(c). No information is lost through this conversion of the oscillator signal into square wave because only the oscillation phases contain information and the phase information is preserved when the signal is converted into a square wave. Operating on square waves significantly reduces the power consumption in the delay circuitry in the synapses because the circuit only draws current during the transitions. A capacitor AC-couples the output of the oscillator to an appropriate bias voltage. The first part of the preamplifier has five branches connected to the nominal supply voltage in this technology, $V_{\text{DDH}}$. In the first branch, both transistors are diode connected and increased in size relative to the minimum at the chosen technology node to setup a bias current, which is mirrored into the second branch. The second branch uses this bias current to generate the appropriate bias voltage for the next two branches which form a two-stage amplifier. The third and fourth branches are connected in a common source topology, in which the n-channel MOSFET behaves as the input gain element, while the p-channel MOSFET behaves like a current source. The last branch, which is an inverter, produces a full-swing (0 V to $V_{\text{DDH}}$) square wave output. In order to achieve sufficiently high gains, the transistors forming the gain stages of the preamplifier are sized to be twice the minimum length for the particular technology model we use to simulate the circuits. The final part of the preamplifier consists of two appropriately sized inverters operating at a reduced supply voltage $V_{\text{DDL}} = 0.35$ V to match the synaptic circuit. The output is fanned out to the synaptic elements. The output of the preamplifier, denoted by $V_i^{\text{Amp}}$, is shown in figure 6(g).

A key feature of this network is the energy efficiency of the synaptic circuits that allow the implementation of complex weights for the feedback currents. The synaptic circuits consist of a time delay network and an amplitude scaling network as shown in figures 6(d) and (e). The time delay network is composed of a series of five delay cells. These delay cells, shown in the inset of figure 6(d), consist of two low voltage inverters and a memristor. The charging time of the gate of the second inverter of the delay cell is programmable through the $RC$ time constant set by the gate capacitance $C$ and the effective resistance $R$ which consists of the MOSFET

channel resistances and the tunable resistance of the memristor $M_{ij}^{\text{Delay}}$, thus determining the delay of each cell. The delay elements operate at a reduced supply voltage $V_{\text{DDL}}$. The active power in the delay networks scales as $V_{\text{DDL}}^2$. Therefore, a lower $V_{\text{DDL}}$ restricts the charging currents minimizing the power consumption of the delay network. Combining multiple delay elements ensures that the delays can be tuned to allow phase shifts between 0 to $2\pi$ as determined by $M_{ij}^{\text{Delay}}$. An additional inverter is included to output the complementary signal as needed in the amplitude scaling network.

Additional parasitics in the time delay network, such the capacitance of the memristor, would increase the energy cost of producing delays while also affecting the range of achievable delays. However, typical memristor parasitic capacitances are a fraction of the gate capacitances used in this study [58, 59] and we estimate that the achievable delays per RC block of the time delay network decrease by less than fifteen percent. Such reductions in the achievable delay ranges could be readily compensated by the addition of additional delay blocks or by allowing a larger resistance values for each memristor.

The scaled synaptic current is generated by feeding the inverted and non-inverted outputs of the delay cells to a differential transconductance amplifier, as shown in figure 6(e). The differential transconductance amplifier provides a scaled current output proportional to the applied differential voltage, with the scaling set by the memristor $M_{ij}^{\text{Scale}}$. The output of the delay cells is low-pass filtered by an *RC* network which sets the bias point of the input common mode to $V_{\text{DDL}}/2$. The memristor $M_{ij}^{\text{Scale}}$ controls the total currents in both branches and hence sets the gain of the input stage. The p-channel and the n-channel MOSFETs that comprise the current mirror steer the copied currents until they are subtracted from each other at the output branch. An offset tuning memristor $M_{ij}^{\text{Offset}}$ is added to account for drain−source voltage mismatches between the right and left branches. Since only a differential current is being output by the circuit, its magnitude can be kept low, while the individual branches of the circuit still operate at a larger current, constrained by the bias currents required for correct circuit functionality. The bandwidth of the scaling circuit designed here is much higher than the fundamental frequency of the oscillator. When we included the parasitic capacitances of the memristors, the changes to the feedback current were small enough that we did not observe any noticeable change in the circuit operation.

The oscillator phases are most affected by frequency components of $I_i^{\text{AC}}$ that are close to the oscillator's set frequency. Feedback components not in the range of the oscillator's locking frequencies, in particular, higher harmonics, have a weaker effect on the oscillator dynamics as long as their amplitude is low. We have tested the oscillator model we use by simulating the injection of sine waves, triangle waves, and square waves. All give similar locking ranges. We conclude from these simulations that for the size of the input currents we consider, $I_{ij}^{\text{Syn}}$ in figure 6(i), the higher harmonics do not affect the performance of the oscillator.

These CMOS circuits were designed using the Predictive Technology Model 16 nm high-power technology [60] and were simulated in Cadence Virtuoso[5]. They either operate at the nominal supply voltage for this technology, $V_{\text{DDH}} = 700$ mV, or a reduced supply voltage, $V_{\text{DDL}} = 350$ mV. Using a reduced supply voltage helps achieve the desired range of delays while reducing the power consumption. The memristors in the circuit are treated as simple resistors. Programming the memristors requires circuitry in addition to what we describe here, as we discuss in section 5.

Representative waveform outputs measured at various nodes labeled in figure 6(a) are shown in figures 6(f)−(i). These waveforms are measured for an unrolled network, without feedback. We assume a 248 MHz sine wave output for a spin-torque oscillator, as seen in figure 6(f), and track the intermediate nodal voltages and the output synaptic current of a synapse connected to this spin-torque oscillator. Figure 6(g) shows the saturated square output of the preamplifier. The preamplifier drives a time-delay network and a dummy inverter load that is sized to emulate the large fan-out at the end of the preamplifier. The time-shifted output voltages of the time-delay circuit for two chosen delays are shown in figure 6(h). The waveforms are square waves with a full voltage swing phase shifted by the selected phases. Finally, figure 6(i) shows the synaptic current output of the scale network that scales the two time-delayed waveforms in figure 6(h) by two different chosen values of scaling. Notice that each circuit in figures 6(b)−(e) introduces intrinsic constant phases to the oscillator output. The physical layout of this network will cause additional delays associated with the different path lengths. However, these intrinsic phase shifts can be compensated by appropriately adjusting the delays produced by the synaptic delay elements.

The synaptic weights are stored in memristors that control the phase shifting and the scaling of the feedback current. Figure 7(a) shows, as a function of the memristor resistance $M_{ij}^{\text{Delay}}$, the measured delay (left axis) produced at the end of the delay network relative to the spin-torque oscillator output. The right axis gives the

---

[5] Certain commercial products or company names are identified here to describe our study adequately. Such identification is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology, nor is it intended to imply that the products or names identified are necessarily the best available for the purpose.
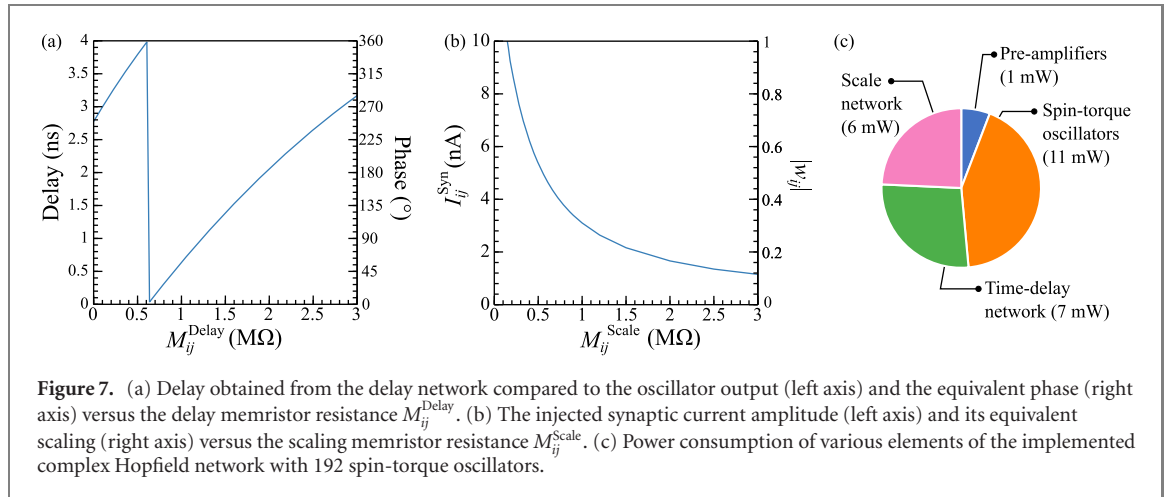
**Figure 7.** (a) Delay obtained from the delay network compared to the oscillator output (left axis) and the equivalent phase (right axis) versus the delay memristor resistance $M_{ij}^{\text{Delay}}$. (b) The injected synaptic current amplitude (left axis) and its equivalent scaling (right axis) versus the scaling memristor resistance $M_{ij}^{\text{Scale}}$. (c) Power consumption of various elements of the implemented complex Hopfield network with 192 spin-torque oscillators.

phase corresponding to this delay. As a result of the intrinsic delays introduced in other parts of the circuit, the zero-delay case occurs for a non-zero value of $M_{ij}^{\text{Delay}}$. Notice that a complete range of phase shifts from $0°$ to $360°$ can be produced for a resistance range of approximately 20 k$\Omega$ to 3 M$\Omega$. Figure 7(b) shows the measured synaptic current (left axis) and its corresponding weight scale factor (right axis) plotted against the memristor resistance $M_{ij}^{\text{Scale}}$. The range of realizable scale factors is limited by the possible range of resistances. Furthermore, the variation of the synaptic current with respect to $M_{ij}^{\text{Scale}}$ is highly non-linear, and as a result, for the training process, a look-up table is essential to correctly program the value of $M_{ij}^{\text{Scale}}$ for a desired scaling value.

For the technology we simulate, figure 7(c) gives the power consumed by various components of the 192-oscillator complex Hopfield network in figure 6(a). The 192 spin-torque oscillators along with the bias network draw about 11 mW of power and the corresponding preamplifier circuits following these oscillators draw about 1 mW of power. Although the power dissipation in individual synapses is small compared to the spin-torque oscillator networks, the large number of synapses (192 × 191) results in a significant power dissipation in the synaptic network. In the networks that we have designed, we estimate 7 mW of power consumption across all the time-delay networks and an additional 6 mW of power across the scaling networks. These estimates suggest an estimated 25 mW power consumption for this 192-spin-torque-oscillator- and memristor-based implementation of the complex Hopfield network. Assuming a 5 $\mu$s inference time following the results in figure 4(d), we estimate 130 nJ of energy consumed in the network per image retrieval. These power estimates do not include the energy spent in the preparation phase, which add roughly 55 nJ, nor does it include the energy needed to extract the oscillator phases for subsequent processing. The oscillations of the spin-torque oscillators could be extracted with the preamplifiers, making a negligible contribution to the energy. The cost of processing these phase fronts would depend on how the subsequent processing would be done.

## 5. Discussion

We have theoretically demonstrated inference on a spin-torque-oscillator-based complex Hopfield network with 192 oscillators. The goal of this paper is to establish the requirements on device performance and reproducibility that would enable the implementation of this and related networks. While the CMOS circuitry we use can be readily fabricated, the spin-torque oscillators and memristors we simulate are beyond the current state of the art. In this section, we discuss some of the current limitations that need to be addressed. Hopefully, this discussion can guide future development efforts designed at taking advantage of spin-torque oscillators and memristors.

There are several obstacles to the immediate implementation of this approach. The first is the ability to synchronize a large number of spin-torque oscillators. Current experimental demonstrations of electrically synchronizing spin-torque oscillators are limited to a few oscillators with synchronization times limited to a few milliseconds [61]. Such experimental demonstrations are performed in small-scale research labs that lack large scale production capabilities. Commercial fabrication facilities could have the needed capabilities to produce sufficiently uniform devices, however, no attempts have been made to do so. Therefore, implementing large arrays of spin-torque oscillators for phase-based computing remains unexplored. Strict control over the frequency dispersion of the individual oscillators, which could be achieved by refining the quality of materials during fabrication, is essential to increase the number of oscillators that can be synchronized. For the low feedback currents considered here, synchronizing 192 oscillators would require a frequency dispersion of less

than 0.1% as shown in figure 4(e). The upper bound on the feedback coupling strength is set by non-linearities associated with the oscillators. As the summed feedback currents become comparable the bias current, the phase locking characteristics of the oscillator to a mixture of ac signals is no longer linear. Additional distortions may be introduced by the CMOS feedback network with increase in the feedback strength. Smaller oscillator networks allow for larger feedback strengths which correspond to a higher tolerance of frequency dispersion.

One aspect that is important to investigate before realizing oscillator networks in practice is noise. Spin-torque oscillators operating at finite temperatures have phase noise [62, 63], which will interfere with synchronization. These measurements report resonance peak widths that are 0.1% to 1% of their free running frequencies. These widths are broader than the frequency spreads required by the network described above. However, synchronization with an external source, here, the feedback from the other oscillators, significantly reduces the phase noise. Whether this finite temperature phase noise requires greater feedback than is used in this network is left to future work.

The CMOS components used in the feedback circuitry also introduce noise. This contribution can be particularly significant to the feedback currents that are in the range of a few nanoamperes. We performed SPICE simulations including thermal noise arising from the CMOS scaling network and determined that the thermal noise amplitudes have an root mean square value of 15 nA, thus being comparable to the feedback currents arising from the scaling network. However, because the oscillator locking range is restricted to a megahertz or less depending on the strength of the feedback, the presence of a broadband noise such as that arising from thermal sources has little effect on its frequency locking behavior, as we have checked in simulations of the oscillators. Consequently, we expect that CMOS noise has little impact on the network performance provided feedback currents are not made too small.

Another obstacle is programming the memristors used to provide local memory in the network. We require analog control of memristor resistances to encode the complex weights and to control the bias currents of the individual oscillators in the case that oscillators need to be individually tuned to match their frequencies. Local analog memory using memristors is an active area of research [43, 64] and our approach would not be viable without its realization. There are different constraints on the memristors we use in different parts of the network. While errors in precise analog control of memristors used for programming the complex weights only introduce errors in recalled vectors, errors in programming the bias control memristors result in catastrophic decoherence of the oscillator networks.

We have not included memristor programming and control circuitry in figure 6. While memristors with the chosen resistance ranges have been fabricated [65, 66], such memristors require high-voltage thick oxide transistors to decouple the low-voltage circuits from larger (up to $\approx$2 V) forming and write voltages. Designing such circuits would require either working in a technology that includes these higher voltages and associated transistors or the development of memristors with improved capabilities. Improving the voltage-level compatibility of memristors with highly scaled CMOS transistors is an important goal of ongoing research [49, 50, 67] for almost all envisioned applications of memristors. Without sufficient reduction in memristor writing voltages, the overhead associated with the write circuitry and the associated energy costs could be far higher than that of the inference circuitry used in this study.

If it were possible to fabricate oscillators with a sufficiently low frequency spread, this paper shows that low power CMOS circuitry can be used to couple them in ways to enable energy-efficient processing. Identifying paths to increase the efficiency of this approach is complicated by the combination of technologies involved. The power shown in figure 7(c) is balanced between the neuronal and synaptic parts of the circuit but for larger numbers of neurons will be dominated by the synaptic parts of the network that scale as the number of oscillators squared. As the size of images is increased, the power will scale as the square of the size of the stored images. However, the energy per inference may scale more steeply than that if it takes longer for the larger network to converge. It may be possible to reduce the power consumed by the synapses by using a more advanced technology node, but scaling arguments are not simple. Unlike scaling arguments for pure CMOS circuits, the properties of the spin-torque oscillators and the memristors must scale in the right way as well.

Power consumed by the spin-torque oscillators could be reduced further by scaling down the dimensions of the oscillators. Again, the scaling arguments are not simple for several reasons. Larger current densities are required to sustain oscillations as spin-torque oscillators are scaled down [63]. As a result, although the total DC current is reduced by scaling down the oscillators, this reduction is lower than the area reduction obtained by scaling. On the other hand, scaling down the oscillators increases the oscillation frequency. Use of faster spin-torque oscillators reduces the synchronization time, and thus would be expected to reduce the image retrieval times and the associated overall energy costs. Faster oscillations also allow the design of synapse circuitry with lower $RC$ time constants, further driving down the power dissipated. Current experimental realizations of spin-torque oscillators have been limited to diameters of hundreds of nanometers [68, 69], as presumed in this study. Theoretically predicted scaling suggests that the spin-torque oscillator radius could be reduced to tens of nanometers [37]. As discussed in appendix A, the rate of phase convergence depends on the strength of

the field-like torque, so using materials that increase this parameter will lead to faster convergence and a lower energy cost.

Non-linear oscillators could be implemented using alternative technologies such as spin-Hall oscillators [31], vanadium-dioxide oscillators [70] and opto-electronic oscillators [26]. The delay and scaling networks proposed in this work can be tuned to operate between tens of megahertz to a gigahertz. Slower oscillations require large $RC$ time constants, thus increasing the power consumption. Therefore, alternative methods to realize tunable delay elements would be needed for slow oscillators such as those based on vanadium-dioxide. Frequencies higher than a gigahertz would be possible but would require redesigning the circuits at the cost of increasing the power required. However, we consider it likely that the energy per computation would be similar or lower because the power increase would be offset by a decrease in computation time resulting from a higher operating frequency.

There has been significant recent progress in developing more powerful Hopfield networks by implementing energy functions with terms with higher powers of the neuron states than quadratic [19, 71]. The quadratic model used here allows for all-to-all connections for all pairs of neurons. Implementing higher order terms in the energy functional electrically would require many neuron interactions and exploding levels of circuitry. For example, a quartic term would generate for each oscillator contributions to the feedback from all possible trios of oscillators, increasing the number of synaptic pathways from the roughly $4 \times 10^4$ considered here to $16 \times 10^8$. The difficulty in wiring such many-body interactions reflects the complications involved in treating many-body interactions in physical systems. It may be possible to make approximate implementations modeled after some of the successful approximations from many body physics.

The all-to-all connections in this type of network make it most suitable for storing information in which each 'pixel' is strongly correlated with each other one. One possible way to improve the performance of Hopfield networks for image processing could be to take advantage of the locality of information in images. In existing implementations, each pixel is connected equally strongly to all other pixels no matter how close they are spatially. Neural networks [72] frequently use local convolutional filters in the initial layers to process pixels that are spatially close. A similar approach, such as making the interactions in the energy functional local, might significantly reduce the energy consumption without substantially reducing the performance of the network. For other applications, reducing the range of interactions might lead to more efficient implementations without loss of performance if the range is tuned to match the range of correlations in the stored data.

This section discusses several different lines of development needed for the hardware implementations of Hopfield networks proposed here. A key enabler will be advances in manufacturing uniformity and reliability for both spin-torque oscillators and memristors as these technologies transition from laboratories to fabrication facilities. Our results show that it is possible for hardware implementations of Hopfield networks to encode multilevel or continuous information, the possibility of which greatly increases the types of data that could be usefully stored in such an array.

## Acknowledgments

## Data availability statement

The data that support the findings of this study are available upon reasonable request from the authors.

## Appendix A. Spin-torque oscillator model

There have been several types of spin-torque oscillators that have been experimentally characterized [36, 40]. For specificity, we focus on a vortex-based spin-torque oscillator based on a circular MTJ in which the magnetization of the free layer assumes a vortex configuration [36]. Under the influence of the spin-polarized tunneling current, the resulting spin transfer torque causes the vortex center to precess around the center of

**Table A1.** Spin-torque oscillator parameters used in the study.

| Parameter | Description | Value | |
|---|---|---|---|
| $G$ | Gyrovector amplitude | $1.14 \times 10^{-13}$ | J s m$^{-2}$ rad$^{-1}$ |
| $D_0$ | 1st-order damping constant | $5.08 \times 10^{-16}$ | J s m$^{-2}$ rad$^{-1}$ |
| $D_1$ | 2nd-order damping constant | $9.51 \times 10^{-17}$ | J s m$^{-2}$ rad$^{-1}$ |
| $\kappa_{MS}^0$ | 1st-order magnetostatic confinement constant | $1.41 \times 10^{-4}$ | J m$^{-2}$ |
| $\kappa_{MS}^1$ | 2nd-order magnetostatic confinement constant | $3.53 \times 10^{-5}$ | J m$^{-2}$ |
| $\kappa_{Oe}^0$ | 1st-order Oersted field confinement constant | $3.40 \times 10^{-16}$ | J A$^{-1}$ |
| $\kappa_{Oe}^1$ | 2nd-order Oersted field confinement constant | $-1.70 \times 10^{-16}$ | J A$^{-1}$ |
| $a_J$ | Orthogonal spin-transfer efficiency parameter | $3.10 \times 10^{-16}$ | J A$^{-1}$ |
| $b_J$ | In-plane spin-transfer efficiency parameter | $8.26 \times 10^{-17}$ | J A$^{-1}$ |
| $r_0$ | Free-layer radius | 100 | nm |

the disc. As it does, the relative orientation between the magnetization in the precessing vortex layer and the fixed layer changes giving rise to an oscillating resistance.

We model the zero-temperature dynamics of the free-layer of the vortex-based spin-torque oscillator using a Thiele approach [73]. In this approach, the dynamics of the vortex state is parameterized by the polar coordinates $\mathbf{r} = r_0(\rho \cos \theta, \rho \sin \theta, 0)$ of the vortex center in the plane of the free layer, where $r_0$ is the radius of the disc, making the radial coordinate $\rho$ dimensionless. This approach captures the dominant *gyrotropic mode* of the vortex and ignores vortex distortions. The time evolution of the vortex is described by [68]

$$\dot{\rho} = a\rho - b\rho^3 - c\,\cos(\theta),$$
$$\dot{\theta} = \omega_0 + \omega_1 \rho^2 + \frac{c}{\rho}\,\sin(\theta), \tag{A.1}$$

where

$$a = \frac{a_J J_i}{G} - \frac{D_0}{G}\omega_0,$$
$$b = \frac{D_1}{G}\omega_0 + \frac{D_0}{G}\omega_1,$$
$$c = \frac{b_J J_i}{G},$$
$$\omega_0 = \frac{1}{G}\left(\kappa_{MS}^0 + \kappa_{Oe}^0 J_i\right),$$
$$\omega_1 = \frac{1}{G}\left(\kappa_{MS}^1 + \kappa_{Oe}^1 J_i\right). \tag{A.2}$$

The description and values corresponding to the geometry- and material-dependent parameters that appear in equation (A.2) are given in table A1. The nominal bias current through the $i$th spin-torque oscillator $I_i^{DC} = J_i^{DC}(\pi r_0^2)$ is set to 80 $\mu$A so that the spin-torque oscillator produces self-sustained oscillations at 248 MHz. The total injected current $I_i = I_i^{AC} + I_i^{DC}$ (and the corresponding total current densities $J_i = J_i^{AC} + J_i^{DC}$), where $I_i^{AC}$ is the sum total AC feedback from all synaptic connections.

In the network simulations in section 3, we integrate these equations to obtain the time evolution of the vortex cores of the oscillators. The oscillators evolve to a synchronized state determined by their AC feedback currents. To connect equation (A.1) to the formulation used in the main text in terms of the complex number $\hat{a}_i = e^{i\phi_i}$ encoding the phase, we write $\theta_i = \omega t + \phi_i$. The time dependence of $\hat{a}_i$ is then

$$\frac{d\hat{a}_i}{dt} = i\hat{a}_i(\dot{\theta}_i - \omega).$$

The DC current is adjusted for each oscillator so that in the absence of an AC current, $\omega = \omega_0 + \omega_1 a/b$ is the chosen operating frequency, where for each oscillator $\omega_0$, $\omega_1$, $a$, and $b$ take values in the DC limit. The time evolution of $\hat{a}_i$ depends on the instantaneous values of $\theta$ and $\rho$ and the input AC component of the current through a complicated function determined by equation (A.1). Unfortunately, we have not been able to identify a limit in which this function becomes physically transparent.

We can gain insight into the phase-locking process by focusing on low order effects. First, consider the case with no AC current so that all parameters in equation (A.1) are time independent. Neglecting higher harmonic behaviors, we assume

$$\rho = \rho_0 + \rho_1 \cos(\omega t + \chi)$$

$$\theta = \omega t + \phi + \phi_1 \cos(\omega t + \zeta).$$

The steady state contributions in equation (A.1) lead to

$$\rho_0^2 = a_0/b$$

$$\omega = \omega_0 + \omega_1 \rho_0^2 = \omega_0 + \omega_1 a_0/b, \tag{A.3}$$

where $a_0$ is the part of $a$ due to just the DC current. There are additional contributions to this equation from the static part of quantities like $\rho_1^2 \cos^2(\omega t + \zeta)$, but they can be neglected because $\phi_1$ and $\rho_1$ are small. The equation of motion for $\rho$ becomes

$$-\omega\rho_1 \sin(\omega t + \chi) = a_0\rho_1 \cos(\omega t + \chi) - 3b\rho_0^2\rho_1 \cos(\omega t + \chi) - c \cos(\omega t + \phi), \tag{A.4}$$

where we neglect the $\phi_1 \cos(\omega t + \zeta)$ inside the cos because it leads to higher harmonics and static contributions that are higher order in $\phi_1$. Using the static results and noting that $a_0 \ll \omega$ gives $\rho_1 \approx c/\omega$, which is small because $c \ll \omega$, and $\chi \approx \phi + \pi/2$. After canceling the static parts and neglecting higher harmonics, the equation of motion for $\theta$ becomes

$$-\omega\phi_1 \sin(\omega t + \zeta) = 2\omega_1\rho_0\rho_1 \cos(\omega t + \chi) + \frac{c}{\rho_0} \sin(\omega t + \phi).$$

Substituting $\pi/2 + \phi$ for $\chi$ and $c/\omega$ for $\rho_1$, collecting terms, and simplifying gives $\phi_1 \approx c/(\omega\rho_0)$ and $\zeta = \phi + \pi$. This fixes all parameters of steady precession in the absence of AC currents except for the arbitrary phase $\phi$. Adding in the AC current fixes $\phi$ relative to the phase of the AC input. Numerical simulations consistently give $\phi = \pi/2$ in the phase-locked state. The small values we find for $\delta_1$ and $\theta_1$ mean that the precession is very circular in this regime and the AC voltage output very sinusoidal.

These results indicate which parameters in equation (A.1) determine different aspects of the motion. The steady state results, equation (A.3) show that the damping-like torque, through $a$, and the damping, primarily through $b$, determine the steady state radius of gyration, $\rho_0$ and the frequency $\omega$. The frequency-dependent contribution, which distorts the circular orbit, as seen in $\rho_1$ and $\phi_1$ in equation (A.4), enters through the field-like torque parameter, $c$. With no AC current present, the overall phase of the oscillation, $\phi$ is undetermined. Introducing an AC current adds an AC driving term through $a$ into equation (A.1). The only terms in equation (A.1) that depend on the phase of the gyration are those derived from the field-like torque. Thus, the strength of the field-like torque determines how quickly the phase locks to the phase of the AC driving current.

Assuming that the fixed layer is magnetized in-plane along the $x$-direction, the variation resistance of the spin-torque oscillator as a function of the oscillating core coordinates is

$$\Delta R = \lambda\xi\Delta R_0\rho \sin(\theta). \tag{A.5}$$

Here, $\lambda$ is the average magnetization to vortex displacement ratio, taken to be 2/3, $\xi = \pm1$ depending on the helicity and the direction of oscillation of the vortex core. $\Delta R_0 = (R_{AP} - R_P)/2$, with $R_{AP}$ and $R_P$ being the resistance of the MTJ when the free layer is magnetized entirely antiparallel and parallel, respectively, with respect to the fixed layer.

## ORCID iDs

Nitin Prasad ⓘ https://orcid.org/0000-0001-8088-0164
Prashansa Mukim ⓘ https://orcid.org/0000-0002-1150-1826
Advait Madhavan ⓘ https://orcid.org/0000-0002-4121-1336
Mark D Stiles ⓘ https://orcid.org/0000-0001-8238-4156

## References

[1] Villars R L, Olofson C W and Eastwood M 2011 *White paper, IDC* vol 14 pp 1–14
[2] Williams R S 2017 *Comput. Sci. Eng.* **19** 7–13
[3] Indiveri G and Liu S C 2015 *Proc. IEEE* **103** 1379–97
[4] Kim S *et al* 2015 *IEEE Int. Electron Devices Meeting (IEDM)* pp 17.1.1–4
[5] Wang Z *et al* 2017 *Nat. Mater.* **16** 101–8
[6] Torrejon J *et al* 2017 *Nature* **547** 428–31
[7] Gautrais J and Thorpe S 1998 *Biosystems* **48** 57–65
[8] Brette R 2015 *Front. Syst. Neurosci.* **9** 151

[9]  Hoppensteadt F C and Izhikevich E M 1999 *Phys. Rev. Lett.* **82** 2983–6
[10] Levine D S, Brown V R and Shirey T 1999 *Oscillations in Neural Systems* (Mahwah, NJ: Lawrence Erlbaum Associates)
[11] Thompson R F *et al* 1997 *Int. Rev. Neurobiol.* **41** 151–89
[12] Izhikevich E M and Kuramoto Y 2006 *Encycl. Math. Phys.* **5** 448
[13] Nikonov D E, Csaba G, Porod W, Shibata T, Voils D, Hammerstrom D, Young I A and Bourianoff G I 2015 *IEEE J. Explor. Solid-State Comput. Devices Circuits* **1** 85–93
[14] Abbott L F 1990 *J. Phys. A: Math. Gen.* **23** 3835–59
[15] Hopfield J J 1982 *Proc. Natl Acad. Sci. USA* **79** 2554–8
[16] Storkey A and Valabregue R 1999 *Neural Netw.* **12** 869–76
[17] Krotov D and Hopfield J J 2016 *Adv. Neural Inf. Process. Syst.* vol 29 pp 1172–80
[18] Demircigil M, Heusel J, Löwe M, Upgang S and Vermet F 2017 *J. Stat. Phys.* **168** 288–99
[19] Ramsauer H *et al* 2021 Hopfield networks is all you need (arXiv:2008.02217)
[20] Rieger H 1990 *J. Phys. A: Math. Gen.* **23** L1273–9
[21] Zurada J M, Cloete I and van der Poel E 1996 *Neurocomputing* **13** 135–49
[22] Jankowski S, Lozowski A and Zurada J 1996 *IEEE Trans. Neural Netw.* **7** 1491–6
[23] Best R E 2007 *Phase-Locked Loops: Design, Simulation, and Applications* 6th edn (New York: McGraw-Hill)
[24] Parihar A, Shukla N, Jerry M, Datta S and Raychowdhury A 2017 *Sci. Rep.* **7** 911
[25] Núñez J, Avedillo M J, Jiménez M, Quintana J M, Todri-Sanial A, Corti E, Karg S and Linares-Barranco B 2021 *Front. Neurosci.* **15** 655823
[26] Farhat N H, Psaltis D, Prata A and Paek E 1985 *Appl. Opt.* **24** 1469–75
[27] Jang J S, Jung S W, Lee S Y and Shin S Y 1988 *Opt. Lett.* **13** 248–50
[28] Yuasa S, Nagahama T, Fukushima A, Suzuki Y and Ando K 2004 *Nat. Mater.* **3** 868–71
[29] Ralph D C and Stiles M D 2008 *J. Magn. Magn. Mater.* **320** 1190–216
[30] Apalkov D *et al* 2013 *J. Emerg. Technol. Comput. Syst.* **9** 1–35
[31] Chen T *et al* 2016 *Proc. IEEE* **104** 1919–45
[32] Liu X, Zhang W, Carter M J and Xiao G 2011 *J. Appl. Phys.* **110** 033910
[33] Kim J V 2012 *Solid State Phys.* **63** 217–94
[34] Rippard W H, Pufall M R, Kaka S, Silva T J, Russek S E and Katine J A 2005 *Phys. Rev. Lett.* **95** 067203
[35] Tamaru S, Kubota H, Yakushiji K, Yuasa S and Fukushima A 2015 *Sci. Rep.* **5** 18134
[36] Pribiag V S, Krivorotov I N, Fuchs G D, Braganca P M, Ozatay O, Sankey J C, Ralph D C and Buhrman R A 2007 *Nat. Phys.* **3** 498–503
[37] Romera M *et al* 2018 *Nature* **563** 230–4
[38] Leroux N, Marković D, Martin E, Petrisor T, Querlioz D, Mizrahi A and Grollier J 2021 *Phys. Rev. Appl.* **15** 034067
[39] Koo M, Pufall M R, Shim Y, Kos A B, Csaba G, Porod W, Rippard W H and Roy K 2020 *Phys. Rev. Appl.* **14** 034001
[40] Kaka S, Pufall M R, Rippard W H, Silva T J, Russek S E and Katine J A 2005 *Nature* **437** 389–92
[41] Zahedinejad M, Awad A A, Muralidhar S, Khymyn R, Fulara H, Mazraati H, Dvornik M and Åkerman J 2020 *Nat. Nanotechnol.* **15** 47–52
[42] Zahedinejad M, Fulara H, Khymyn R, Houshang A, Fukami S, Kanai S, Ohno H and Åkerman J 2021 *Nat. Mater.* **21** 81–7
[43] Wang Z, Wu H, Burr G W, Hwang C S, Wang K L, Xia Q and Yang J J 2020 *Nat. Rev. Mater.* **5** 173–95
[44] Yi W, Tsang K K, Lam S K, Bai X, Crowell J A and Flores E A 2018 *Nat. Commun.* **9** 4661
[45] Choi B *et al* 2005 *J. Appl. Phys.* **98** 033715
[46] Govoreanu B *et al* 2011 *IEEE Int. Electron Devices Meeting (IEDM)* (IEEE) pp 31–6
[47] Xiao Z and Huang J 2016 *Adv. Electron. Mater.* **2** 1600100
[48] Li Y, Zhong Y, Xu L, Zhang J, Xu X, Sun H and Miao X 2013 *Sci. Rep.* **3** 1619
[49] Golonzka O *et al* 2019 *Symp. on VLSI Technology* (Piscataway, NJ: IEEE) pp T230–1
[50] Jain P *et al* 2019 *IEEE Int. Solid-State Circuits Conf.-(ISSCC)* (Piscataway, NJ: IEEE) pp 212–4
[51] Burr G, Narayanan P, Shelby R, Sidler S, Boybat I, di Nolfo C and Leblebici Y 2015 *IEEE Int. Electron Devices Meeting (IEDM)* (IEEE) p 4
[52] Prezioso M, Merrikh-Bayat F, Hoskins B, Adam G C, Likharev K K and Strukov D B 2015 *Nature* **521** 61–4
[53] Yu S, Jiang H, Huang S, Peng X and Lu A 2021 *IEEE Circuits Syst. Mag.* **21** 31–56
[54] Muezzinoglu M, Guzelis C and Zurada J 2003 *IEEE Trans. Neural Netw.* **14** 891–9
[55] Kanter I and Sompolinsky H 1987 *Phys. Rev. A* **35** 380–92
[56] Movellan J R 1991 Contrastive Hebbian learning in the continuous Hopfield model *Connectionist Models* ed D S Touretzky, J L Elman, T J Sejnowski and G E Hinton (San Mateo, CA: Morgan Kaufmann Publishers) pp 10–7
[57] Gorodnichy D 1999 *IJCNN'99. Int. Joint Conf. on Neural Networks. Proc. (Cat. No.99CH36339)* vol 1 pp 663–8
[58] Fouda M E, Eltawil A M and Kurdahi F 2017 *IEEE Trans. Circuits Syst.* I **65** 270–82
[59] Madhavan A, Sherwood T and Strukov D B 2018 *IEEE Trans. VLSI Syst.* **26** 2759–72
[60] Predictive Technology Model 2021 http://ptm.asu.edu (accessed 26 October 2021)
[61] Tsunegi S, Taniguchi T, Lebrun R, Yakushiji K, Cros V, Grollier J, Fukushima A, Yuasa S and Kubota H 2018 *Sci. Rep.* **8** 13475
[62] Keller M W, Kos A B, Silva T J, Rippard W H and Pufall M R 2009 *Appl. Phys. Lett.* **94** 193105
[63] Dussaux A, Khvalkovskiy A, Bortolotti P, Grollier J, Cros V and Fert A 2012 *Phys. Rev.* B **86** 014402
[64] Sarwat S G, Kersting B, Moraitis T, Jonnalagadda V P and Sebastian A 2022 *Nat. Nanotechnol.* **17** 507–13
[65] Xia Q and Yang J J 2019 *Nat. Mater.* **18** 309–23
[66] Li C *et al* 2018 *Nat. Electron.* **1** 52
[67] Yao P, Wu H, Gao B, Tang J, Zhang Q, Zhang W, Yang J J and Qian H 2020 *Nature* **577** 641–6
[68] Khalsa G, Stiles M D and Grollier J 2015 *Appl. Phys. Lett.* **106** 242402
[69] Tsunegi S *et al* 2016 *Sci. Rep.* **6** 26849
[70] Corti E, Gotsmann B, Moselund K, Ionescu A M, Robertson J and Karg S 2020 *Solid-State Electron.* **168** 107729
[71] Krotov D and Hopfield J J 2021 Large associative memory problem in neurobiology and machine learning (arXiv:2008.06996)
[72] Lawrence S, Giles C L, Tsoi A C and Back A D 1997 *IEEE Trans. Neural Netw.* **8** 98–113
[73] Guslienko K Y 2008 *J. Nanosci. Nanotechnol.* **8** 2745–60