Neural networks three ways: unlocking novel computing schemes using magnetic tunnel junction stochasticity

Matthew W. Daniels[a], William A. Borders[a], Nitin Prasad[b,c], Advait Madhavan[b,c], Sidra Gibeault[c], Temitayo Adeyeye[c], Liam Pocher[c], Lei Wan[d], Michael Tran[d], Jordan A. Katine[d], Daniel Lathrop[c], Brian Hoskins[a], Tiffany Santos[d], Patrick Braganca[d], Mark D. Stiles[a], and Jabez J. McClelland[a]

[a]Physical Measurement Laboratory, National Institute of Standards and Technology, Gaithersburg, MD, USA
[b]Associate, Physical Measurement Laboratory, National Institute of Standards and Technology, Gaithersburg, MD, USA
[c]Institute for Research in Electronics and Applied Physics, University of Maryland, College Park, MD, USA
[d]Western Digital Research, San Jose, CA, USA

ABSTRACT

Due to their interesting physical properties, myriad operational regimes, small size, and industrial fabrication maturity, magnetic tunnel junctions are uniquely suited for unlocking novel computing schemes for in-hardware neuromorphic computing. In this paper, we focus on the stochastic response of magnetic tunnel junctions, illustrating three different ways in which the probabilistic response of a device can be used to achieve useful neuromorphic computing power.

Keywords: Spintronics, neuromorphic computing, magnetic tunnel junction, Ising model, neural networks

1. INTRODUCTION

At both the edge and in datacenters, the energy efficiency of computing systems is an increasingly important optimization target. Two technological paradigms are well-suited to reaching this target: domain-specific computing systems optimized for machine learning applications, which increasingly dominate edge and cloud workloads; and non-volatile memory technologies, which can drastically reduce the static power consumption of traditional circuits and are conceptually well-suited to neuromorphic computing architectures.[1,2]

Among nonvolatile memory devices, magnetic tunnel junctions (MTJs) are of particular interest.[3] They exist in multiple fabrication facilities at multiple technology nodes, and are more technologically mature and less susceptible to reproducibility issues than other types of novel non-volatile memories like resistive switches. They also have broad multiphysical dynamics; slightly different growth and fabrication of nominally similar devices can produce memory elements, nano-oscillators, or random bit generators.[4] The stochastic properties of these devices are particularly well-suited for hardware implementation, as the stochastic behavior arises chiefly from thermodynamic processes and does not rely on defects or device imperfections which can be difficult to precisely replicate at scale.

In this paper, we non-exhaustively survey applications of magnetic tunnel junction stochasticity to neuromorphic computing in hardware. For each application, we take a critical eye to when the use of this application in hardware makes sense and what the limiting technological factors are. Through that lens, we discuss device properties of magnetic tunnel junctions and prescribe what fabrication targets would be needed to advance the technology readiness of each application.

Further author information: (Send correspondence to M.W.D.)
M.W.D.: E-mail: matthew.daniels@nist.gov, Telephone: 1 301 975 5601
J.J.M.: E-mail: jabez.mcclelland@nist.gov, Telephone: 1 301 975 3721

Spintronics XVI, edited by Jean-Eric Wegrowe, Joseph S. Friedman, Manijeh Razeghi, Proc. of SPIE Vol. 12656, 126560B
 · 0277-786X · doi: 10.1117/12.2682099

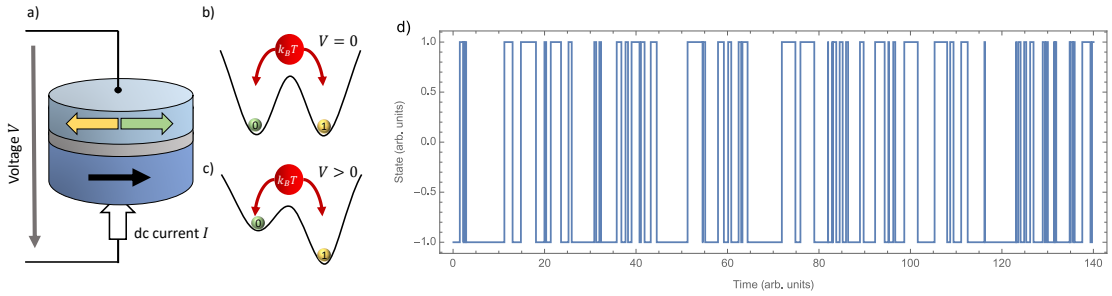

Figure 1. a) A schematic theoretical model of a superparamagnetic tunnel junction. The free layer (top) has two metastable states, either parallel or antiparallel to the fixed layer (bottom). b) In an ideal theoretical model, either of the states is equally likely when no voltage is applied to the device. An applied magnetic field is often needed to achieve this in practice. c) When a voltage is applied, the spin transfer torque on the free layer creates a statistical bias toward one of the states. d) Asymmetric telegraph noise of the type typically seen in superparamagnetic tunnel junctions. Real experimental data usually include voltage fluctations around the two states, but can be digitized to produce two clean states like those depicted here.

The rest of the paper is organized as follows. First we briefly review the physics of magnetic tunnel junctions in Sec. 2. In Sec. 3, we discuss the simulation of Ising models using superparamagnetic tunnel junctions (SMTJs), with applications to associative memories and combinatorial optimization problems. In Sec. 4, we discuss stochastic computing with a particular focus on deep neural network architectures, again using superparamagnetic tunnel junctions. In Sec. 5, we continue our discussion of deep neural networks, but consider how magnetic tunnel junctions can be used to augment the capabilities of low-precision digital accelerators.

2. MAGNETIC TUNNEL JUNCTIONS

The standard theoretical model of a magnetic tunnel junction consists of three components: a *fixed layer* of magnetic material whose magnetization has been locked in a chosen direction; a *tunneling barrier* of insulating material through which charge can pass via spin-dependent quantum tunneling processes; and a *free layer* of magnetic material whose magnetization is bistable and, in equilibrium, is oriented either parallel (P) or antiparallel (AP) to the state of the fixed layer. This theoretical structure is depicted schematically in Fig. 1(a).

The bistable configuration of the free layer is often conceptualized as a particle (representing the component of the magnetization projected onto the fixed layer's preferred axis) in a double-well potential. The stability of the two local minima is then controlled by the height of the potential barrier as in Figs. 1(b,c). Applying a voltage across the junction can make one well or the other energetically favorable by modifying the torques experienced by the free layer. Similar biasing of the states can be achieved by application of an external magnetic field.

For the purposes of this paper, there are two key sources of stochasticity that interest us. First, if the free energy barrier separating the two local minima of the energy landscape is lowered to be on the order of a few kT (throughout we generally consider T to be room temperature; k is Boltzmann's constant), then one expects the system to switch between the two states according to Kramer's escape rate theory. This process was originally formalized by Brown[5] and led to the Néel-Brown model of superparamagnetic tunnel junctions, which models the device dynamics as a two-state Markov process with escape rate

$$\varphi_j = \varphi_0 \exp\left[-\frac{\Delta}{kT}\frac{(-1)^j I}{I_c}\right] \tag{1}$$

out of state $j \in \{0, 1\}$, where I is the current through the device, I_c a characteristic current scale, φ_0 is a characteristic escape rate (sometimes called the attempt rate). The Néel-Brown model thus gives rise to asymmetric random telegraph noise, whose statistical properties have been detailed by Fitzhugh.[6]

For many of the applications we consider, the Néel-Brown model is a sufficient description of the device dynamics – usually because support circuitry is used to discretize the device state. In some cases, however,

the analog behavior of the magnetization dynamics can become important, and modeling that behavior will be important for compact models of these devices in circuit design tools. Circuit models for these devices have been developed at various levels of approximation, from the Néel-Brown level to macrospin approximations of the magnetic dynamics.[7–9] Even so, the use of these models in circuit simulator tools can be tricky, as simulators are often not equipped to carry out mathematically-correct stochastic integration routines. Progress in both models and tooling will be an important step in moving stochastic applications of magnetic tunnel junctions from experiment to prototype.

3. ISING MODELS

The connection between MTJs and Ising spins seems immediate; both are two state systems and the magnetic origin of these states makes for a tempting analogy. Indeed, much of the contemporary work in using magnetic tunnel junctions attempts to exploit this analogy, often for the purposes of simulated annealing of a spin system with some energy functional curated to solve a problem of interest.[10,11]

Although MTJs can clearly represent the *states* of Ising systems, coupling MTJs is a more difficult proposition. Although it is possible to imagine a dipolar coupling between devices, the locality (on the order of 300 nm) of that physical interaction limits the construction of arbitrary energy functionals and puts strong constraints on device design. More general Ising machines are more likely to be constructed using electrical coupling.

The leading efforts on digital coupling of SMTJs for Ising model emulation are based on so-called p-bits. Originally introduced in the context of reversible logic,[12] p-bits are proposed as a computational element that bridges the gap between deterministic bits and qubits. Since their introduction, p-bits have seen extensive development in combinatorial optimization problems. One of the first nontrivial demonstrations of an experimental p-bit system demonstrated that a p-bit computer could factor semiprimes up to approximately 8 bits.[13]

FPGA- and ASIC-based coupling of p-bit devices described in the previous section has proven to be extremely versatile for a wide range of computational problems.[14] However, it is tempting to ask whether we can connect these devices with direct electrical signals rather than digital control, the latter of which often comes with non-negligible area and energy consumption overheads.

3.1 Two-device coupling

One mechanism that was recognized early on for coupling SMTJs is through stochastic resonance. A series of papers by Mizrahi *et al.* investigate stochastic resonance of SMTJs in simulation, theory, and experiment.[15–17] As a computational platform, however, spin-torque oscillators continue to be a favored spintronic platform for implementing coupled oscillator systems.[18]

Perhaps the simplest form of analog circuit coupling between two SMTJs is investigated experimentally by Talatchian *et al.*[19] In their system, depicted in Fig. 2(a), the total voltage across the circuit is determined by the parallel resistance of two SMTJs in a voltage divider with a static resistor R. In the limiting case where the voltage source and series resistor become a current source, a change in conductance of one device maximally changes the current through the other device, thereby maximally changing the second device's escape probability into a different state. In this regime, the correlation between devices is maximal; clearly in the limit that the series resistor goes to zero and becomes a voltage source, there is no coupling between the devices at all.

3.2 n-device Ising chains

The basic premise of the coupling mechanism from Talatchian *et al.* is that the two devices share the same pool of current. In the present paper, we envision extending the system to beyond two devices by constructing a chain as in Fig. 2(b). Current conservation at the node above each MTJ in the chain leads us to the voltage drop across the MTJ given by

$$V_j = \frac{i_j + V_{j-1} g_{j-1,j} + V_{j+1} g_{j,j+1}}{G_j + g_{j-1,j} + g_{j,j+1}} \tag{2}$$

where G_j is the MTJ's conductance, $g_{i,j}$ is the conductance of the resistor connecting devices i and j, and i_j the current source on column j. Although we will restrict ourselves to an infinite chain topology for the sake of brevity, note that we could have connected the node above MTJ j to any number of other nodes and we simply

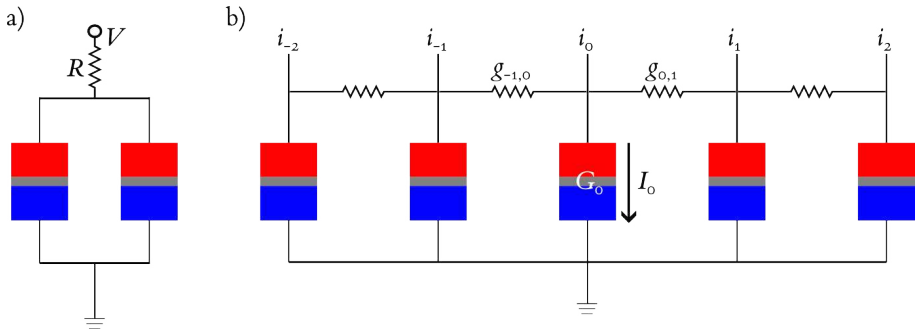

Figure 2. a) The experimental setup investigated by Talatchian *et al.*; the devices couple by virtue of one device's conductance changing the current through the other device, especially in the current source limit. b) A chain of coupled devices that work on the same principle as Fig. 2(a); solving for the transition probabilities indicates that this system mimics a 1D Ising chain.

would have added more terms to the top and bottom of Eq. (2). Whatever the topology, this implicit equation for V_j can be viewed as a matrix equation $\vec{V} = M\vec{V} + \vec{U}$; letting $\Gamma_j = G_j + \sum_{\langle jk \rangle} g_{jk}$ where the sum is over nodes k connected to j, we have $U_j = i_j/\Gamma_j$ and

$$M = \begin{bmatrix} \ddots & \ddots & \ddots & & & \\ & \frac{g_{j-2}}{\Gamma_{j-1}} & 0 & \frac{g_j}{\Gamma_{j-1}} & & \\ & & \frac{g_{j-1}}{\Gamma_j} & 0 & \frac{g_{j+1}}{\Gamma_j} & \\ & & & \frac{g_j}{\Gamma_{j+1}} & 0 & \frac{g_{j+2}}{\Gamma_{j+1}} \\ & & & & \ddots & \ddots & \ddots \end{bmatrix} \tag{3}$$

with zeros off the tridiagonal; we have given M for the chain topology but in general the j^{th} row would contain nonzero terms $\{g_i/\Gamma_j\}_{\langle ij \rangle}$. The Neumann series $(1-M)^{-1} = \sum_{j=0}^{\infty} M^j$ always converges and we can write down a series expansion for $\vec{V} = (1-M)^{-1}\vec{U}$; at leading order in g/Γ, we have

$$V_j = U_j + \sum_{jk} \frac{g_k}{\Gamma_j} U_k + O(g^2/\Gamma^2). \tag{4}$$

Note that g/Γ is always small if the degree of connectivity is high. However, we will make the further assumption that the MTJs are far more conductive than the transverse resistors, $G > g\mathcal{D}$ in general where $\mathcal{D}$ is the degree of connectivity. This localizes the interactions; when $g \to 0$, the SMTJs are isolated, and when g becomes small but nonzero they can be affected just a bit by their neighbors. In this limit, we have (with $R_j = 1/G_j$)

$$\frac{1}{\Gamma_j} = R_j \left(1 - \frac{\sum_{jk} g_k}{G_j} + O(g^2/G^2)\right). \tag{5}$$

Keeping terms only up to linear order in g/G and dividing both sides of Eq. (4) by R_j to get the current I_j through the device leads us to conclude that

$$I_j = i_j \left(1 - R_j \sum_{\langle jk \rangle} g_{jk}\right) + \sum_{\langle jk \rangle} g_{jk} i_k R_k. \tag{6}$$

Now suppose the MTJs are all identical, and have $R_j = R_0 + \rho s_j$ where $s_j = \pm 1$ is the MTJ's state. Substituting these in, we have

$$I_j = i_j + R_0 \sum_{\langle jk \rangle} g_{jk}(i_k - i_j) + i_j s_j \rho \sum_{\langle jk \rangle} g_{jk} + \rho \sum_{\langle jk \rangle} g_{jk} s_k i_k. \tag{7}$$

Now that we know the current through each MTJ, what can we say about its transition rate out of its current state? Recall Eq. (1) and let us further assume – as is the case with many real devices – that we require a current bias of i_* to achieve equal probabilities of the two states. Then the switch probability of device j out of state s_j is

$$\varphi_{j,s_j} = \varphi_0 \exp\left(-\frac{\Delta}{kT}\frac{s_j(I_j - i_*)}{I_c}\right) \tag{8}$$

where we will take from now on $\beta := \Delta/(kTI_c)$ for brevity. Then

$$\varphi_{j,s_j} = \varphi_0 \exp\left[-\beta\left(s_j(i_j - i_*) + s_j R_0 \sum_{\langle jk\rangle} g_{jk}(i_k - i_j) + i_j\rho\sum_{\langle jk\rangle} g_{jk} + \rho\sum_{\langle jk\rangle} s_j s_k g_{jk} i_k\right)\right] \tag{9}$$

$$= \tilde{\varphi}_0 \exp\left[-\beta\left(s_j(i_j - i_*) + s_j R_0 \sum_{\langle jk\rangle} g_{jk}(i_k - i_j) + \rho\sum_{\langle jk\rangle} s_j s_k g_{jk} i_k\right)\right] \tag{10}$$

where in the second equation we absorb the constant term $i_j\rho\sum_{\langle jk\rangle} g_{jk}$ into $\tilde{\varphi}_0$.

Each SMTJ has exponentially distributed switching time with rate parameter φ_{j,s_j}. In that sense, the probability that a device switches in a time τ is just $\varphi_{j,s_j}\tau$. Ignoring the overall prefactor, then, we would like to interpret this switching probability to be of the form $\exp(-\beta\Delta E)$ where ΔE is the change in energy that would result from the changed state – this would roughly map the dynamics to a Metropolis-Hastings algorithm. Applying this mapping to Eq. (10), however, we find two problems.

First, this system is not Hermitian, because the change in the interaction energy between two of the devices would be different depending on which one switches, due to the i_k factor in the $s_j s_k$ sum. If we want a Hermitian system with a well-defined energy landscape, we must take all i_j to be some constant i_0. Note that this also makes the R_0 sum vanish, leaving us with

$$\varphi_{j,s_j} = \tilde{\varphi}_0 \exp\left[-\beta\left(s_j(i_0 - i_*) + \rho i_0 \sum_{\langle jk\rangle} g_{jk} s_j s_k\right)\right] \tag{11}$$

Second, we note that as $\rho g \sim g/G$ is a small parameter, the energy will be dominated by the $s_j(i_0 - i_*)$ term. However, if we simply choose $i_0 = i_*$, then the physics is dominated by the couplings,

$$\varphi_{j,s_j} = \tilde{\varphi}_0 \exp\left(-\frac{\Delta}{kT}\frac{i_*}{I_c}\sum_{\langle jk\rangle} J_{jk} s_j s_k\right) \tag{12}$$

where we have defined $J_{jk} = \rho g_{jk}$. Note that as the change in energy has sign $\mathrm{sgn}(s_j s_k)$ for each coupling, the corresponding effective energy function for this system is

$$E_{\text{eff}} = -\frac{i_*}{2I_c}\sum_{\langle jk\rangle} J_{jk} s_j s_k \tag{13}$$

where the sum is over all connected columns. One can choose a ferromagnetic or antiferromagnetic interaction simply by changing the sign of i_*, that is, by flipping the polarity of the MTJs in the circuit.

Using a crossbar architecture, one could theoretically construct all-to-all connectivity between a set of SMTJs and construct a J_{jk}-programmable Ising model in this way. However, several technical problems remain. One is that in most problems of interest, one would want to anneal the effective temperature of the system. This could be done in theory by increasing the magnitude of the applied currents while correspondingly using an applied field to change i_* at the same time, increasing the effective inverse temperature without reintroducing the $(i_0 - i_*)$ effective field. In practice, of course, this would be very difficult, and even without annealing one would need very well-matched devices to maintain hermiticity while avoiding the introduction of a dynamics-dominating effective field. How to read out the state of such a system is yet another challenge.

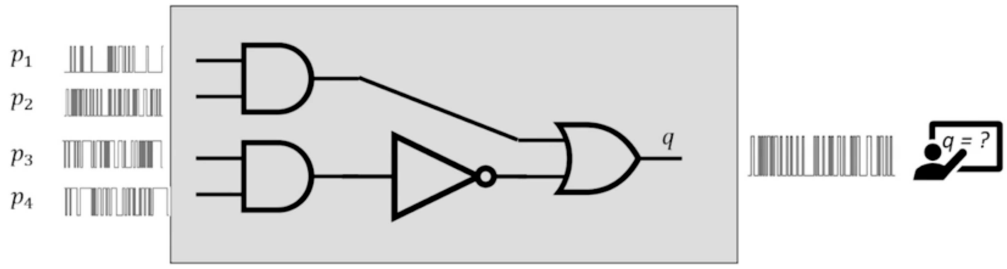

Figure 3. The basic premise of a stochastic computing computation. Bitstreams with some unknown probabilities p_j to be in their logical-true state are input to the computer (from the left, in the figure). The inputs are generally regarded as uncorrelated but need not be; the machine accepts, in general, samples from an unknown multivariate distribution. A network of logic gates induces some transformation on these input bitstreams to produce a (possibly multivariate) output distribution, manifest in some output bitstream(s). In order to read the output of the machine, one must sample from this bitstream and attempt to guess some statistic q (often the expected value of the bitstream) from a finite population of bit samples. The longer one waits to report q, the more precisely one can report the output of the computation.

As the mean SMTJ resistance increases toward the coupling resistance, more terms become relevant in our various series expansions, and couplings between next- and next-next-nearest neighbors begins to appear. This too could be of interest, though such couplings would be less programmable. We leave further exploration of this system to future research, noting that a similar idea of a crossbar architecture coupling SMTJs to achieve a restricted Boltzmann machine architecture is proposed by Phan *et al.* in a recent paper.[20]

4. STOCHASTIC COMPUTING

In the previous section, the stochasticity of devices was used to mimic physical thermodynamic processes, simulating Hamiltonian systems whose energy functions were arranged to encode a problem of interest. In this section, we instead use stochastic dynamics to encode numerical values directly, and operate on the generated random bitstreams to construct arithmetic computations. The idea of using random bitstreams as numerical scalars in a computational circuit is called *stochastic computing* and traces its origins back to von Neumann.[21]

Stochastic computing (demonstrated conceptually in Fig. 3) has received much attention for the elegance of its simple circuits, promise of low-power and low-area operation, and as an ideal application space for stochastic nanodevices. Typically the inputs to a stochastic computer are serially-injected random bitstreams associated with some statistical distribution $\mathcal{I}$. The output of the computation can be regarded as some unknown distribution $\mathcal{D}$ with a functional dependence f on the input distribution, $\mathcal{D} = f(\mathcal{I})$, with f determined by the logic circuits comprising the stochastic computer. In practice, one can never determine $\mathcal{D}$ with perfect certainty, but must instead estimate $\mathcal{D}$ (or, more typically, the expected value of $\mathcal{D}$) by sampling N random bits from the output bitstream of the computer. One can spend additional time and energy to increase the precision and accuracy of the output by increasing N.

4.1 Encoding and efficiency of stochastic computing

Usually numerical information is encoded in the expectation value of a bitstream. One way to do this (called the unipolar encoding) is to directly associate to any bitstream of 0s and 1s the expected value of the bitstream distribution, limiting the representable space to the unit interval – note that this is equivalent to letting the stream be represented by the probability p of finding a 1. The so-called bipolar encoding, by contrast, associates to that same bitstream the value $2\langle p\rangle - 1$, allowing the designer to access values $[-1, 1]$. Other encodings exist and are sometimes used in niche applications. Regardless of the particular mapping, however, these encodings are all examples of unary number systems. Notice that a length-N bitstream in this context can encode only $N + 1$ distinct values, since the order of the bits carries no information. In an information theoretic sense, then, a length-N bitstream carries only $\log_2 N$ bits of information—a significant degradation over the exponential compression of usual binary numbers. This degradation has led to the observation that stochastic computing is

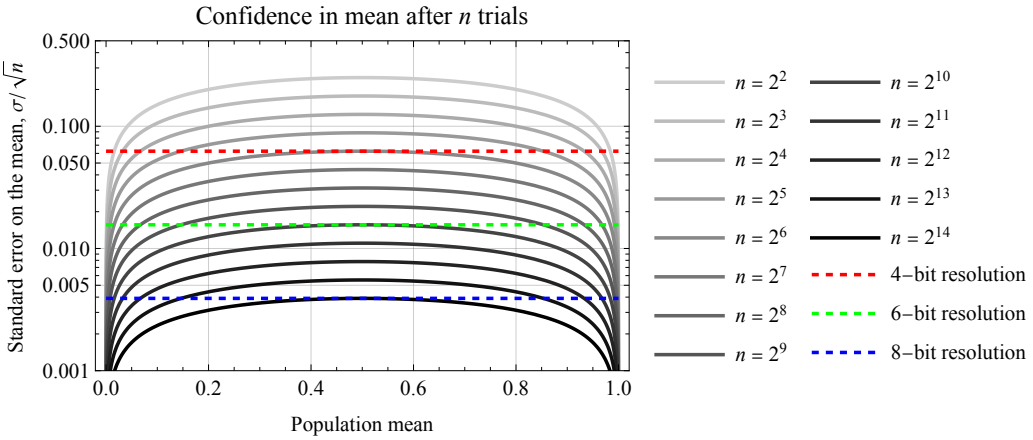

Figure 4. The standard error on the mean when estimating the expected value of some bitstream output from a stochastic computation (*e.g.* estimating q in Fig. 3). Note the logarithmic scale.Note the logarithmic scale. In order to reliably distinguish 16 different outputs with 1σ confidence, one must collect 64 bits; to distinguish 64 levels, one must collect 1024 bits; to distinguish 256 levels, one must collect 16384 bits.

energy- and time-efficient only for small N, as at large N there is almost certainly a binary-coded circuit that can carry out the same computation with far fewer cycles.

Exactly how small N must be for efficient stochastic computing is a question of some debate. From an abstract computational standpoint, Manohar has shown that stochastic computing is almost never optimal.[22] However, when concrete considerations about circuits and architecture are considered, stochastic computing does have the capacity to outperform binary computing for certain tasks, for bitstream lengths up about 256.[23]

In terms of bitstream length, stochastic computing must fight against not only a low-density encoding but also statistical uncertainty in the mean. The latter goes down only as $N^{-1/2}$. Because of uncertainty in the mean of a stochastic bitstream, the effective resolution of a bitstream is less than $\log_2 N$ bits. Figure 4 shows the standard error on the mean, parametrically for different N, as a function of the underlying distributional mean. To reach an effective 8-bit resolution – where you can be reasonably sure that a particular output is distinct from values $\pm 2^{-8}$ to each side – requires the collection of 2^{14} samples from the bitstream, or over 16000 clock cycles. Meanwhile 64 samples, a perhaps more reasonable runtime, can achieve effective 4-bit resolution.

This analysis suggests that stochastic computing is best suited to computational problems where precision can be low and where heavy reliance on dataflow structures can make the most use of stochastic computing's architectural advantages. Many modern machine learning workloads fit these criteria. This observation has motivated an exploration of stochastic computing systems based around deep neural network architectures, which have long been capable of performing low-precision inference and are now capable of training at low precision on a variety of architectures.[24–26]

4.2 Randomness and pseduorandomness in stochastic computing

Randomness sources have traditionally been difficult to integrate in complementary metal-oxide semiconductor (CMOS) processes, and pseudorandom algorithms for generating high-quality randomness tend to be bulky and energy-inefficient when realized in hardware. As a result, traditional stochastic computing work has focused on the use of linear feedback shift registers (LFSRs) for generating cheap but low-quality pseudorandom bitstreams.

For stochastic computing systems with large fan-in, however, low-period random generators become problematic. Most stochastic computing operations rely on input bitstreams being statistically uncorrelated, or at best correlated in some precisely known way.[27] When the number of inputs to a stochastic circuit starts to approach the period of the underlying generator, collisions between perfectly correlated bitstreams become increasingly likely, leading to computational inaccuracies. To manage these problems, a variety of schemes have been introduced; one of the most energy efficient is the use of approximate parallel counters to translate out of and back into the stochastic regime.

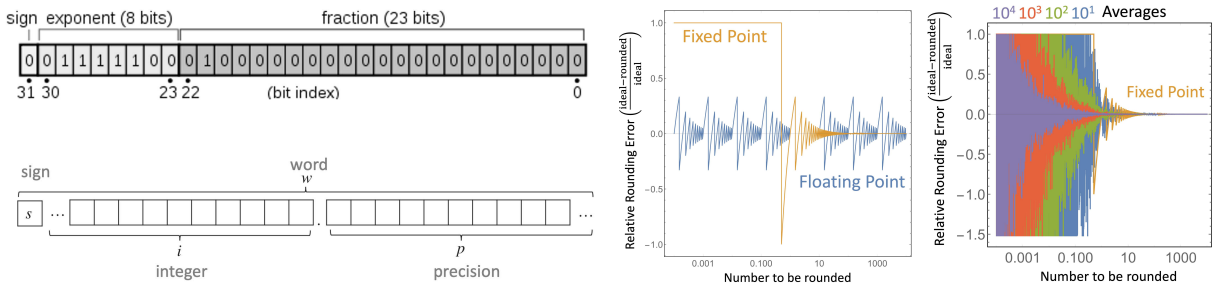

Figure 5. Top left: floating point number representation. Bottom left: fixed-point number representation. Center: The relative error induced by the round-nearest operation in fixed and floating point number schemes. Fixed point relative rounding error becomes 100 % when the number to be represented is under the precision floor of the encoding. Right: below the precision floor of the fixed point coding, the expected value of the relative error can be suppressed by using stochastic rounding; more samples increases the precision.

The possibility of using SMTJs to introduce truly random bitstreams to stochastic computers releases a significant constraint on stochastic circuit design by removing so-called *edge correlations* from a stochastic circuit (c.f. *graph correlations* built up by merging the bitstreams in their computational flow, language introduced in Daniels *et al.*[28]). Initial forays into the use of SMTJs for stochastic computing focused on basic computational primitives and energy efficient circuit designs. The pre-charge sense amplifier, a key circuit for MTJ state detection, has also played an important role in energy efficient SMTJ devices.[28]

As much of the SMTJ community consists of physicists and materials-oriented researchers rather than electrical engineers, the use of SMTJs in traditional stochastic computing applications such as image filtering and other streaming computations on sensor data has been limited, since the former communities have less familiarity with these algorithms and use-cases. By contrast, neuromorphic computing has become widely studied across fields, and so it is no surprise that SMTJs have found use in this new cross-disciplinary area.

One study has used the stochastic bitstreams of SMTJs in a population coding context.[29] Population coding is a multi-channel encoding often used to code a distribution over a discrete domain (the channels) rather than a single scalar value. In Mizrahi *et al.*, SMTJs are envisioned as an array of analog-to-stochastic sensors that convey a brightness versus angle profile for a robot's perceptive operation.[29] They use this coding in a multilayer perceptron using weights stored in MRAM cells and show that the energy efficiency of such a system is considerably lower than an analogous CMOS-only design.

Population-coded perception systems are heavily bio-inspired; Daniels *et al.*,[28] by contrast, takes an algebraic approach and uses traditional stochastic computing primitives to construct a deep neural network directly. Unlike the DNN applications described above, the SMTJ-based work takes advantage of the truly random bitstreams to dramatically simplify the neuron model, reducing a neuron to a single CMOS OR-gate. With some loss of accuracy, the network described there performs 2× to 10× more efficiently than CMOS stochastic networks (which themselves are known to be more energy efficient than traditional methods such as graphical processing units).

5. LOW-PRECISION DEEP NEURAL NETWORKS

One of the oldest and most obvious applications for the stochastic behavior of magnetic tunnel junctions is merely as random bit generators, rather than as computational or encoding elements. Though much recent attention has been given to SMTJs in this context, the stochastic write behavior of stable MTJs has also been exploited for this purpose.[30]

As others have observed,[31] accessibility to massively parallel random bit generators has the potential to dramatically change the landscape of how we do scientific and numerical computing. From Monte Carlo simulations to the integration of stochastic differential equations, random number generation is often a computational bottleneck in modern scientific computing workloads.

One of the most interesting uses of these random bits has been to enable low-precision operations. In high dimensional ODEs, for instance, low-precision has been essential for practical implementation, but it turns out that extra precision can be hidden in the least significant bit through stochastic rounding.[32] Stochastic rounding (Fig. 5) is the practice of rounding up the excess precision of a numerical result (often the result of a multiplication) with probability given through the digits to be truncated. As a simple example, imagine rounding 14.327 to two decimal places. The most well-known rounding scheme, round-nearest, would round this to 14.33. Other deterministic schemes exist like round-even, round-odd, round-toward-zero, and so on. Stochastic rounding would round 14.327 to 14.33 with probability 0.7, and would round to 14.32 with probability 0.3.

Note that the expectation of a number rounded stochastically is equal to the arbitrarily precise value of the roundee. Therefore if sufficiently many (say N) values are stochastically rounded and then summed – as in, for instance, a dot-product operation – then we roughly expect a $1/\sqrt{N}$ convergence to the "true" underlying value of the operation, up to whatever is representable in the output coding. As a contrived example, consider the dot product of two N-vectors each of the form $\vec{x} = (1/2, 1/2, 1/2, \cdots)$, and suppose we have only a single fractional bit available. A deterministic rounding scheme must round each product of $1/4$ to either 0 or 1, so that the dot product results in either 0 or N. Stochastic rounding has expected output $N/4$, plus or minus some error on the mean of order $O(1/\sqrt{N})$. While in many applications a deterministic result is required, the stochastic computing result is certainly closer to the correct answer even in the presence of the stochastic error term.

In the past few years, it has become well-understood that stochastic rounding is a crucial operation for training deep neural networks at low precision.[33] Since the initial demonstration that stochastic rounding could maintain the learning dynamics of a network even after the error propagation dropped beyond the representation floor, many other algorithms have since emerged to push network training to fewer and fewer bits, not only in the weights and gradients of the network but in the errors and activations as well.[34] For these algorithms to make sense in high-performance artificial intelligence hardware, a ready source of random bits is required; magnetic tunnel junction schemes are well-poised to fit this niche, and it has been shown that SMTJ-based generators can be more energy efficient per random bit even than the smallest LFSR circuits.[28]

Extreme cases of low-precision networks include the binary and ternary neural networks, where weights are either -1 or 1 in the former case and -1, 0, or 1 in the latter. These networks require special training algorithms, but are usually sampled stochastically from real-valued "latent" weights. As a first step toward systems of this type based on magnetic tunnel junctions, several groups have tested experimental prototype networks using MTJ crossbars. Zhou *et al.* showed in a small demonstration that an experimentally based MTJ binary neural network could in fact achieve functionality.[35] A subset of the present authors demonstrated an inference engine for the Wine dataset and found that the mapping between software and hardware systems of this type can be made highly nontrivial even in online learning.[36] Recently, a very large network based on an MTJ architecture was demonstrated by Samsung.[37] However, direct on-line training of such systems has been elusive due to the requirement of latent weights described above. Overcoming this challenge will be an important next step for the field.

6. ACKNOWLEDGEMENTS

AM, SG, TA, LP, and DL acknowledge support by the National Science Foundation under Grant No. CCF-2121957.

REFERENCES

[1] Berggren, K., Xia, Q., Likharev, K. K., Strukov, D. B., Jiang, H., Mikolajick, T., Querlioz, D., Salinga, M., Erickson, J. R., Pi, S., Xiong, F., Lin, P., Li, C., Chen, Y., Xiong, S., Hoskins, B. D., Daniels, M. W., Madhavan, A., Liddle, J. A., McClelland, J. J., Yang, Y., Rupp, J., Nonnenmann, S. S., Cheng, K.-T., Gong, N., Lastras-Montaño, M. A., Talin, A. A., Salleo, A., Shastri, B. J., de Lima, T. F., Prucnal, P., Tait, A. N., Shen, Y., Meng, H., Roques-Carmes, C., Cheng, Z., Bhaskaran, H., Jariwala, D., Wang, H., Shainline, J. M., Segall, K., Yang, J. J., Roy, K., Datta, S., and Raychowdhury, A., "Roadmap on emerging hardware and technology for machine learning," *Nanotechnology* **32**, 012002 (Oct. 2020).

[2] Christensen, D. V., Dittmann, R., Linares-Barranco, B., Sebastian, A., Le Gallo, M., Redaelli, A., Slesazeck, S., Mikolajick, T., Spiga, S., Menzel, S., Valov, I., Milano, G., Ricciardi, C., Liang, S.-J., Miao, F., Lanza, M., Quill, T. J., Keene, S. T., Salleo, A., Grollier, J., Marković, D., Mizrahi, A., Yao, P., Yang, J. J., Indiveri, G., Strachan, J. P., Datta, S., Vianello, E., Valentian, A., Feldmann, J., Li, X., Pernice, W. H. P., Bhaskaran, H., Furber, S., Neftci, E., Scherr, F., Maass, W., Ramaswamy, S., Tapson, J., Panda, P., Kim, Y., Tanaka, G., Thorpe, S., Bartolozzi, C., Cleland, T. A., Posch, C., Liu, S., Panuccio, G., Mahmud, M., Mazumder, A. N., Hosseini, M., Mohsenin, T., Donati, E., Tolu, S., Galeazzi, R., Christensen, M. E., Holm, S., Ielmini, D., and Pryds, N., "2022 roadmap on neuromorphic computing and engineering," *Neuromorph. Comput. Eng.* **2**, 022501 (June 2022).

[3] Grollier, J., Querlioz, D., Camsari, K. Y., Everschor-Sitte, K., Fukami, S., and Stiles, M. D., "Neuromorphic spintronics," *Nature Electronics* **3**, 360–370 (July 2020).

[4] Locatelli, N., Cros, V., and Grollier, J., "Spin-torque building blocks," *Nature materials* **13**(1), 11–20 (2014).

[5] Brown, W. F., "Thermal Fluctuations of a Single-Domain Particle," *Phys. Rev.* **130**, 1677–1686 (June 1963).

[6] Fitzhugh, R., "Statistical properties of the asymmetric random telegraph signal, with applications to single-channel analysis," *Mathematical Biosciences* **64**, 75–89 (May 1983).

[7] Torunbalci, M. M., Upadhyaya, P., Bhave, S. A., and Camsari, K. Y., "Modular Compact Modeling of MTJ Devices," *IEEE Transactions on Electron Devices* **65**, 4628–4634 (Oct. 2018).

[8] Yang, X., Zhang, Y., Zhang, Y., and Wang, P., "A Universal Compact Model for Spin-Transfer Torque-Driven Magnetization Switching in Magnetic Tunnel Junction," *IEEE Trans. Electron Devices* **69**, 6453–6458 (Nov. 2022).

[9] Vincent, A. F., Locatelli, N., Klein, J. O., Zhao, W. S., Galdin-Retailleau, S., and Querlioz, D., "Analytical Macrospin Modeling of the Stochastic Switching Time of Spin-Transfer Torque Devices," *Ieee T Electron Dev* **62**, 164–170 (Jan. 2015).

[10] Aadit, N. A., Grimaldi, A., Carpentieri, M., Theogarajan, L., Martinis, J. M., Finocchio, G., and Camsari, K. Y., "Massively parallel probabilistic computing with sparse Ising machines," *Nat Electron* **5**, 460–468 (July 2022).

[11] Aadit, N. A., Mohseni, M., and Camsari, K. Y., "Accelerating Adaptive Parallel Tempering with FPGA-based p-bits," in [*2023 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*], 1–2, IEEE, Kyoto, Japan (June 2023).

[12] Camsari, K. Y., Faria, R., Sutton, B. M., and Datta, S., "Stochastic p-Bits for Invertible Logic," *Phys. Rev. X* **7**, 031014 (July 2017).

[13] Borders, W. A., Pervaiz, A. Z., Fukami, S., Camsari, K. Y., Ohno, H., and Datta, S., "Integer factorization using stochastic magnetic tunnel junctions," *Nature* **573**, 390–393 (Sept. 2019).

[14] Chowdhury, S., Grimaldi, A., Aadit, N. A., Niazi, S., Mohseni, M., Kanai, S., Ohno, H., Fukami, S., Theogarajan, L., Finocchio, G., Datta, S., and Camsari, K. Y., "A full-stack view of probabilistic computing with p-bits: Devices, architectures and algorithms," (Feb. 2023).

[15] Mizrahi, A., Locatelli, N., Matsumoto, R., Fukushima, A., Kubota, H., Yuasa, S., Cros, V., Kim, J.-V., Grollier, J., and Querlioz, D., "Magnetic Stochastic Oscillators: Noise-Induced Synchronization to Underthreshold Excitation and Comprehensive Compact Model," *IEEE Trans. Magn.* **51**, 1401404 (Nov. 2015).

[16] Mizrahi, A., Locatelli, N., Grollier, J., and Querlioz, D., "Synchronization of electrically coupled stochastic magnetic oscillators induced by thermal and electrical noise," *Phys. Rev. B* **94**, 054419 (Aug. 2016).

[17] Accioly, A., Locatelli, N., Mizrahi, A., Querlioz, D., Pereira, L. G., Grollier, J., and Kim, J.-V., "Role of spin-transfer torques on synchronization and resonance phenomena in stochastic magnetic oscillators," *Journal of Applied Physics* **120**, 093902 (Sept. 2016).

[18] Romera, M., Talatchian, P., Tsunegi, S., Araujo, F. A., Cros, V., Bortolotti, P., Trastoy, J., Yakushiji, K., Fukushima, A., Kubota, H., Yuasa, S., Ernoult, M., Vodenicarevic, D., Hirtzlin, T., Locatelli, N., Querlioz, D., and Grollier, J., "Vowel recognition with four coupled spin-torque nano-oscillators," *Nature* **563**, 230–+ (Nov. 2018).

[19] Talatchian, P., Daniels, M. W., Madhavan, A., Pufall, M. R., Jué, E., Rippard, W. H., McClelland, J. J., and Stiles, M. D., "Mutual control of stochastic switching for two electrically coupled superparamagnetic tunnel junctions," *Phys. Rev. B* **104**, 054427 (Aug. 2021).

[20] Phan, N.-T., Soumah, L., Sidi El Valli, A., Hutin, L., Anghel, L., Ebels, U., and Talatchian, P., "Electrical Coupling of Perpendicular Superparamagnetic Tunnel Junctions for Probabilistic Computing," in [*Proceedings of the 17th ACM International Symposium on Nanoscale Architectures*], 1–6, ACM, Virtual OR USA (Dec. 2022).

[21] Neumann, J. V., "Probabilistic Logics and the Synthesis of Reliable Organisms From Unreliable Components," in [*Automata Studies. (AM-34)*], Shannon, C. E. and McCarthy, J., eds., 43–98, Princeton University Press (Dec. 1956).

[22] Manohar, R., "Comparing Stochastic and Deterministic Computing," *IEEE Comput. Arch. Lett.* **14**, 119–122 (July 2015).

[23] Lee, V. T., Alaghi, A., Pamula, R., Sathe, V. S., Ceze, L., and Oskin, M., "Architecture Considerations for Stochastic Computing Accelerators," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **37**, 2277–2289 (Nov. 2018).

[24] Ren, A., Li, Z., Ding, C., Qiu, Q., Wang, Y., Li, J., Qian, X., and Yuan, B., "SC-DCNN: Highly-Scalable Deep Convolutional Neural Network using Stochastic Computing," *SIGPLAN Not.* **52**, 405–418 (May 2017).

[25] Li, J., Yuan, Z., Li, Z., Ding, C., Ren, A., Qiu, Q., Draper, J., and Wang, Y., "Hardware-driven nonlinear activation for stochastic computing based deep convolutional neural networks," in [*2017 International Joint Conference on Neural Networks (IJCNN)*], 1230–1236, IEEE, Anchorage, AK, USA (May 2017).

[26] Li, Z., Li, J., Ren, A., Cai, R., Ding, C., Qian, X., Draper, J., Yuan, B., Tang, J., Qiu, Q., and Wang, Y., "HEIF: Highly Efficient Stochastic Computing-Based Inference Framework for Deep Neural Networks," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **38**, 1543–1556 (Aug. 2019).

[27] Alaghi, A. and Hayes, J. P., "Exploiting correlation in stochastic circuit design," in [*2013 IEEE 31st International Conference on Computer Design (ICCD)*], 39–46, IEEE, Asheville, NC (Oct. 2013).

[28] Daniels, M. W., Madhavan, A., Talatchian, P., Mizrahi, A., and Stiles, M. D., "Energy-Efficient Stochastic Computing with Superparamagnetic Tunnel Junctions," *Phys. Rev. Applied* **13**, 034016 (Mar. 2020).

[29] Mizrahi, A., Hirtzlin, T., Fukushima, A., Kubota, H., Yuasa, S., Grollier, J., and Querlioz, D., "Neural-like computing with populations of superparamagnetic basis functions," *Nat Commun* **9**, 1533 (Apr. 2018).

[30] Fukushima, A., Yakushiji, K., Kubota, H., and Yuasa, S., "Spin dice (physical random number generator using spin torque switching) and its thermal response," in [*2015 IEEE Magnetics Conference (INTERMAG)*], 1–1, IEEE, Beijing (May 2015).

[31] Misra, S., Bland, L. C., Cardwell, S. G., Incorvia, J. A. C., James, C. D., Kent, A. D., Schuman, C. D., Smith, J. D., and Aimone, J. B., "Probabilistic Neural Computing with Stochastic Devices," *Advanced Materials* , 2204569 (Nov. 2022).

[32] Hopkins, M., Mikaitis, M., Lester, D. R., and Furber, S., "Stochastic rounding and reduced-precision fixed-point arithmetic for solving neural ordinary differential equations," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* **378**, 20190052 (Mar. 2020).

[33] Gupta, S., Agrawal, A., Gopalakrishnan, K., and Narayanan, P., "Deep learning with limited numerical precision," in [*Proceedings of the 32nd International Conference on Machine Learning*], Bach, F. and Blei, D., eds., *Proceedings of Machine Learning Research* **37**, 1737–1746, PMLR, Lille, France (July 2015).

[34] Wu, S., Li, G., Chen, F., and Shi, L., "Training and Inference with Integers in Deep Neural Networks," (2018).

[35] Zhou, P., Edwards, A. J., Mancoff, F. B., Houssameddine, D., Aggarwal, S., and Friedman, J. S., "Experimental Demonstration of Neuromorphic Network with STT MTJ Synapses," *arXiv:2112.04749 [cond-mat, physics:physics]* (Dec. 2021).

[36] Goodwill, J. M., Prasad, N., Hoskins, B. D., Daniels, M. W., Madhavan, A., Wan, L., Santos, T. S., Tran, M., Katine, J. A., Braganca, P. M., Stiles, M. D., and McClelland, J. J., "Implementation of a Binary Neural Network on a Passive Array of Magnetic Tunnel Junctions," *Phys. Rev. Applied* **18**, 014039 (July 2022).

[37] Jung, S., Lee, H., Myung, S., Kim, H., Yoon, S. K., Kwon, S.-W., Ju, Y., Kim, M., Yi, W., Han, S., Kwon, B., Seo, B., Lee, K., Koh, G.-H., Lee, K., Song, Y., Choi, C., Ham, D., and Kim, S. J., "A crossbar array of magnetoresistive memory devices for in-memory computing," *Nature* **601**, 211–216 (Jan. 2022).