U.S. DEPARTMENT OF ENERGY | Office of Science

COINFLIPS

A Probabilistic Future for Neuromorphic Computing

Brad Aimone

Center for Computing Research
Sandia National Laboratories

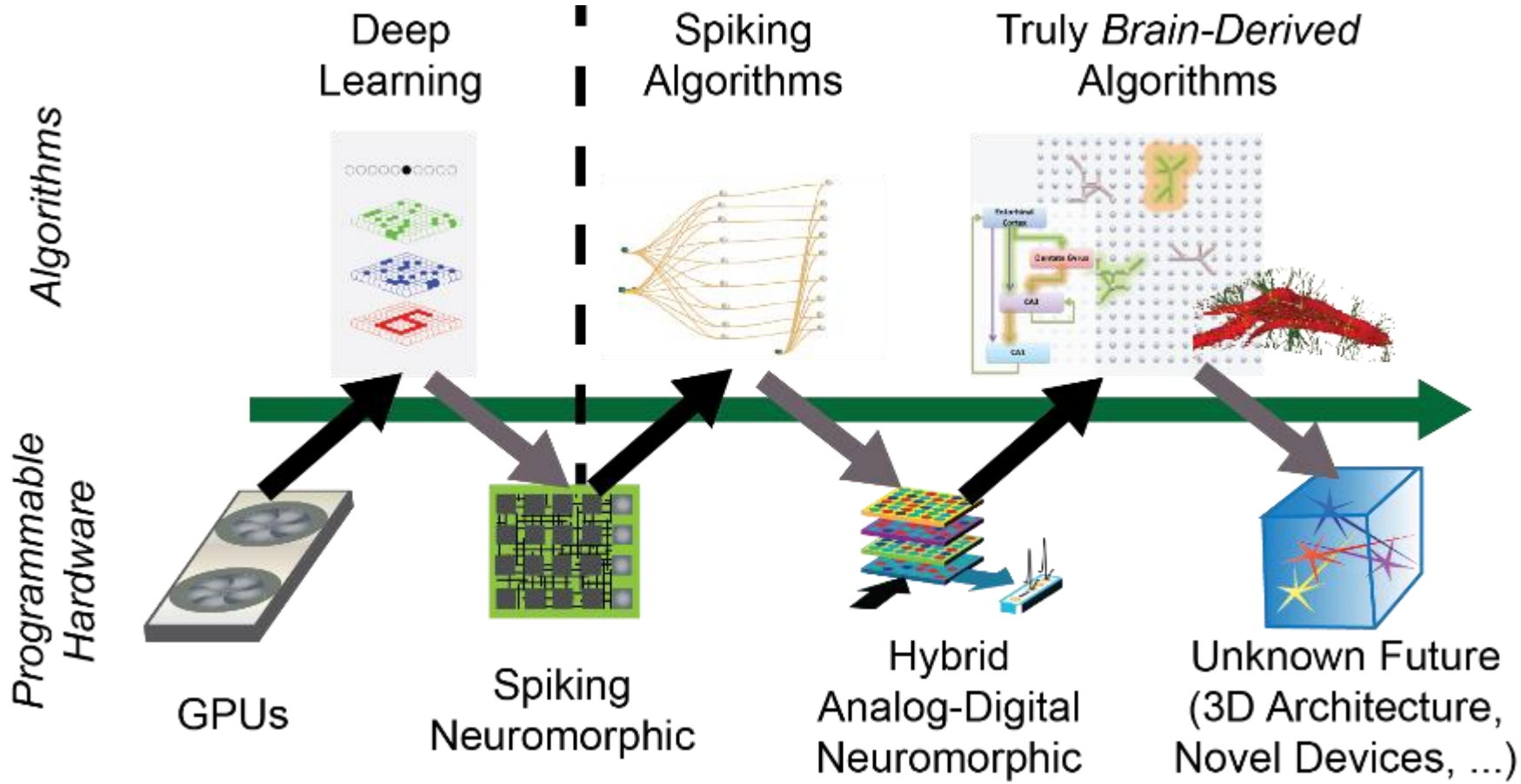
jbaimon@sandia.gov

6/8/2023

Sandia National Laboratories

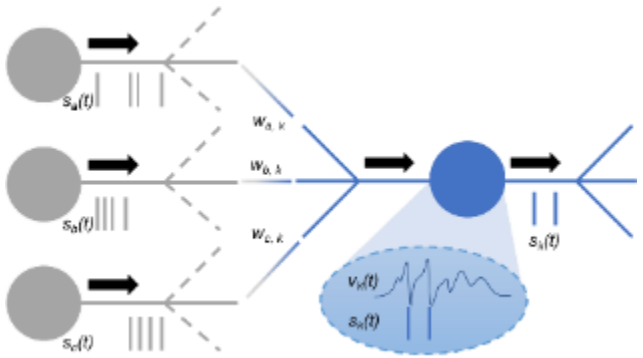
So why the brain?

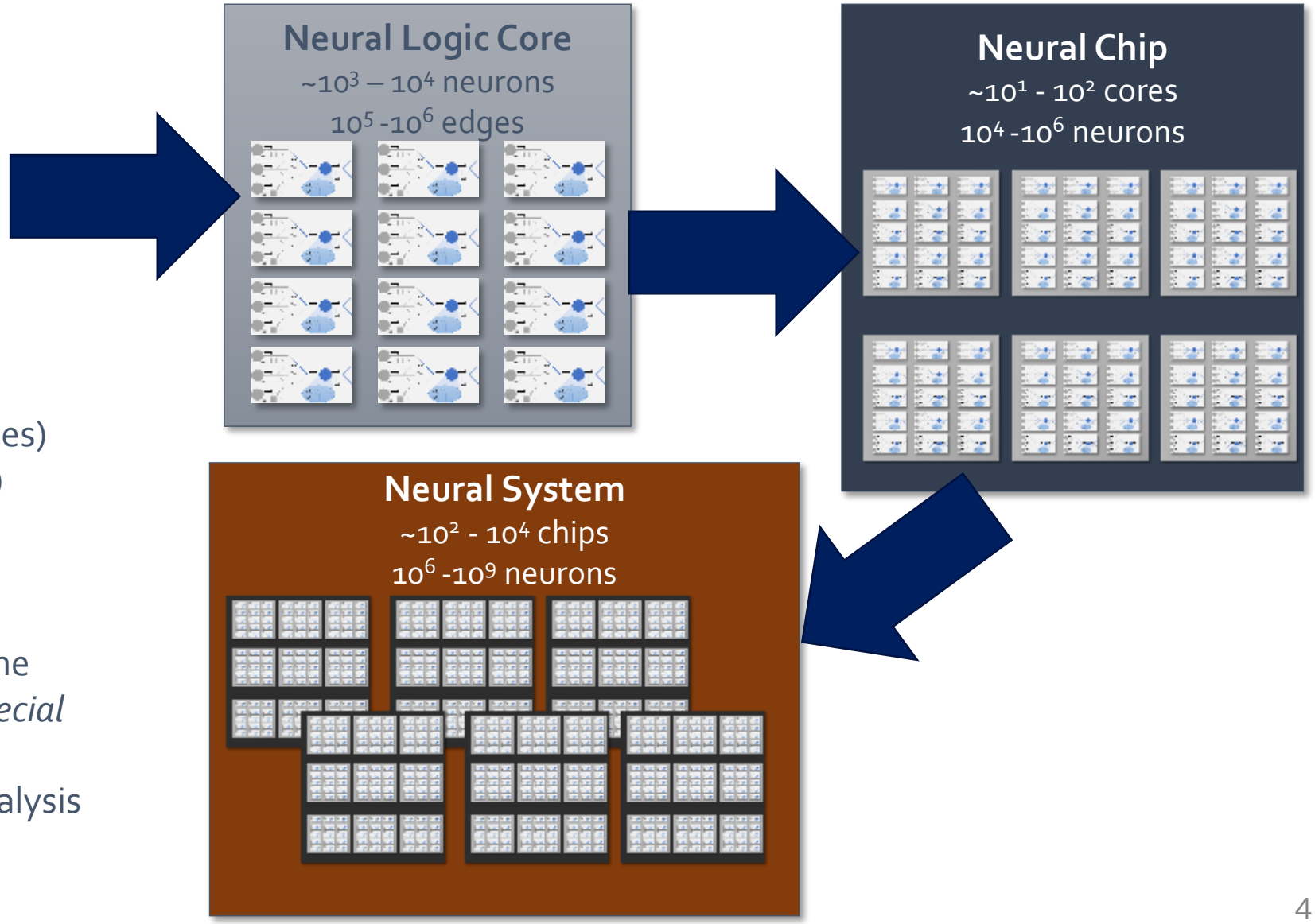
- Energy efficient
- Operationally fast considering slow components
- Data efficient
- Diverse applications
- Robustness



Aimone JB, Advanced Intelligent Systems, 2023

Spiking neuromorphic today: Overview





Computational Primitives:

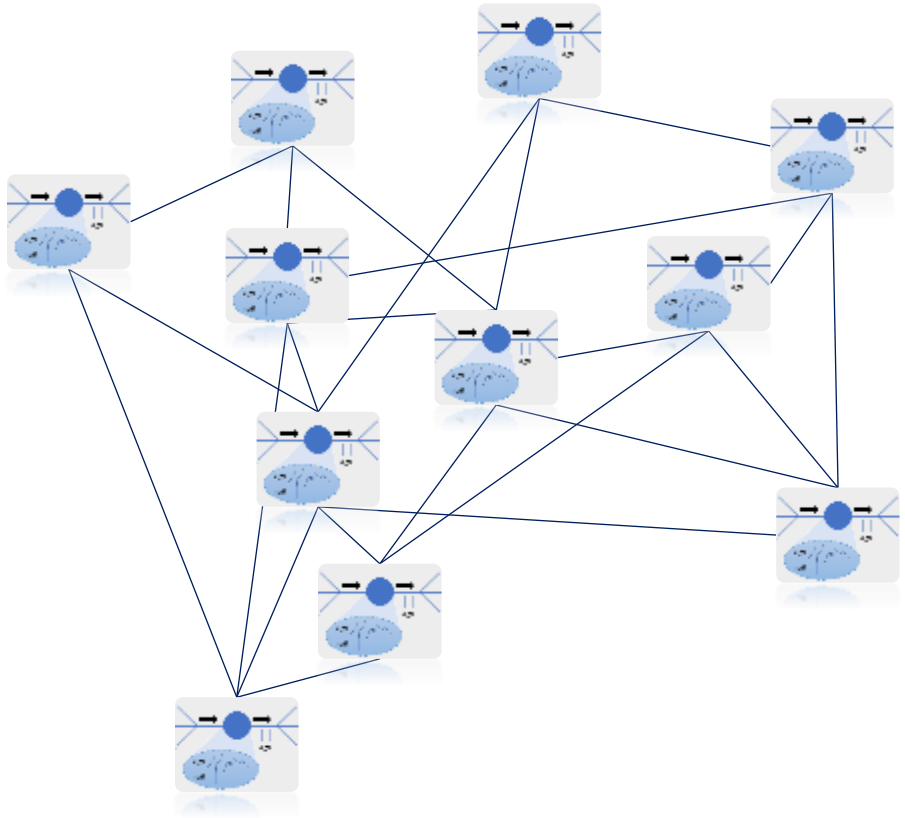
Spiking Neurons (vertices / nodes)

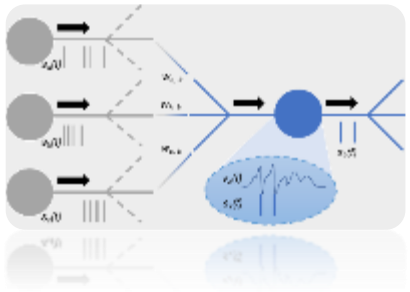
Synapses (connections / edges)

Programmable as arbitrary graphs

- Edges: Directed and weighted
- Nodes: Threshold gate logic + time
- *Artificial neural networks are a special case*
- Programmability, theoretical, analysis and software are open research questions

Neuromorphic hardware jumped ahead of the rest of the stack





Neuromorphic hardware has been built with a "if we build it, neuroscientists will come" hope

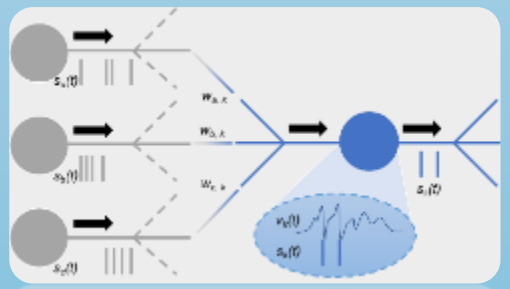
We need

- Driving Applications
- Systems Interface
- Software and Programming Paradigm
- Theoretical Framework

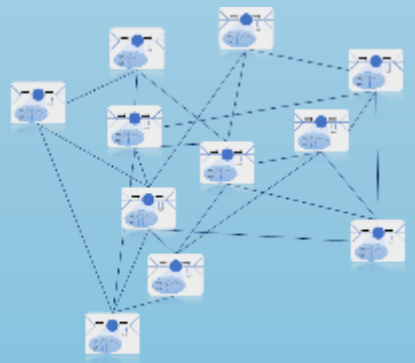
A quick aside: most neuromorphic hardware is *not* designed for artificial neural networks

COINFLIPS

Neuromorphic Hardware



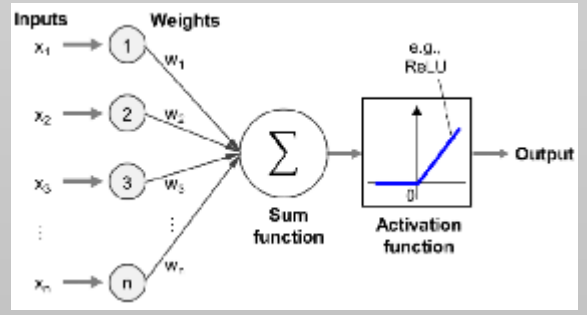
Spiking neurons



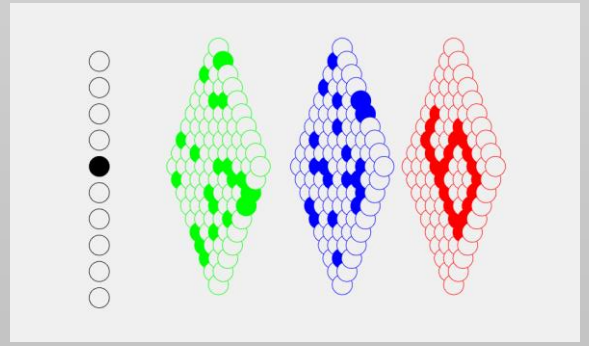
Arbitrary connectivity

- Continual learning integrated into operation
- Inherently temporal
- Dynamical tasks?

Artificial Neural Networks



Continuous neurons



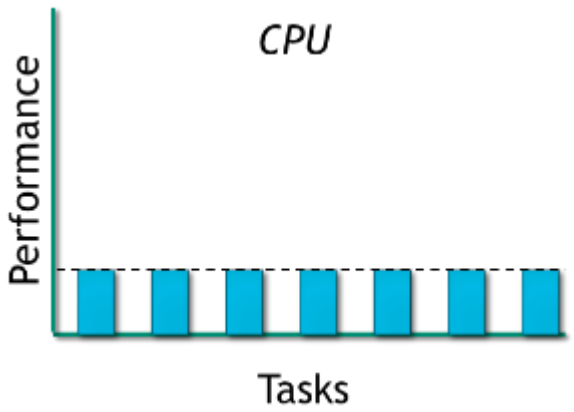
Linear algebra-like networks

- Distinct training and inference modes
- Time is largely avoided
- Computer vision and natural language processing

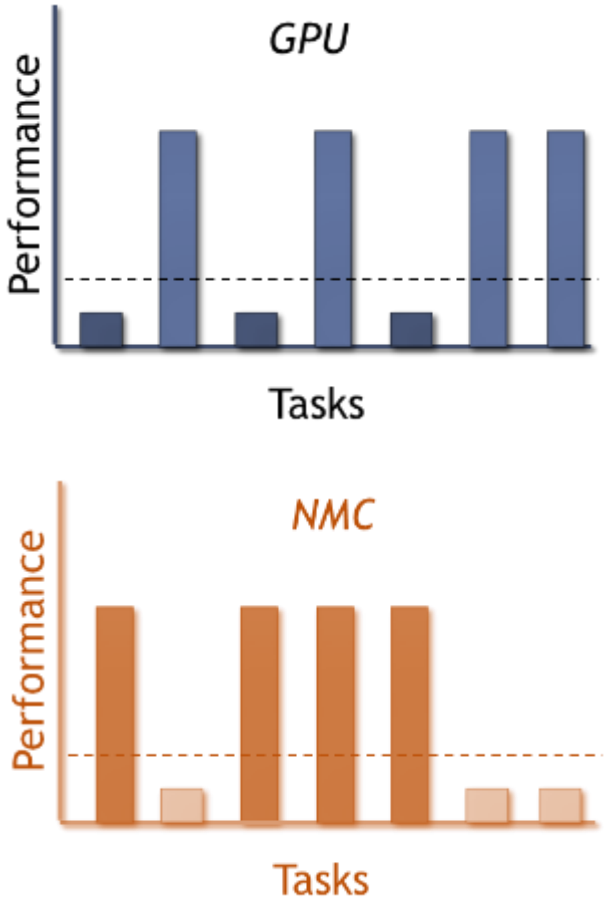
Neuromorphic is likely similar to GPUs in degree of specialization

COINFLIPS

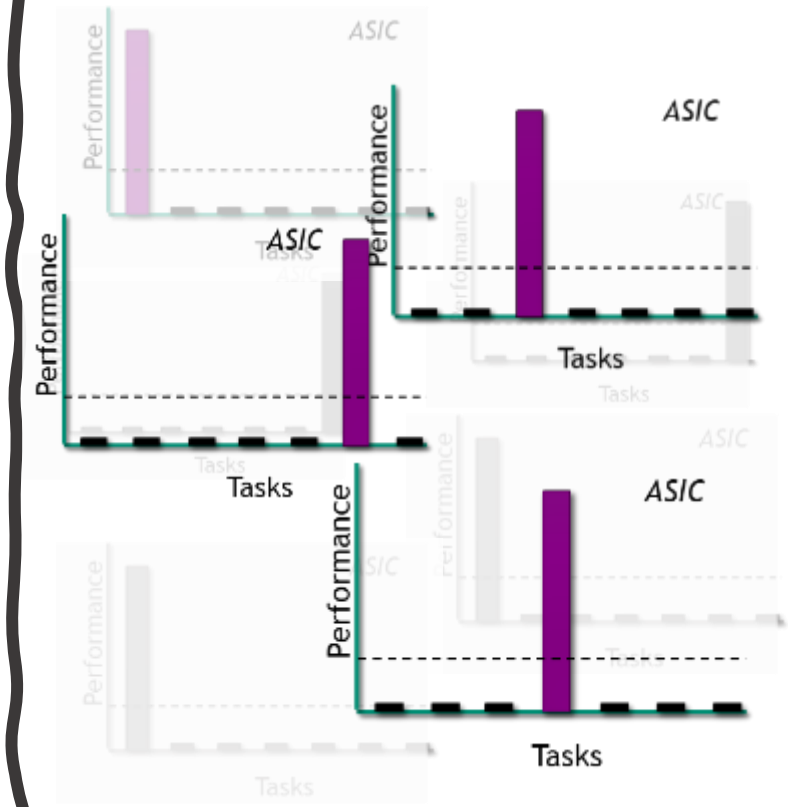
Truly General Purpose



Specialized General Purpose



Application Specific

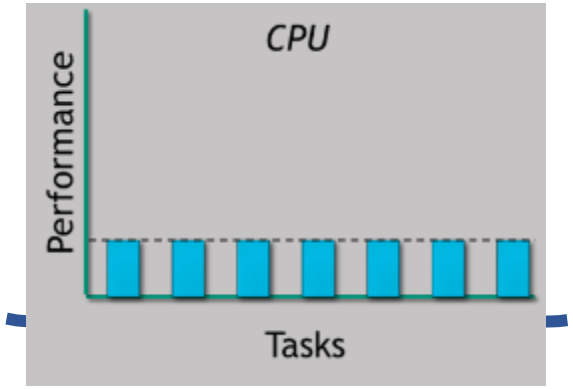


Separating the “can do” from the “should do”

COINFLIPS

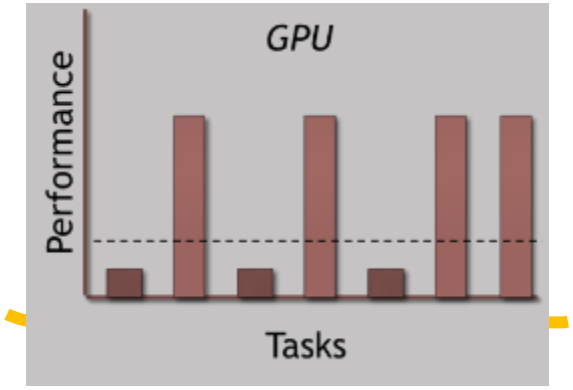
Can implement on NMC, but only to avoid I/O

- Arithmetic (adding, subtraction, multiplication, etc.)
- Data filtering
- Sorting
- Data conversions
- …



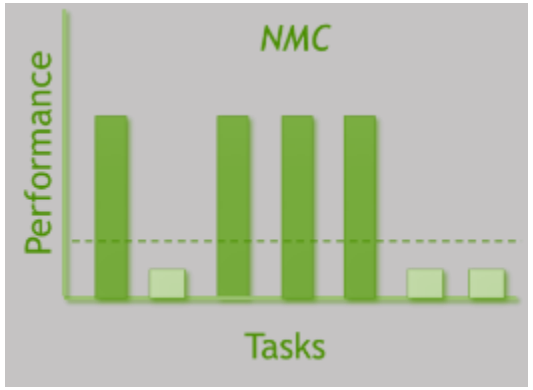
Possibly good on NMC, but there may be alternatives

- Deep learning / conventional artificial neural networks
- Parallel data processing (background and change detection, convolutions, etc)
- Linear algebra (MVM, cross-correlations, L1-norm, etc)
- Classic machine learning (SVMs, k-nearest neighbors, clustering)



Should implement on NMC once systems reach scale

- **Algorithms the brain actually uses (* *we don’t have these yet…*)**
- **Random walks / Discrete Time Monte Carlo**
- ***Some* Graph Algorithms (Dynamic programming, Djikstra, triangle counting, graph cut, etc)**
- ***Some* neural networks**



Neuromorphic computing can impact a broad range of applications

IOPscience

Neuromorphic Computing and Engineering

ACCEPTED MANUSCRIPT • OPEN ACCESS

A review of non-cognitive applications for neuromorphic computing

James Aimone[1], Prasanna Date[2], Gabriel Fonseca-Guerra[3], Kathleen Hamilton[2], Kyle Henke[4], Bill Kay[5], Garrett Kenyon[4], Shruti Kulkarni[2], Susan Mniszewski[6], Maryam Parsa[7], Sumedh Risbud[3], Catherine Schuman[8], William Severa[1] and J. Darby Smith[1] – Hide full author list

Accepted Manuscript online 10 August 2022 • © 2022 The Author(s). Published by IOP Publishing Ltd

64 Total downloads

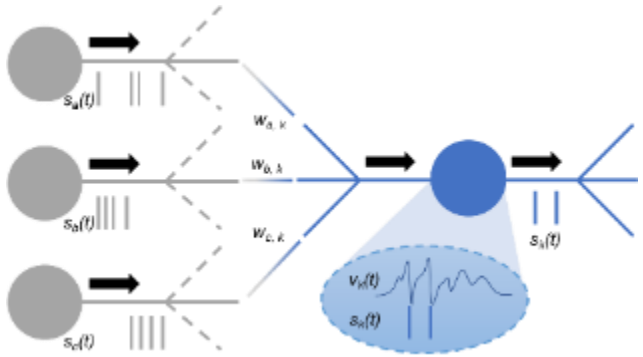
14

Turn on MathJax

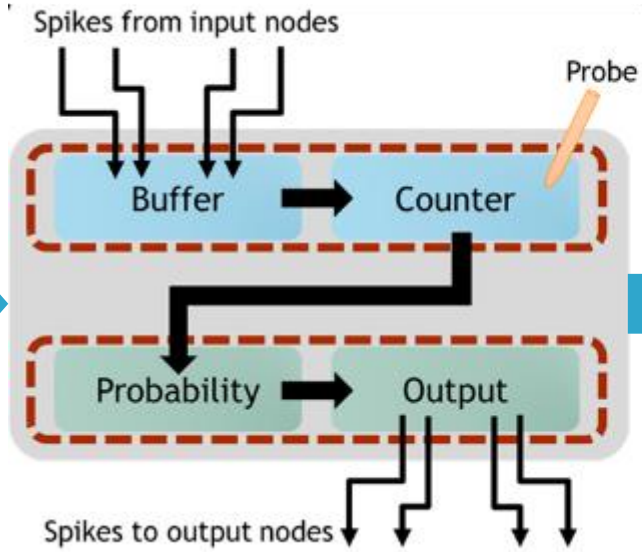
Share this article

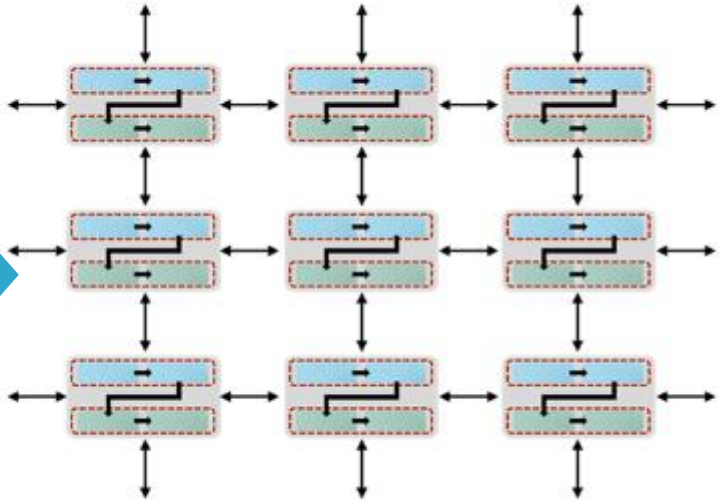
Spiking Scientific Computing

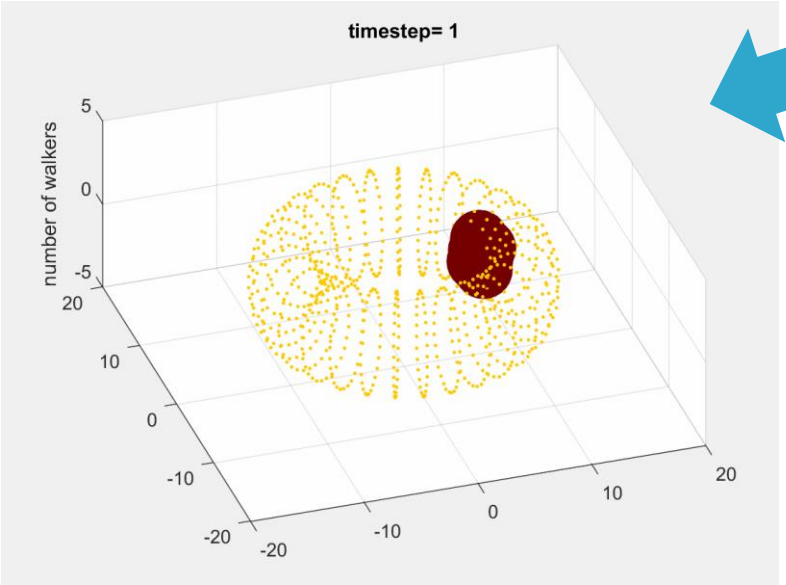
Today's spiking NMC shows energy advantage over conventional approaches on Monte Carlo simulations

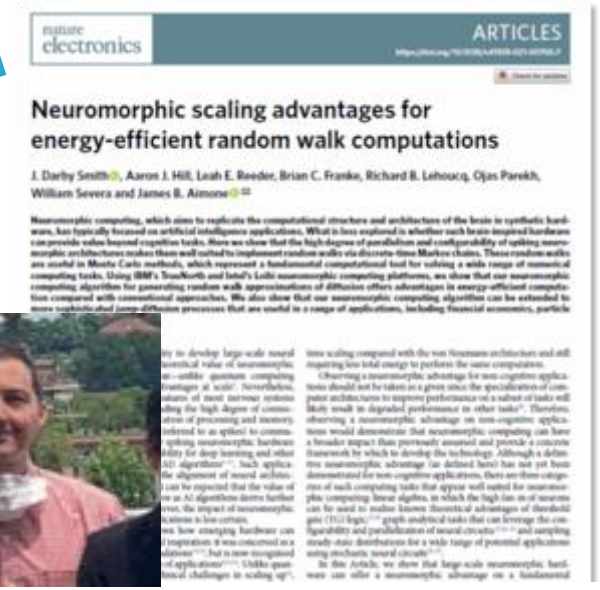


Leaky Integrate and Fire Neuron





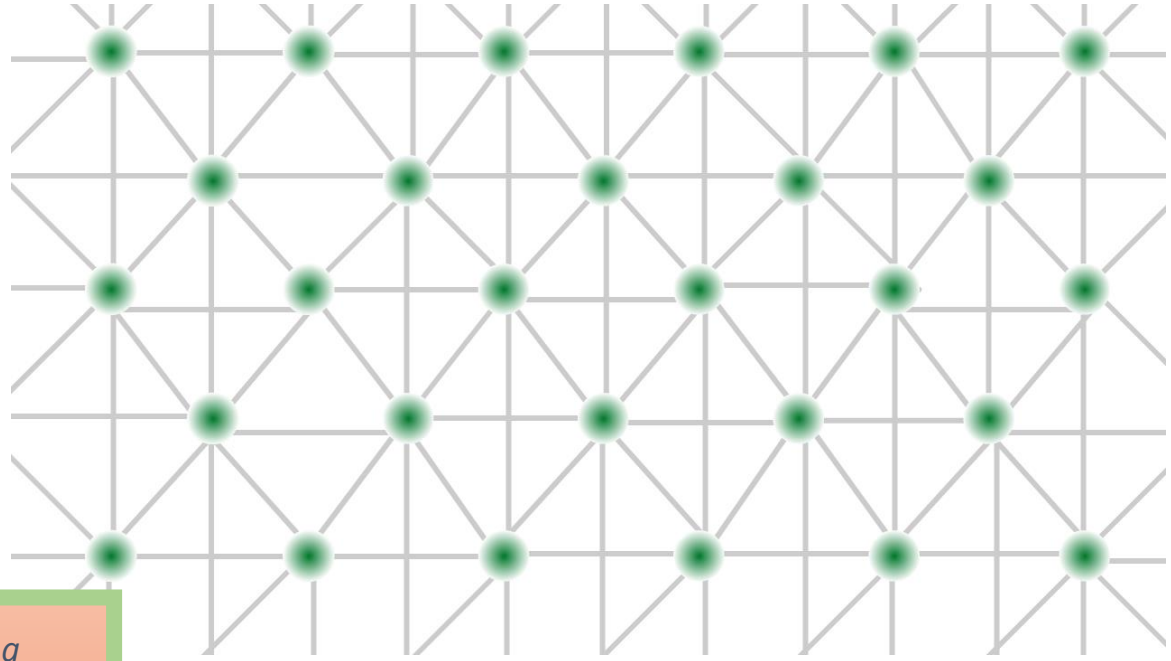




Spiking Scientific Computing

Smith et al., Nature Electronics 2022

Neuromorphic computing advantage appears to be when an algorithm can split task across computational graph with sparse communication



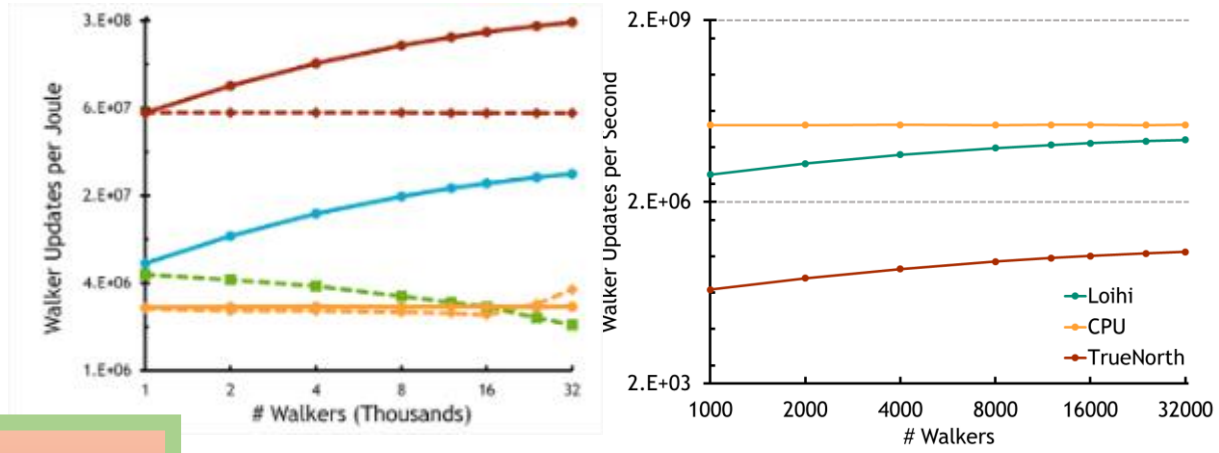
Spiking Scientific Computing

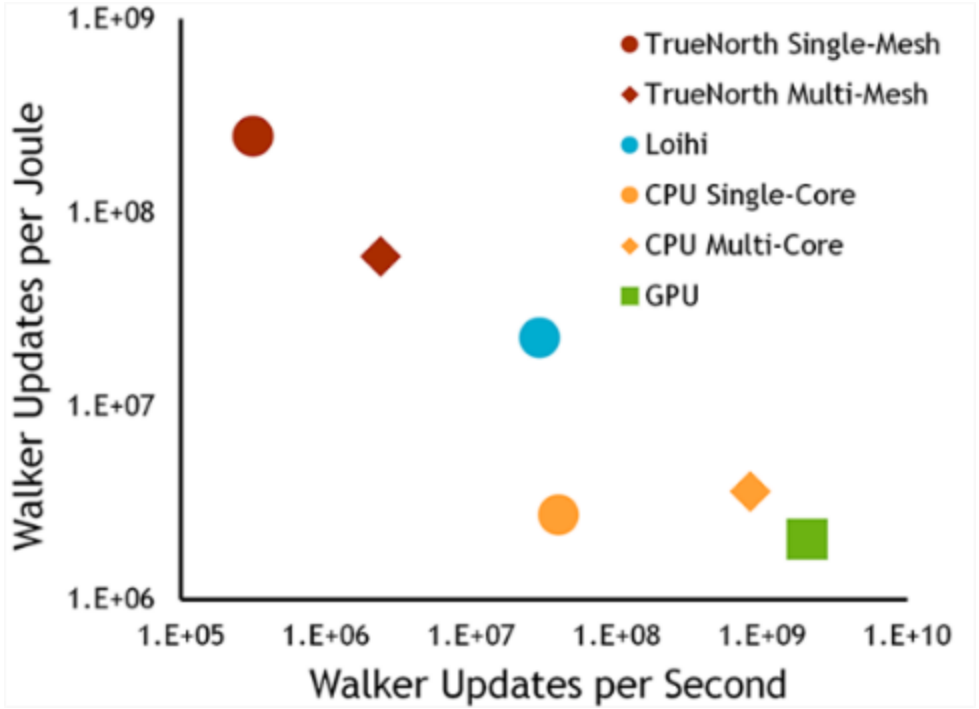
- Monte Carlo simulations
  Discrete Time Markov Chains
- Dynamic programming
- Graph neural networks
- …

We can identify a neuromorphic advantage for simulating random walks

COINFLIPS

We define a *neuromorphic advantage* as an algorithm that shows a demonstrable **advantage** in terms of one resource (e.g., energy) while exhibiting comparable **scaling** in other resources (e.g., time).





Spiking Scientific Computing

Smith et al., Nature Electronics 2022

Math: What PDEs can these stochastic processes be useful for?

COINFLIPS

Class of Partial Integro-Differential Equations:

$$\frac{\partial}{\partial t}u(t,\boldsymbol{x}) = \frac{1}{2}\sum_{i,j}(\boldsymbol{a}\boldsymbol{a}^{\top})_{i,j}(t,\boldsymbol{x})\frac{\partial^2}{\partial x_i \partial x_j}u(t,\boldsymbol{x}) + \sum_i b_i(t,\boldsymbol{x})\frac{\partial}{\partial x_i}u(t,\boldsymbol{x})$$
$$+\lambda(t,\boldsymbol{x})\int\Big(u\big(t,\boldsymbol{x}+\boldsymbol{h}(t,\boldsymbol{x},q)\big)-u(t,\boldsymbol{x})\Big)\phi_Q(q;t,\boldsymbol{x})\mathrm{d}q$$
$$+c(t,\boldsymbol{x})u(t,\boldsymbol{x}) + f(t,\boldsymbol{x}), \qquad x\in\mathbb{R}^d, t\in[0,\infty).$$

Stochastic Process:

NMC Hardware Simulates This Stochastic Process

$$\mathrm{d}\boldsymbol{X}(t) = \boldsymbol{b}\big(t,\boldsymbol{X}(t)\big)\mathrm{d}t + \boldsymbol{a}\big(t,\boldsymbol{X}(t)\big)\mathrm{d}\boldsymbol{W}(t) + \boldsymbol{h}\big(t,\boldsymbol{X}(t),q\big)\mathrm{d}P\big(t;Q,\boldsymbol{X}(t)\big).$$

Monte Carlo Approximates This Expectation

Spiking Scientific Computing

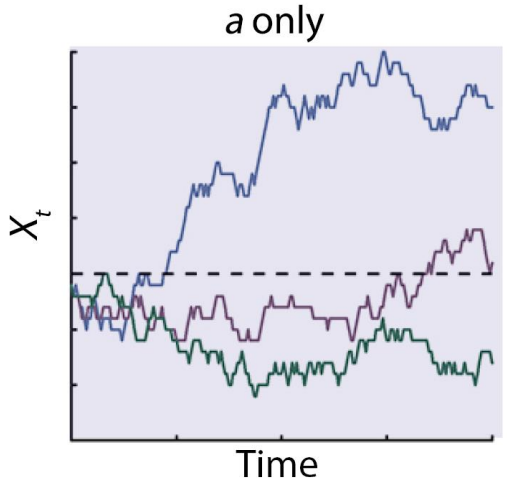
Solution to initial value problem (u(o,x)=g(x)):

$$u(t,\boldsymbol{x}) = \mathbb{E}\left[g\big(\boldsymbol{X}(t)\big)\exp\left(\int_0^t c\big(s,\boldsymbol{X}(s)\big)\mathrm{d}s\right) + \int_0^t f\big(s,\boldsymbol{X}(s)\big)\exp\left(\int_0^s c\big(\ell,\boldsymbol{X}(\ell)\big)\mathrm{d}\ell\right)\mathrm{d}s \,\middle|\, \boldsymbol{X}(0)=\boldsymbol{x}\right].$$

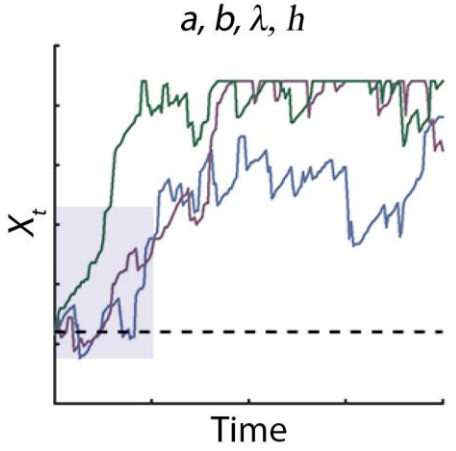
Smith et al., Nature Electronics 2022

Neural MC algorithm can run wide range of stochastic processes

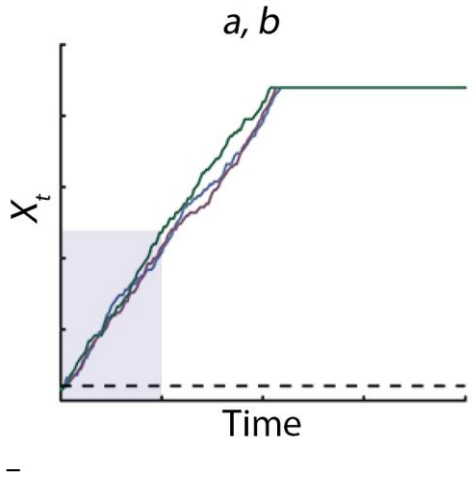
Diffusion



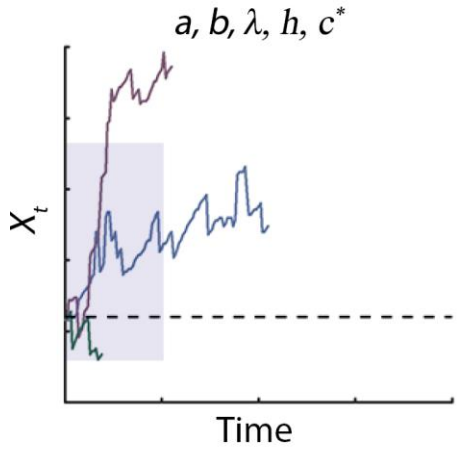
Jump processes



Drift



Absorption / Decay



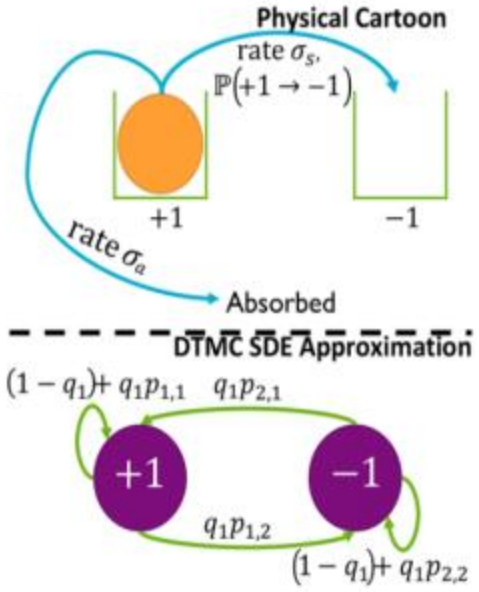
Spiking Scientific Computing

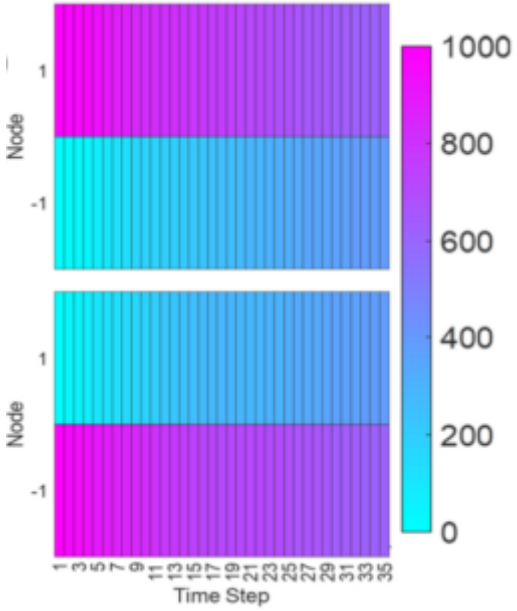
Smith et al., Nature Electronics 2022

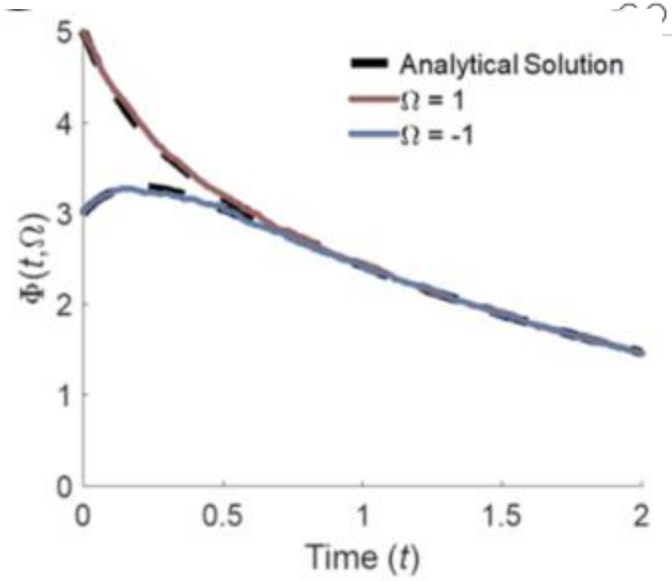
Some more applied examples

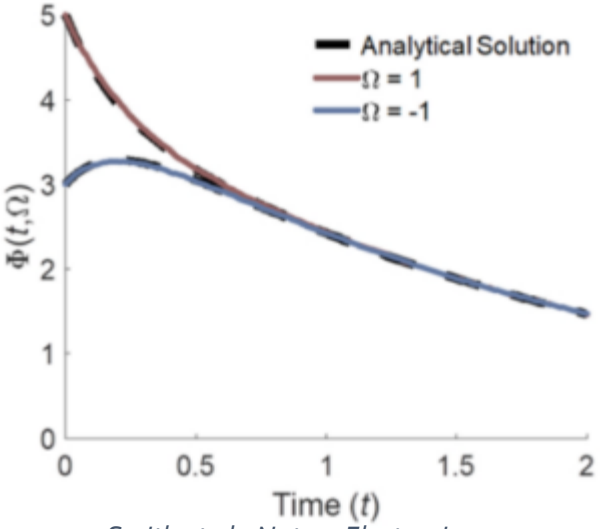
- Boltzmann state transition
  - Particle can exist in 2 states (+1 or -1) or be absorbed.
  - Implement as simple stochastic process on TrueNorth

Spiking Scientific Computing





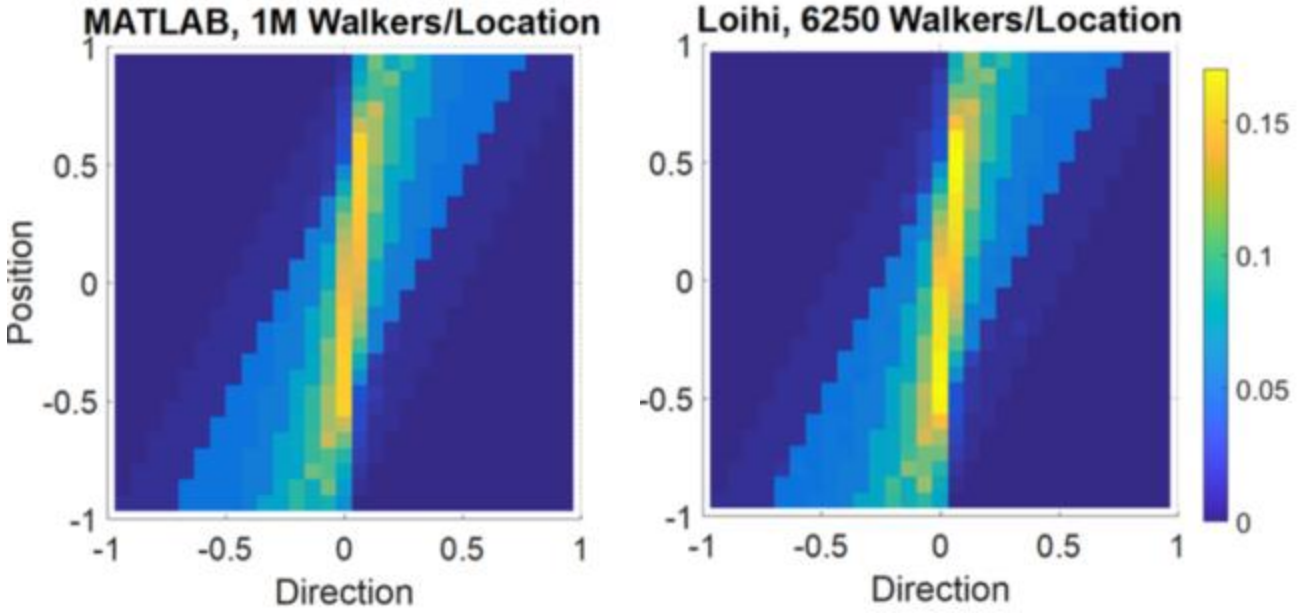


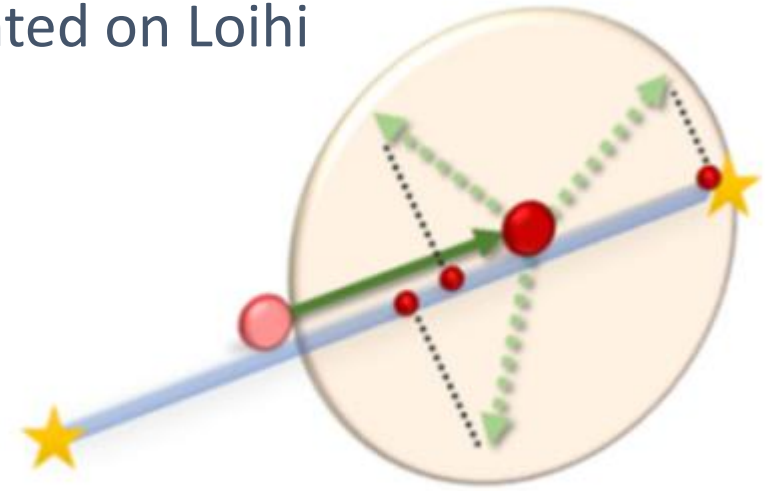


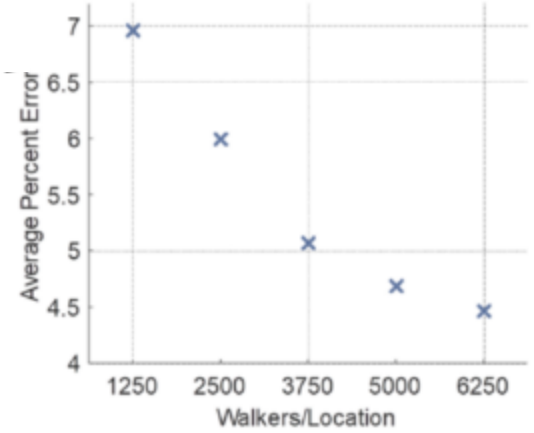
Smith et al., Nature Electronics 2022

Some more applied examples

- 1D particle transport
  - Particle moves in 2D, only track 1D.
  - At point x=0, particle reflects in random direction
  - Track velocity in x-dimension and angle
  - Implemented on Loihi







Smith et al., Nature Electronics 2022

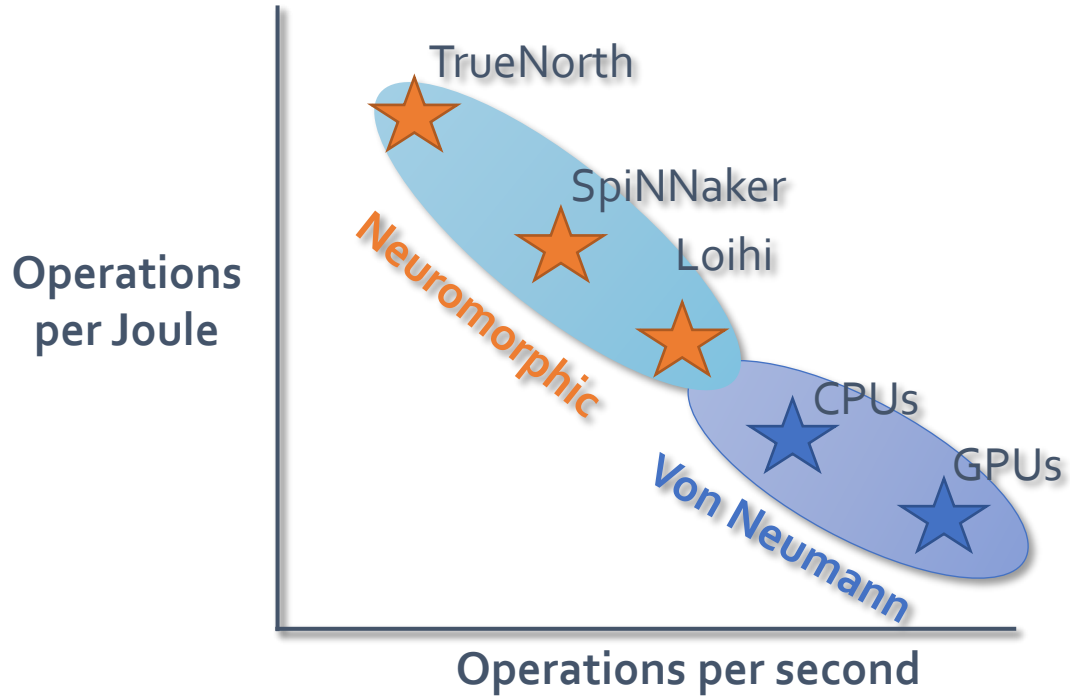
Spiking Scientific Computing

Today's large scale neuromorphic systems are on *Pareto Frontier* of computing

- Broad class of algorithms fit this tradeoff
  - Monte Carlo / Probabilistic
  - Graph analytics
  - Artificial intelligence
  - Optimization
- Architectural advantage
  - Event-driven processing
  - Massive parallelism
- Limitations
  - Still CMOS devices
  - Architecture is a one time benefit not an extension to Moore's Law

Spiking Scientific Computing

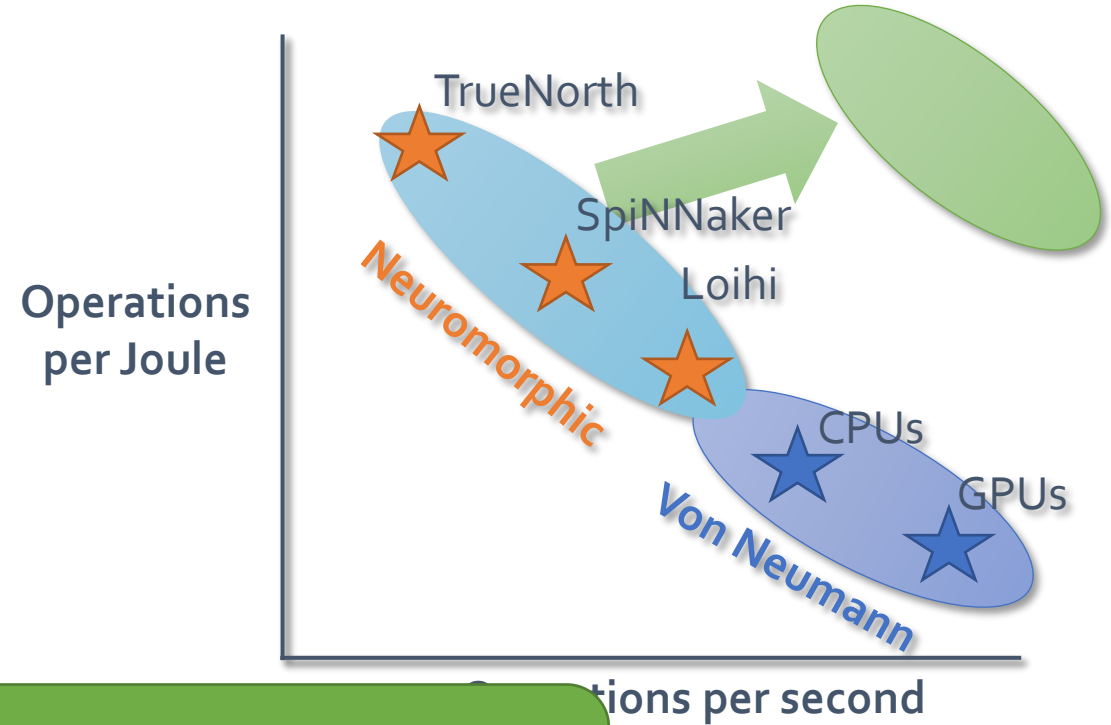
If we're honest; who will pick energy efficiency over speed?



Today's large scale neuromorphic systems are on *Pareto Frontier* of computing

COINFLIPS

- Broad class of algorithms fit this tradeoff
  - Monte Carlo / Probabilistic
  - Graph analytics
  - Artificial intelligence
  - Optimization
- Architectural advantage
  - Event-driven processing
  - Massive parallelism
- Limitations
  - Still CMOS devices
  - Architecture is a one time benefit not an extension to Moore's Law



Spiking Scientific Computing

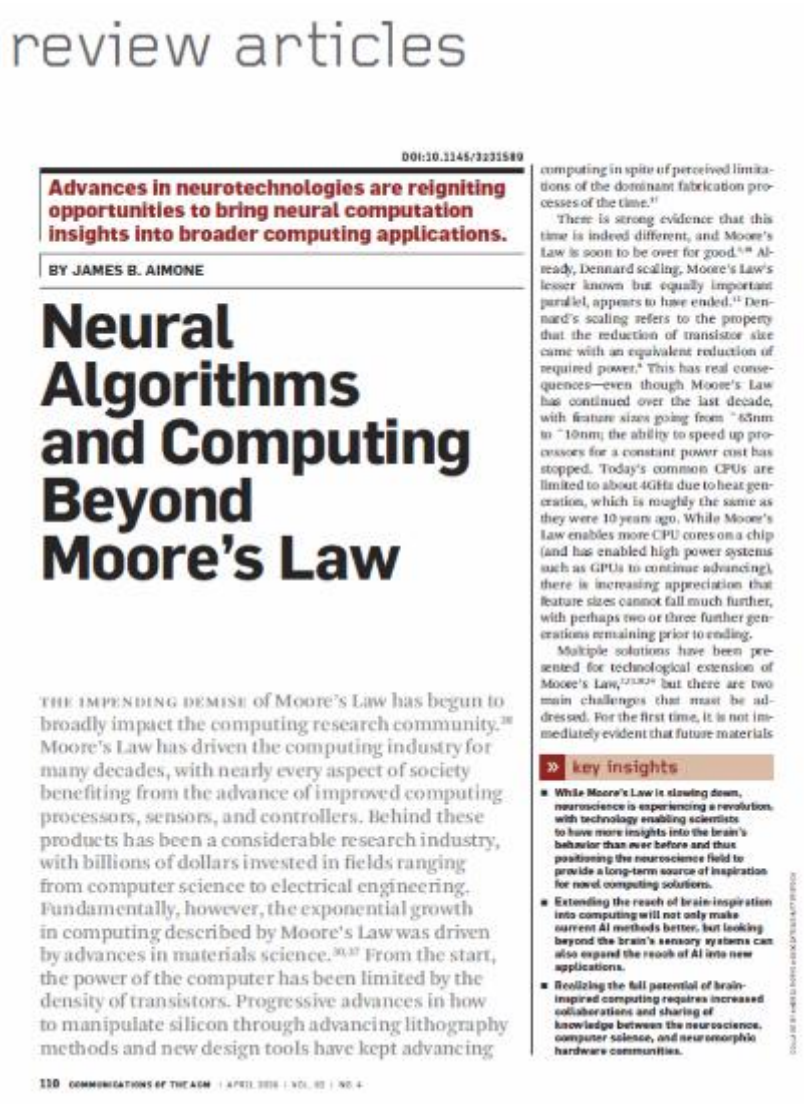
Opportunity for Brain-Inspired Materials, Devices & Algorithms

Increasing processing (density, speed, capabilities, etc) while preserving energy advantage and jump neuromorphic over Pareto Frontier

COINFLIPS

So what about algorithms from the brain?

COINFLIPS



review articles

DOI:10.1145/3231589

Advances in neurotechnologies are reigniting opportunities to bring neural computation insights into broader computing applications.

BY JAMES B. AIMONE

Neural Algorithms and Computing Beyond Moore's Law

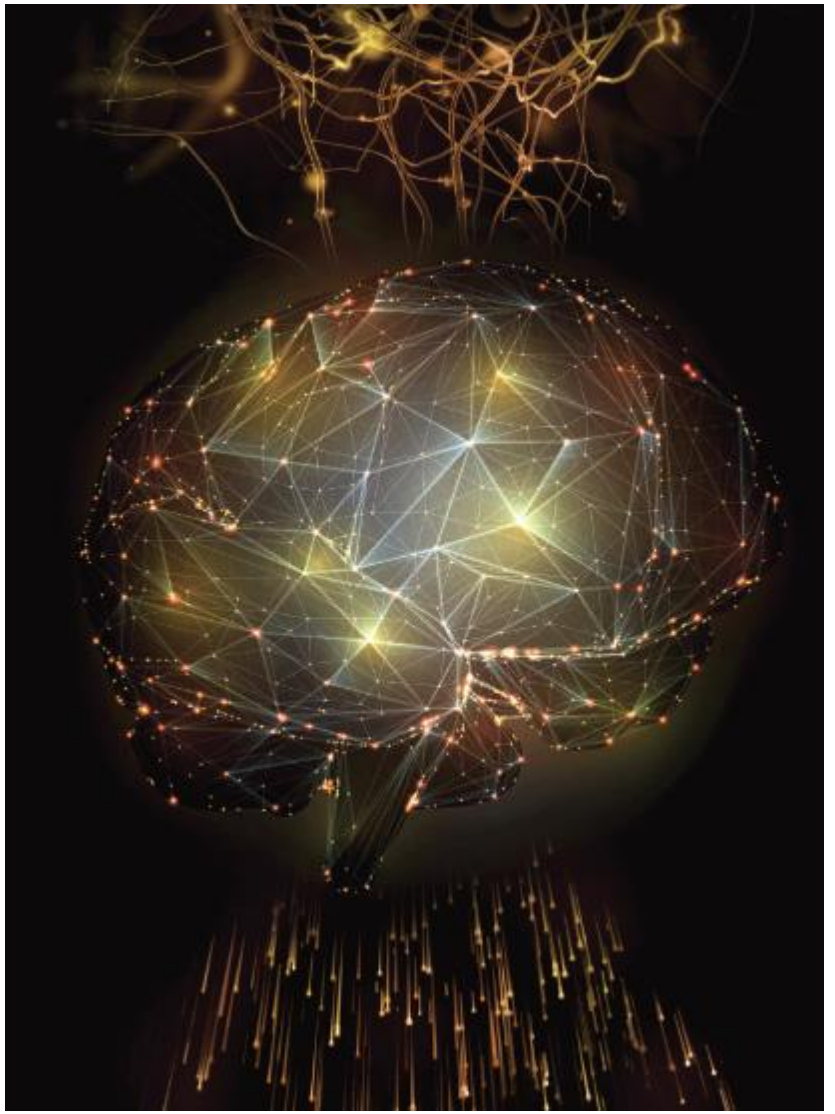
THE IMPENDING DEMISE of Moore's Law has begun to broadly impact the computing research community.[28] Moore's Law has driven the computing industry for many decades, with nearly every aspect of society benefiting from the advance of improved computing processors, sensors, and controllers. Behind these products has been a considerable research industry, with billions of dollars invested in fields ranging from computer science to electrical engineering. Fundamentally, however, the exponential growth in computing described by Moore's Law was driven by advances in materials science.[20,37] From the start, the power of the computer has been limited by the density of transistors. Progressive advances in how to manipulate silicon through advancing lithography methods and new design tools have kept advancing computing in spite of perceived limitations of the dominant fabrication processes of the time.[37]

There is strong evidence that this time is indeed different, and Moore's Law is soon to be over for good.[5,28] Already, Dennard scaling, Moore's Law's lesser known but equally important parallel, appears to have ended.[11] Dennard's scaling refers to the property that the reduction of transistor size came with an equivalent reduction of required power.[8] This has real consequences—even though Moore's Law has continued over the last decade, with feature sizes going from ~65nm to ~10nm; the ability to speed up processors for a constant power cost has stopped. Today's common CPUs are limited to about 4GHz due to heat generation, which is roughly the same as they were 10 years ago. While Moore's Law enables more CPU cores on a chip (and has enabled high power systems such as GPUs to continue advancing), there is increasing appreciation that feature sizes cannot fall much further, with perhaps two or three further generations remaining prior to ending.

Multiple solutions have been presented for technological extension of Moore's Law,[7,13,29,34] but there are two main challenges that must be addressed. For the first time, it is not immediately evident that future materials

» key insights

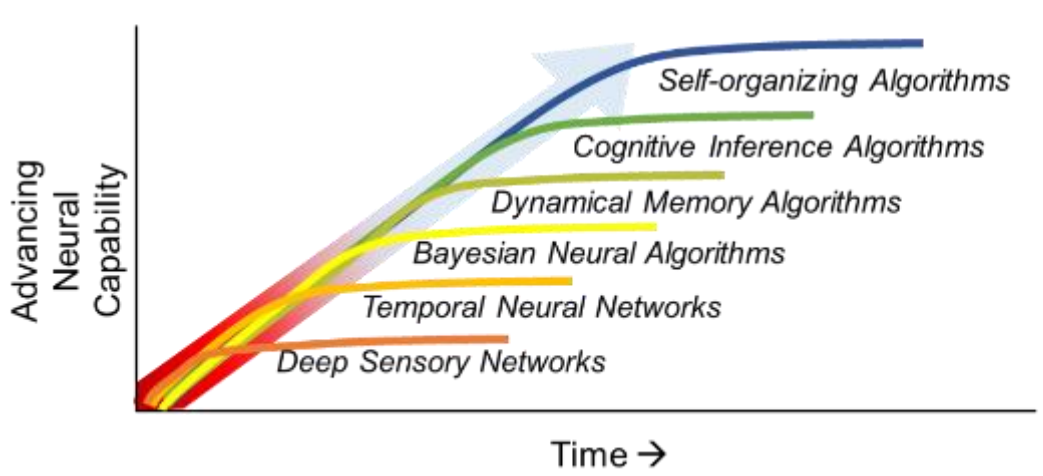
- While Moore's Law is slowing down, neuroscience is experiencing a revolution, with technology enabling scientists to have more insights into the brain's behavior than ever before and thus positioning the neuroscience field to provide a long-term source of inspiration for novel computing solutions.
- Extending the reach of brain-inspiration into computing will not only make current AI methods better, but looking beyond the brain's sensory systems can also expand the reach of AI into new applications.
- Realizing the full potential of brain-inspired computing requires increased collaborations and sharing of knowledge between the neuroscience, computer science, and neuromorphic hardware communities.

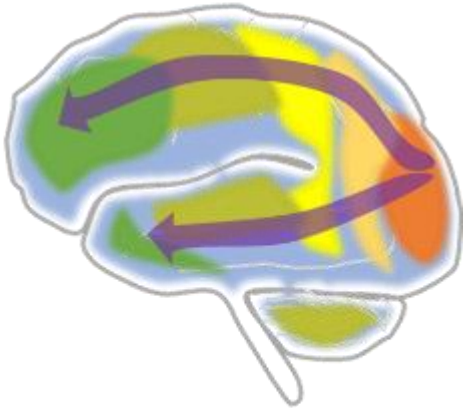


Brain Inspiration

COINFLIPS

Algorithm Class	Current Algorithms	Inspiration	Application
Deep Vision Processing	Deep Convolutional Networks (VGG, AlexNet, GoogleNet, etc.), HMax, Neocognitron	Hierarchy of sensory nuclei and early sensory cortices	Static feature extraction (e.g., images) & pattern classification
Temporal Neural Networks	Deep Recurrent Networks (long short-term memory), Hopfield Networks	Local recurrence of most biological neural circuits, especially higher sensory cortices	Dynamic feature extraction (e.g., videos, audio) & classification
Bayesian Neural Algorithms	Predictive Coding, Hierarchical Temporal Memory	Substantial reciprocal feedback between "higher" and "lower" sensory cortices	Inference across spatial and temporal scales
Dynamical Memory and Control Algorithms	Liquid State Machines, Echo State Networks, Neural Engineering Framework	Continual dynamics of hippocampus, cerebellum, and prefrontal and motor cortices	Online learning content-addressable memory & adaptive motor control
Cognitive Inference Algorithms	Reinforcement learning (e.g., Q-learning)	Integration of multiple modalities and memory into prefrontal cortex, which provides top-down influence on sensory processing	Context and experience dependent information processing and decision making
Self-organizing Algorithms	Neurogenesis Deep Learning	Initial development and continuous refinement of neural circuits to specific input and outputs	Automated neural algorithm development for unknown input and output transformations

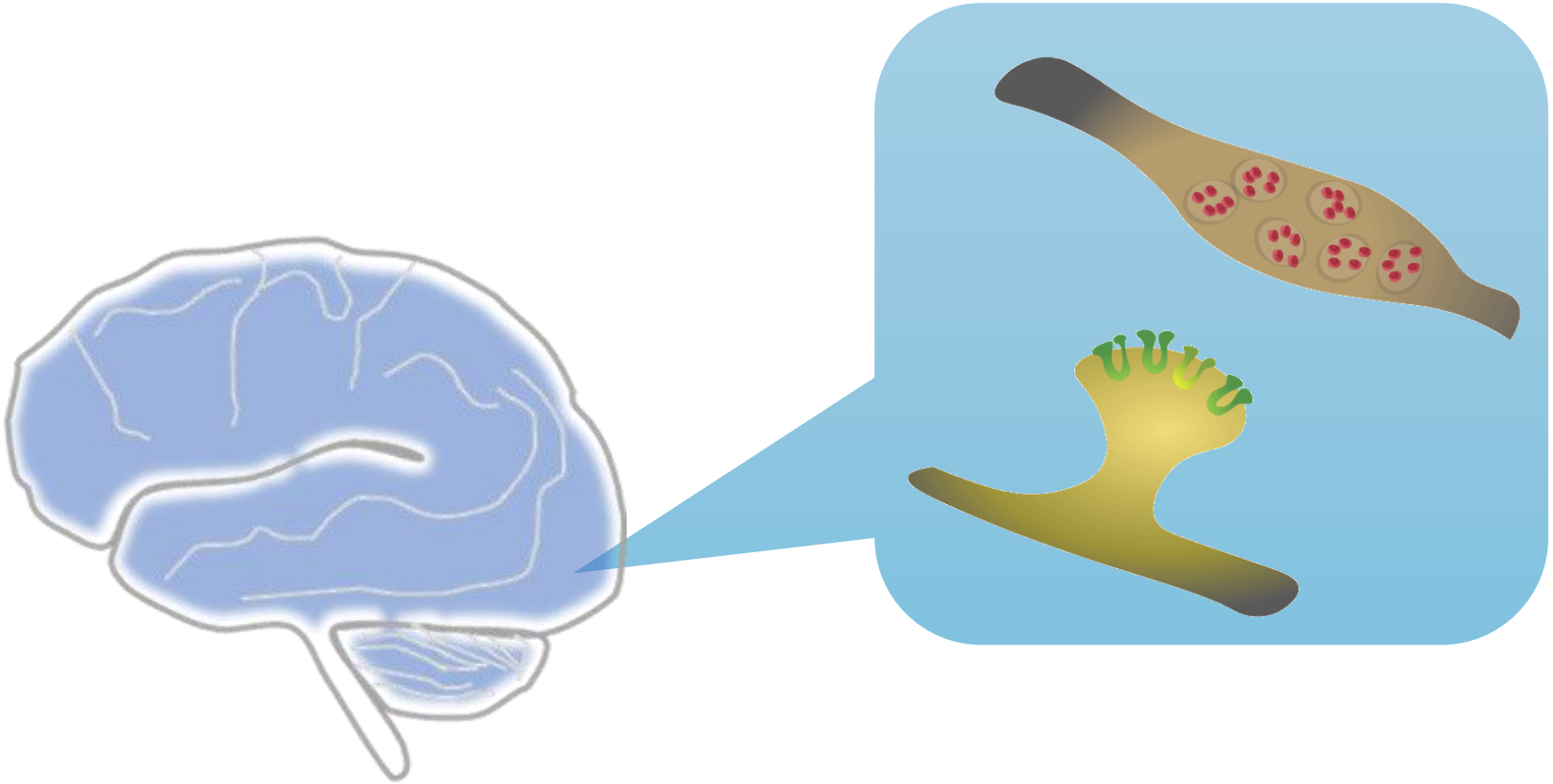




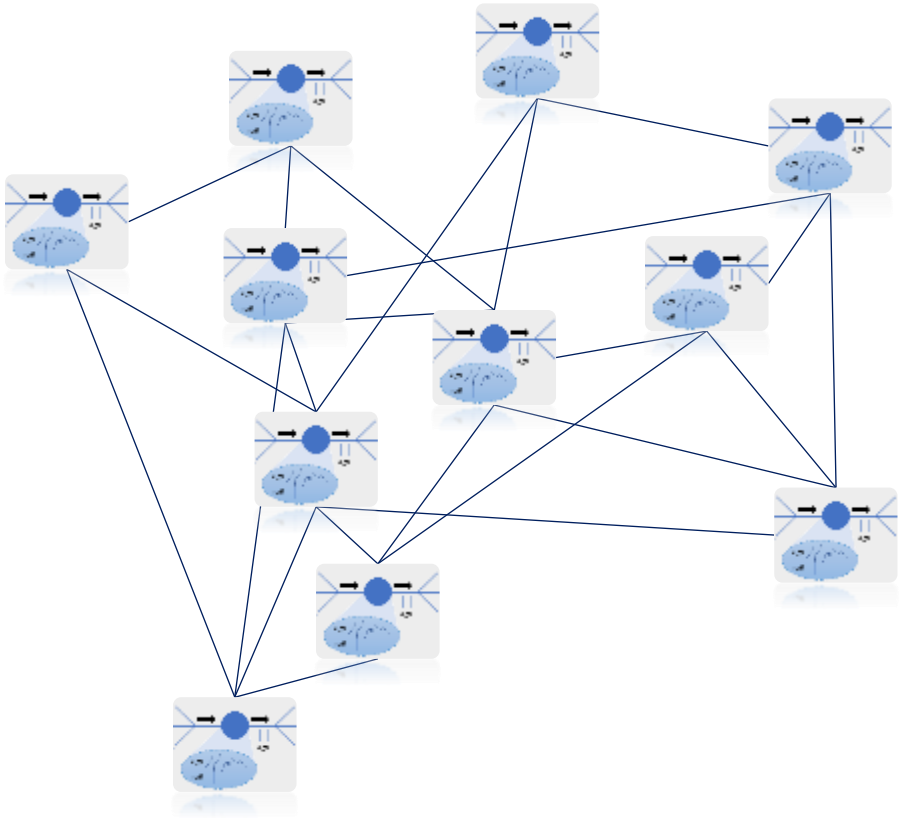
Brain Inspiration

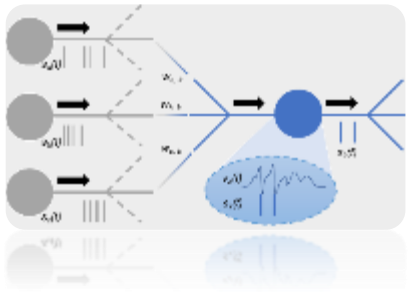
Aimone, CACM 2019

Our brains are stochastic all the way down…



What are the dynamical algorithms of the brain?





- Our brains consist of *billions* of asynchronous sparsely connected dynamical neurons with ubiquitous stochasticity

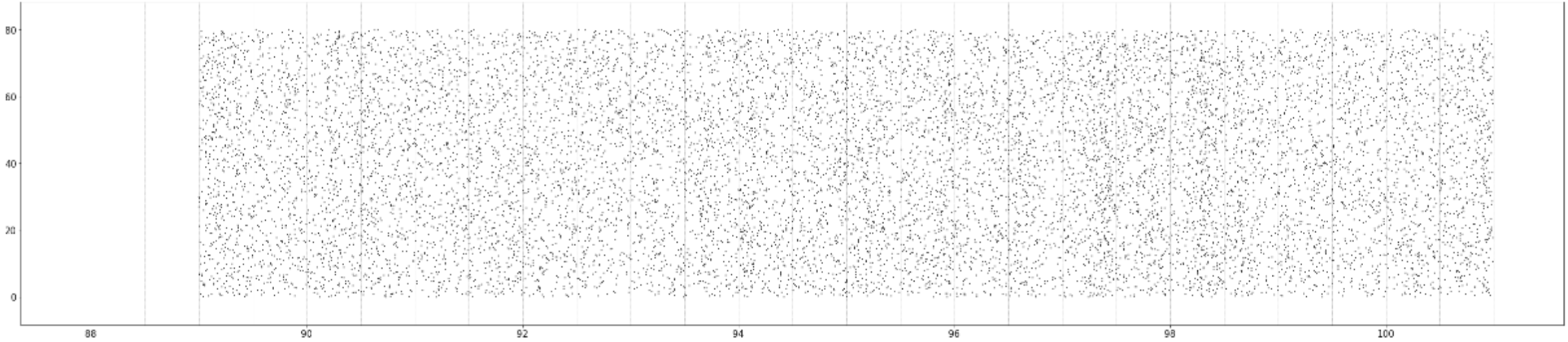
- Neuromorphic chips consist of *millions* of asynchronous sparsely connected dynamical neurons with modest stochasticity available

Yet…

- We keep trying to impose algorithms designed for densely connected synchronized layers of thousands of neurons operating deterministically

What are the algorithms the brain is using?

COINFLIPS



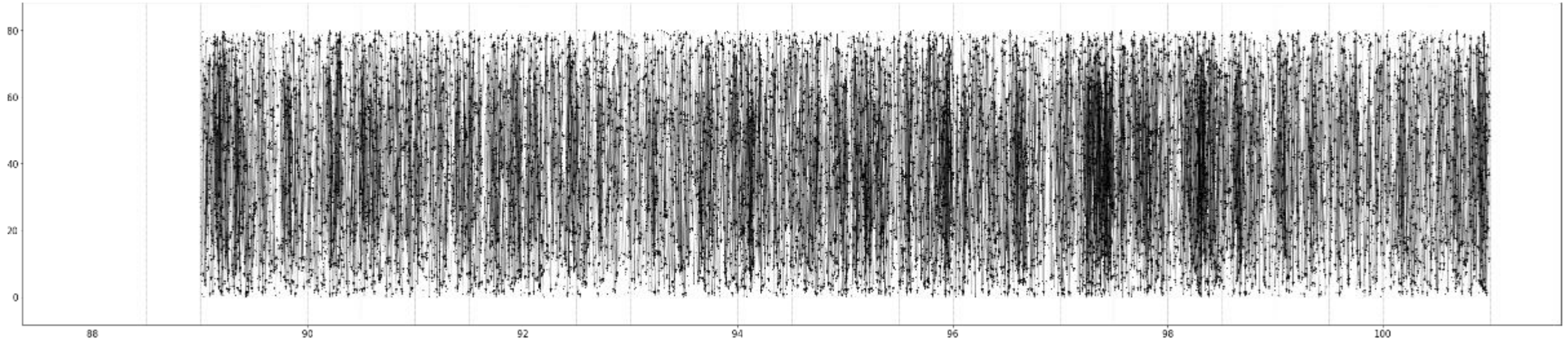
- Neuron connectivity is primarily recurrent
- Mix of inhibition and excitation
- Deterministic spike generation, random synaptic transmission, unknown inputs
- Asynchronous, chaotic like patterns of activity

- *This is *very* difficult to interpret, much less leverage for computing!*

GNATs

Graphical Neural Activity Threads (GNATs)

COINFLIPS



Connect Causally-Related Spikes

- What is causal?
  - Synapse exists between neurons
  - Causally-timed Spike occurs within time window

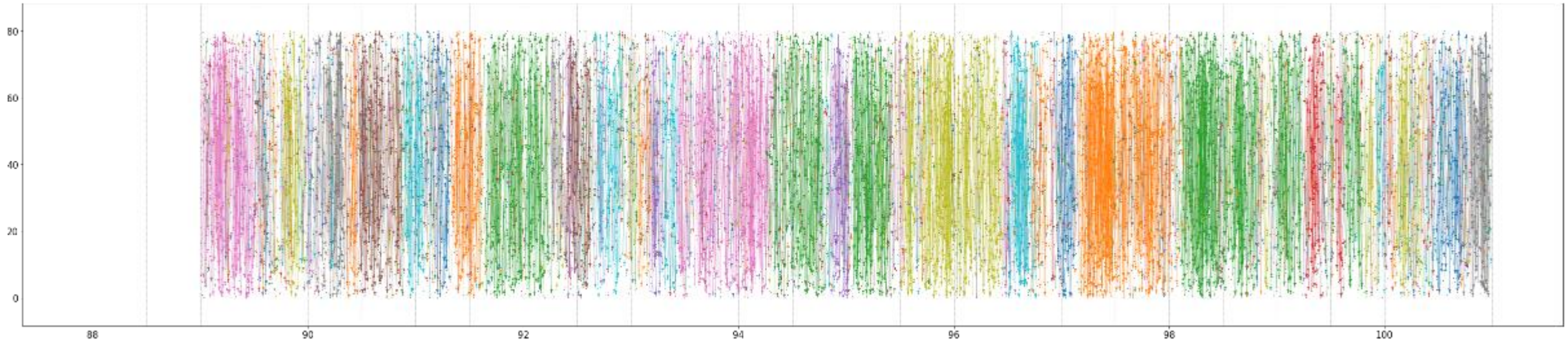
$$\Omega_{\alpha\beta}(t_\alpha - t_\beta) = W_{\alpha\beta}\theta[t_\alpha - t_\beta - \delta_{\alpha\beta}]e^{-(t_\alpha - t_\beta - \delta_{\alpha\beta})/\tau}$$

GNATs

Theilman et al., Submitted 2023

Graphical Neural Activity Threads (GNATs)

COINFLIPS



Connect Causally-Related Spikes

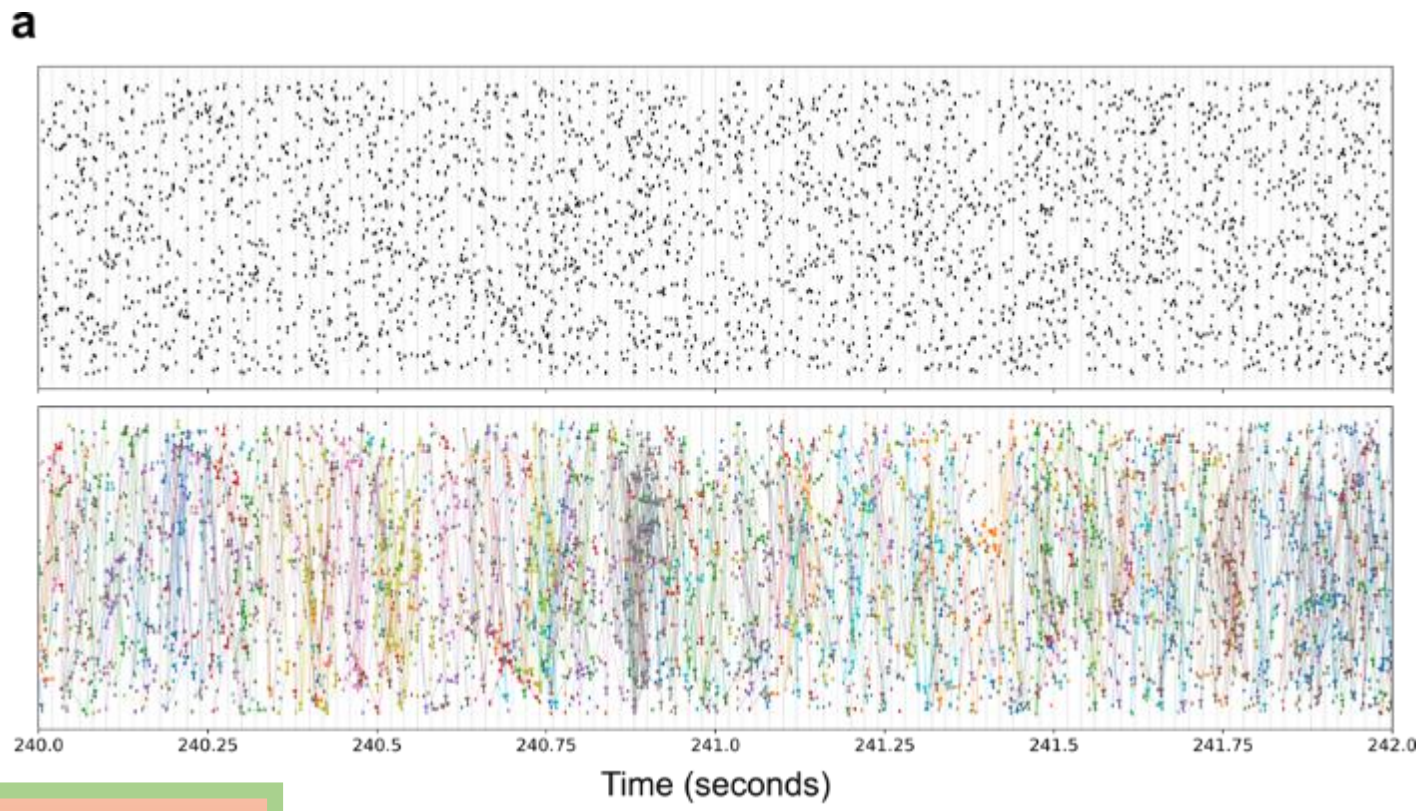
$$\Omega_{\alpha\beta}(t_\alpha - t_\beta) = W_{\alpha\beta}\theta[t_\alpha - t_\beta - \delta_{\alpha\beta}]e^{-(t_\alpha - t_\beta - \delta_{\alpha\beta})/\tau}$$

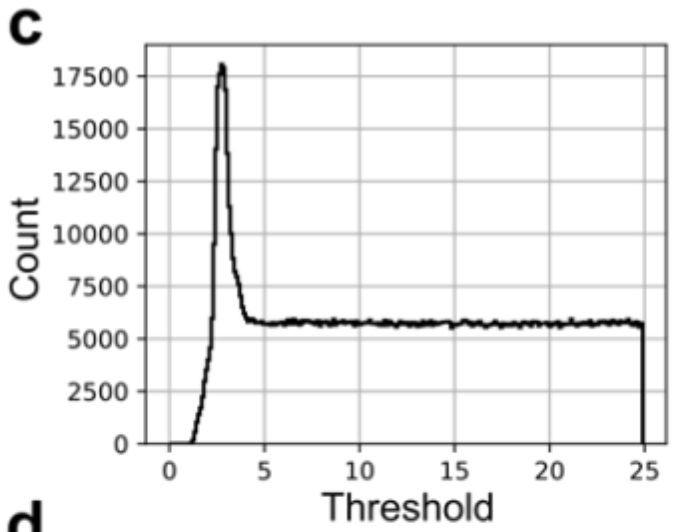
Color Disjoint Connected Components

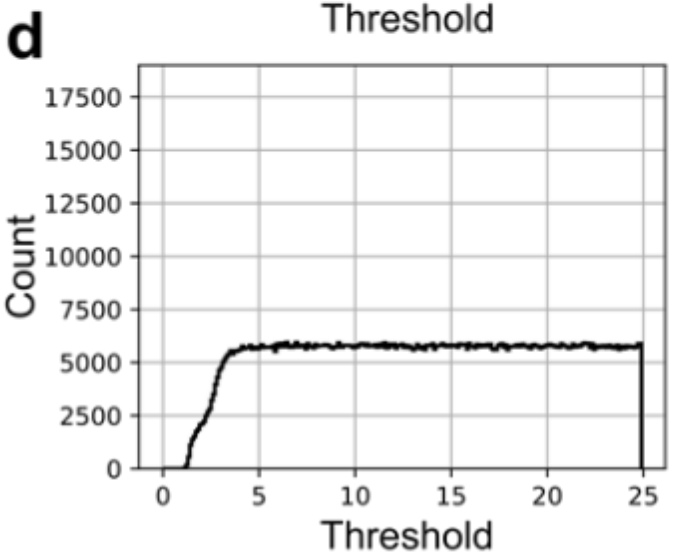
GNATs

GNATs emerge from structure of 80/20 networks

COINFLIPS

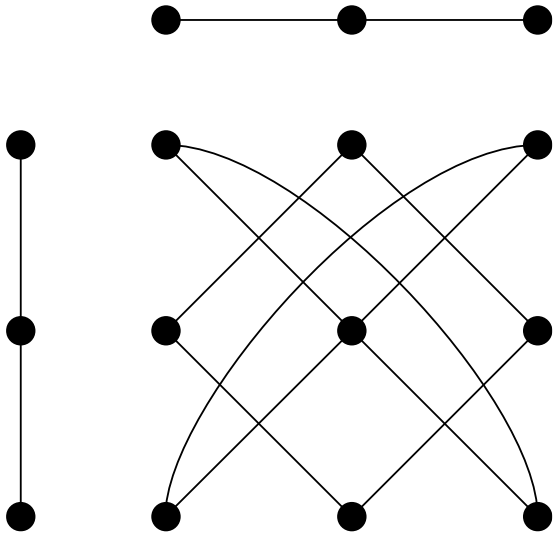




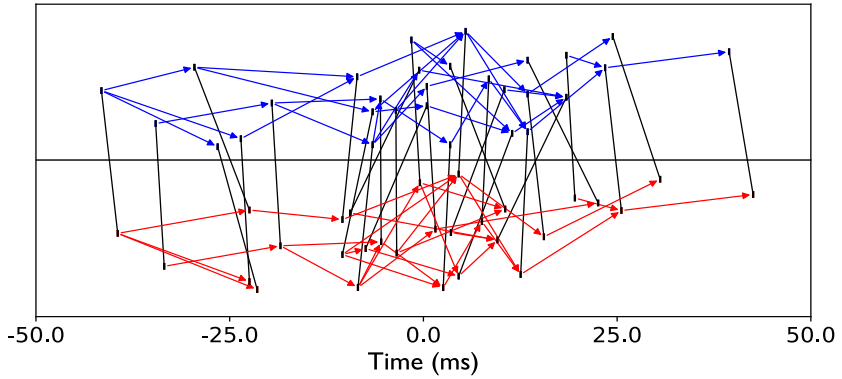


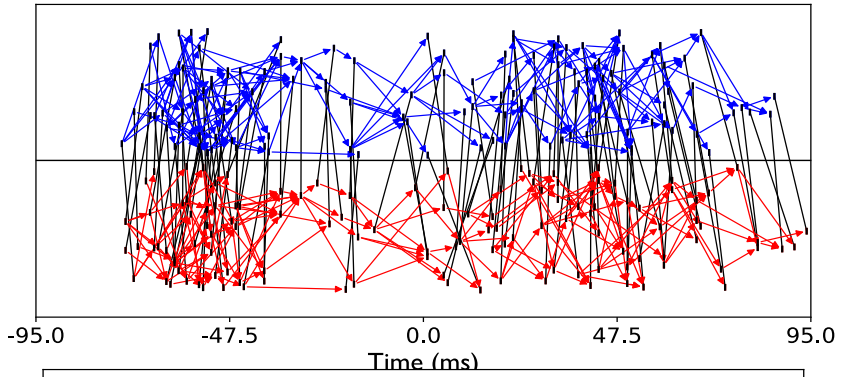
GNATs

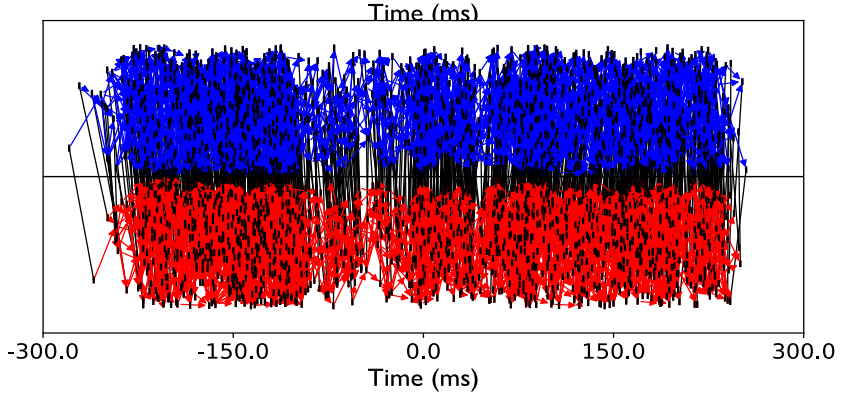
Isomorphic GNATs = *computational motifs*?



Modular graph product



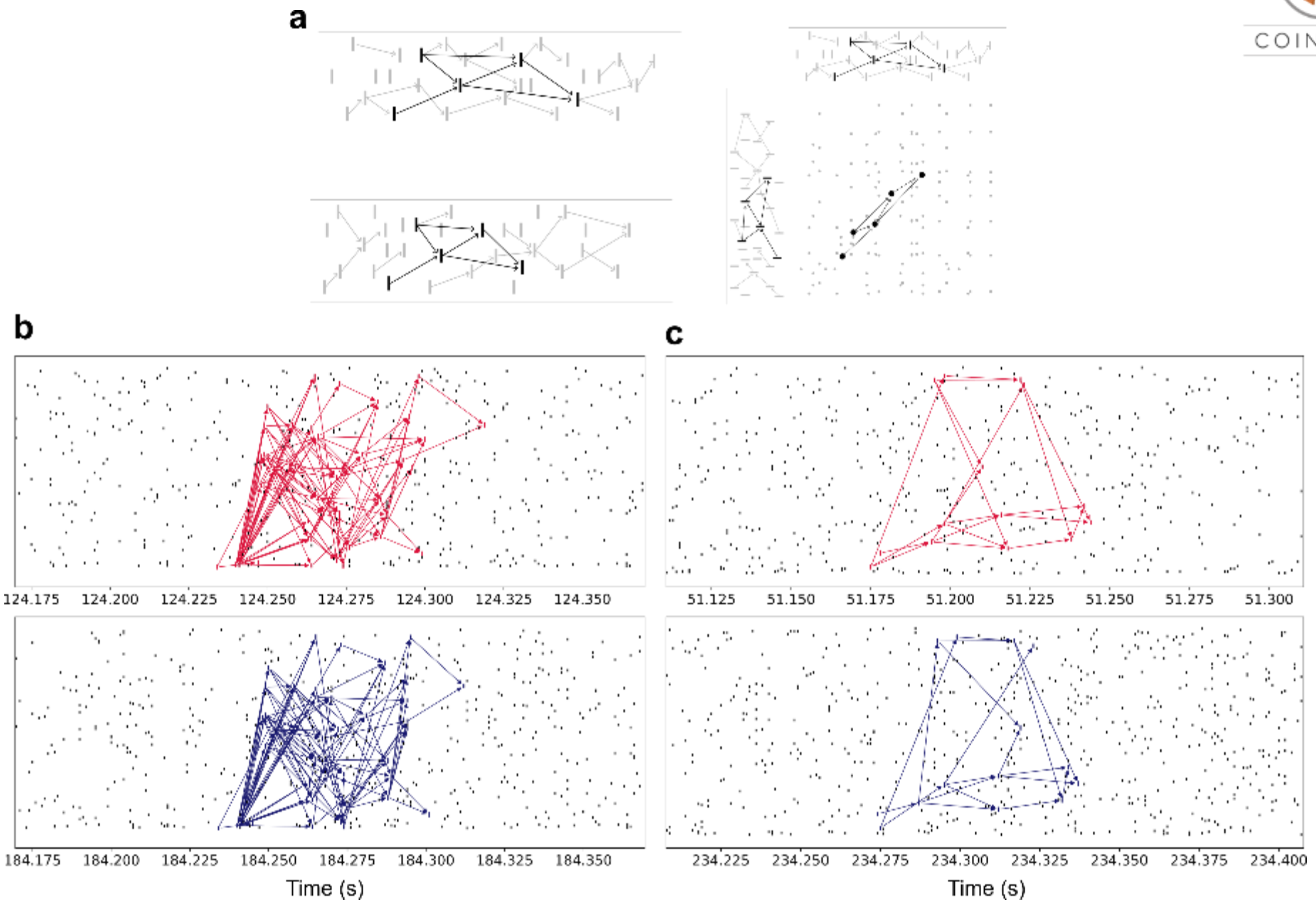




- Similar causal sequences reappear embedded in larger spiking contexts
- Precise timing is not preserved, but causal influence is preserved

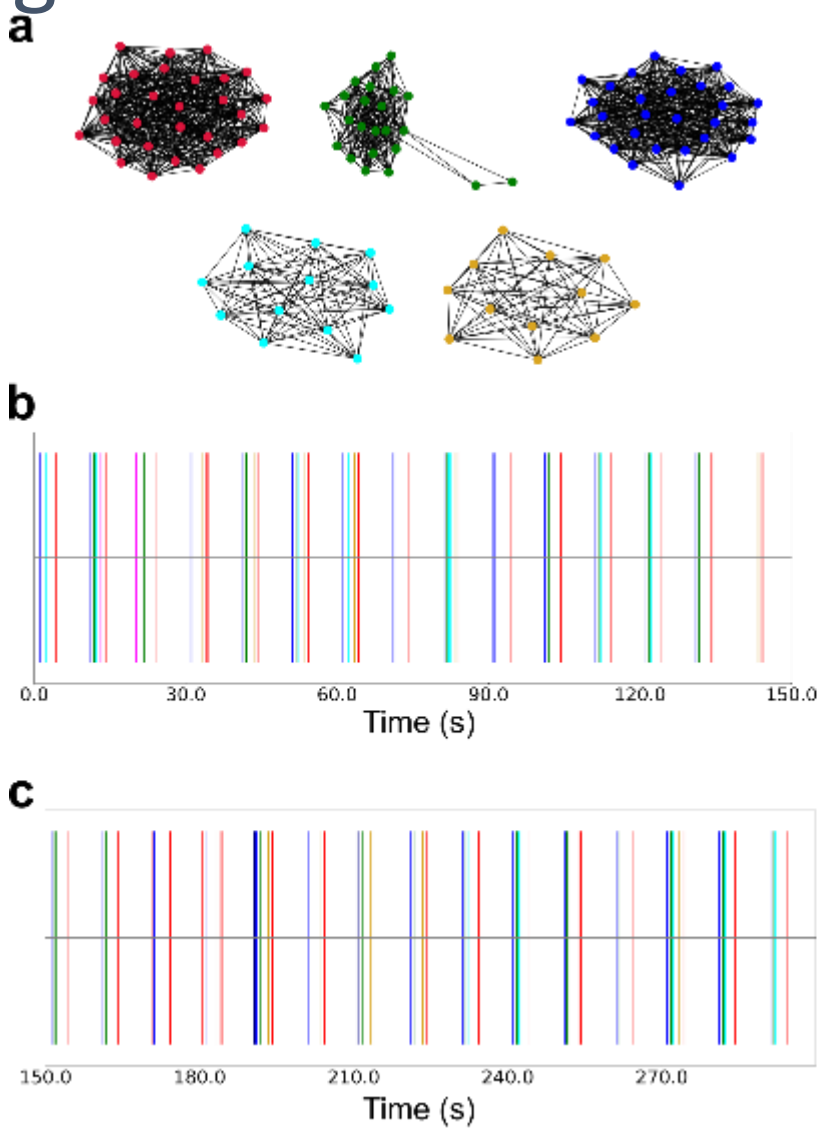
GNATs

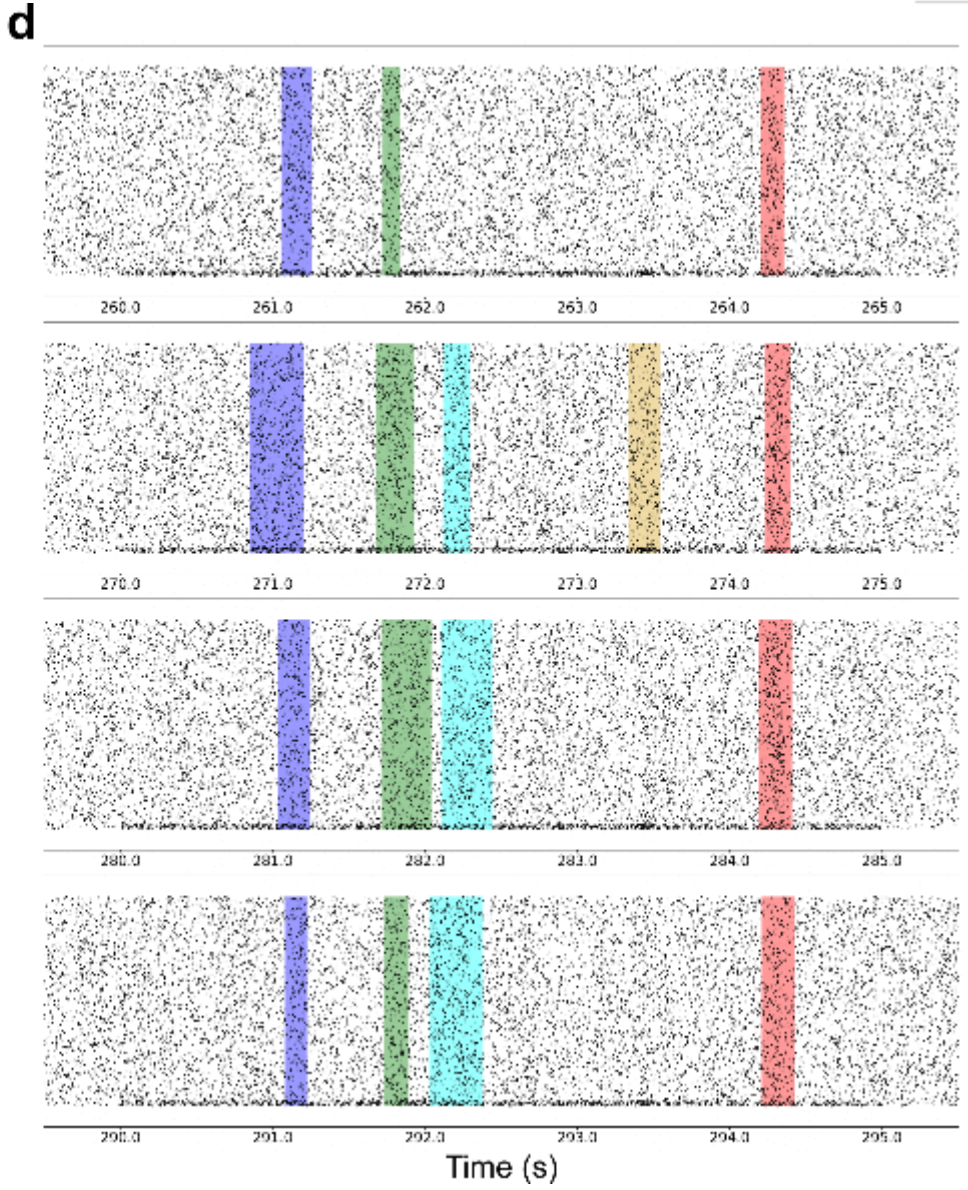
COINFLIPS



GNATs

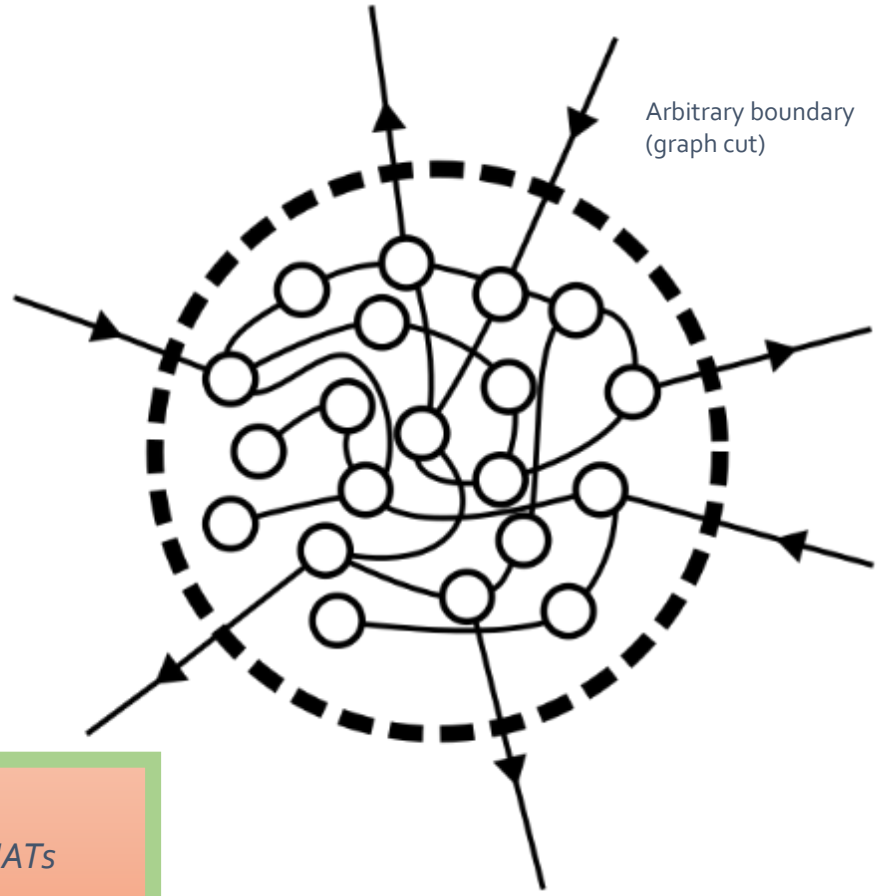
GNATs appear to provide an input-dependent sampling



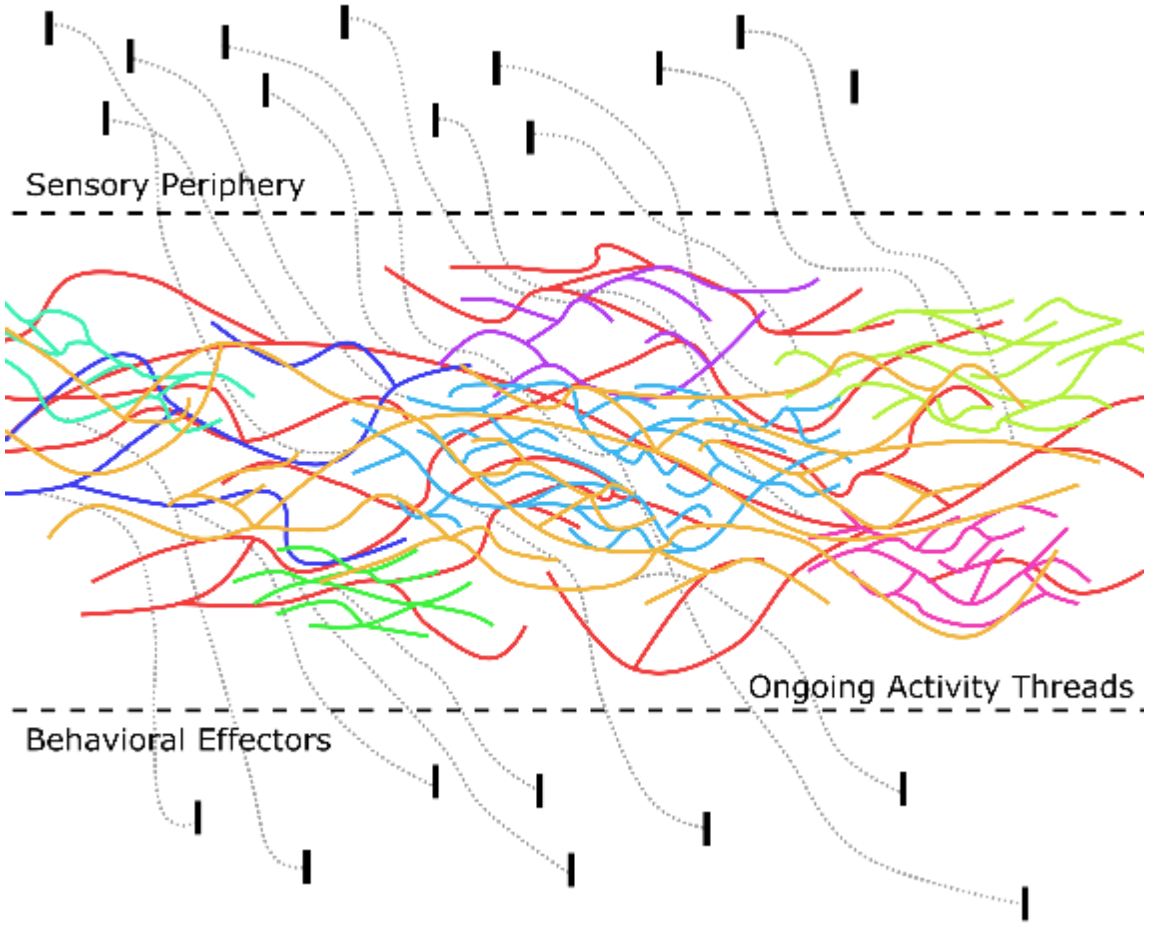


GNATs

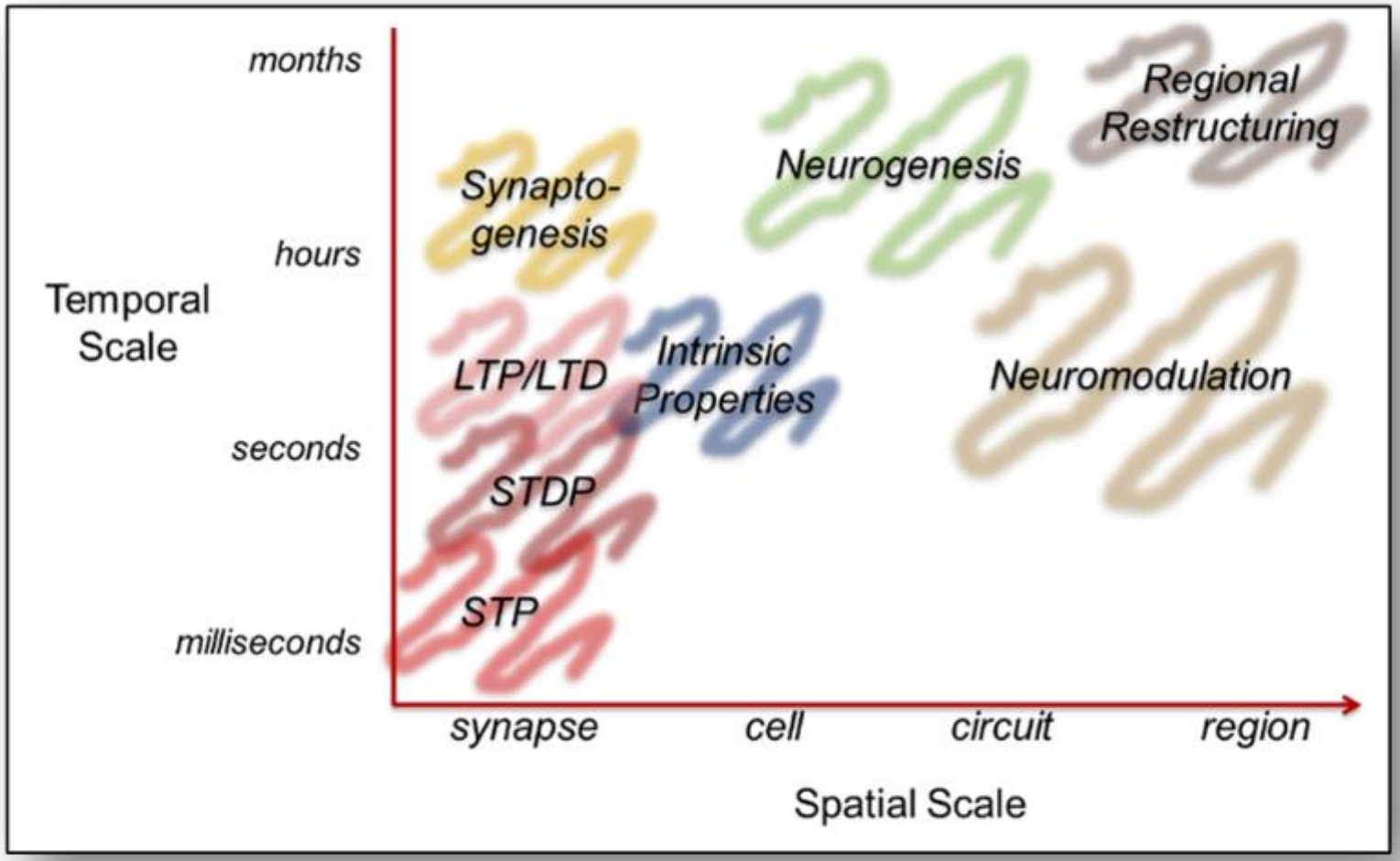
Towards GNAT-based computation?
…moving away from spikes to threads



GNATs



Beyond dynamics…
The brain is learning at *all* time and spatial scales



Brain Inspiration

Aimone, CACM 2019

COINFLIPS

A concrete future direction:

Brain-inspired systems that embrace stochasticity

COINFLIPS

We are benefitting from 70 years of microelectronics that embrace *deterministic* components to solve *deterministic* problems

COINFLIPS sees an opportunity to embrace *stochastic* computing to solve *uncertainty* problems

Today's computers emulate uncertainty by using pseudo-random number generation

COINFLIPS

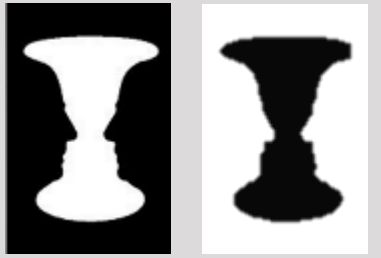
"Any one who considers arithmetical methods of producing random digits is, of course, in a state of sin."

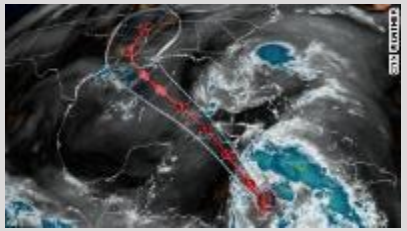
John von Neumann, 1951

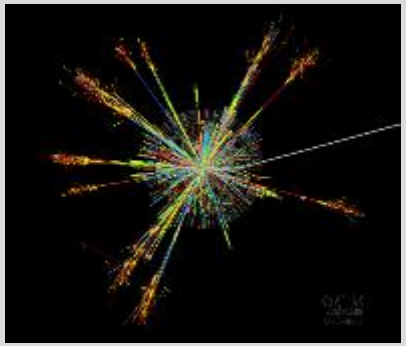
70 years later…

- Pseudo-RNGs can be quite effective, and do offer some advantages in verification, etc.
- But they are expensive, and when they go wrong the implications can be disastrous

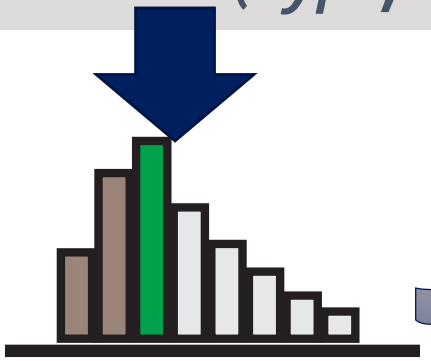
COINFLIPS aims to integrate true random number generators using stochastic devices into neuromorphic architectures





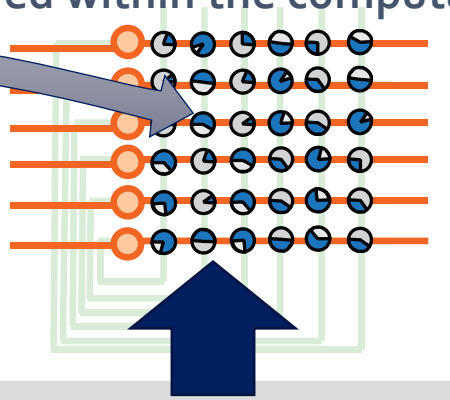


Improved Random Number Generation
(Type, Quantity, Quality)



Sample a random number from the *exact* distribution we require

And sample that number *where* it is needed within the computation



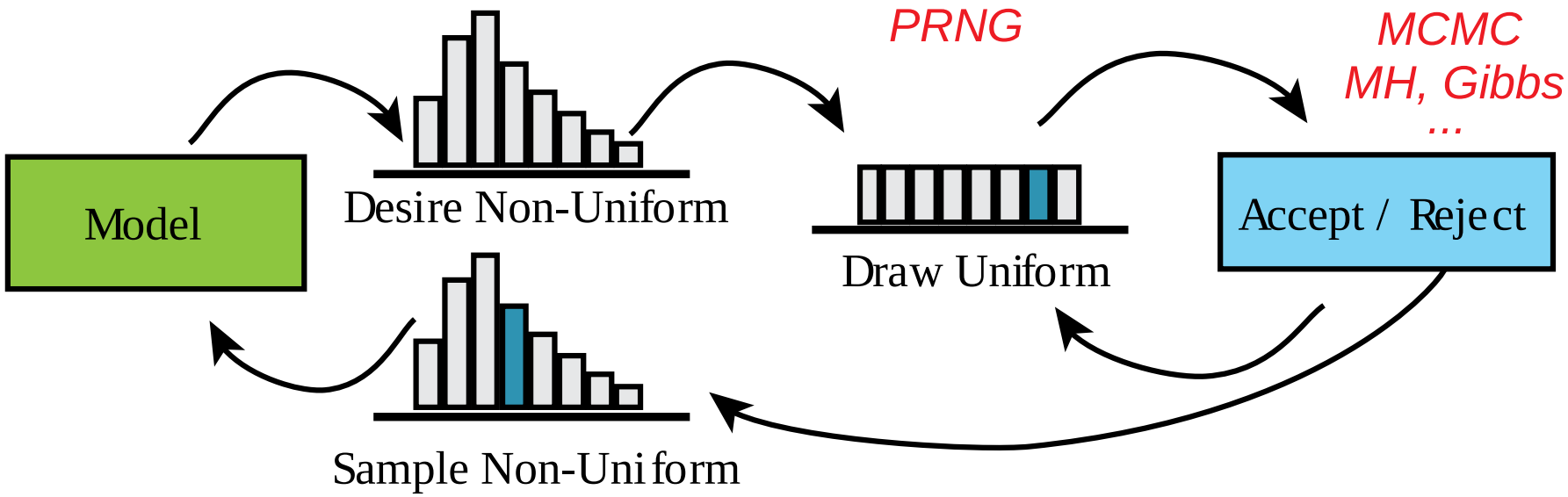
Neuromorphic architecture that integrates ubiquitous stochastic devices with computing and memory

COINFLIPS aims to improve both speed and energy of probabilistic computing applications

Evaluate opportunity of a probabilistic computing paradigm

COINFLIPS

Today

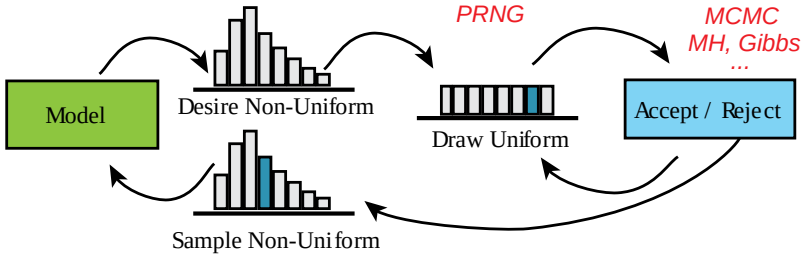


COINFLIPS

Evaluate opportunity of a probabilistic computing paradigm

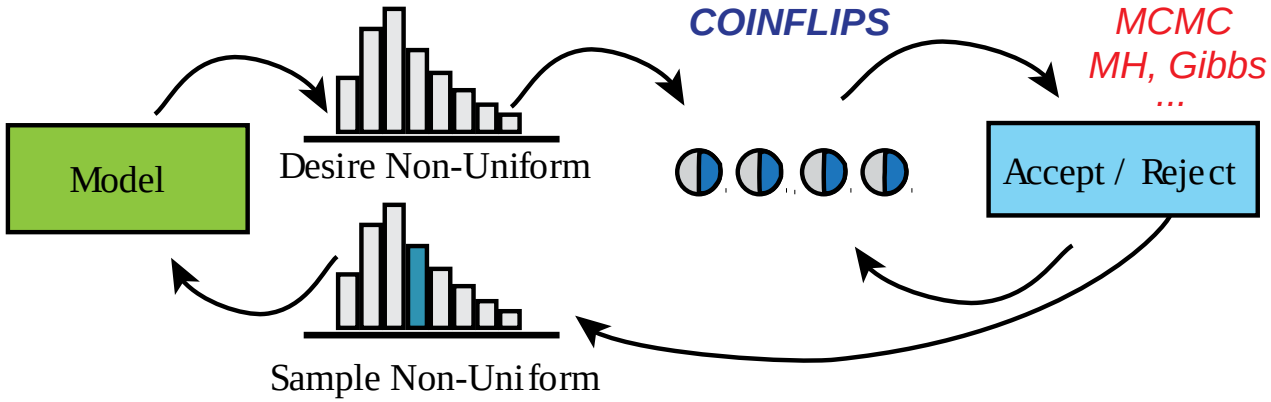
COINFLIPS

Today



COINFLIPS

Future?

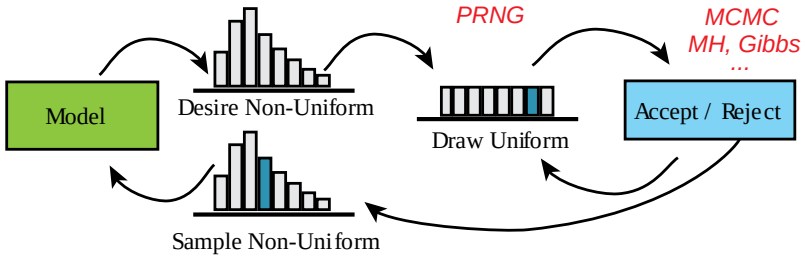


Step 1: Draw suitable uniform RNs from hardware

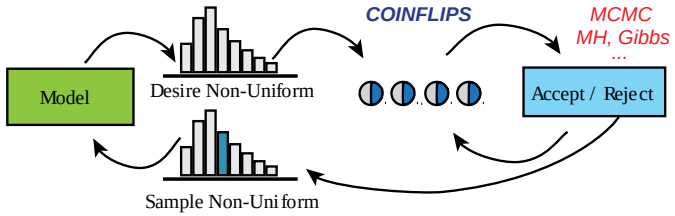
Evaluate opportunity of a probabilistic computing paradigm

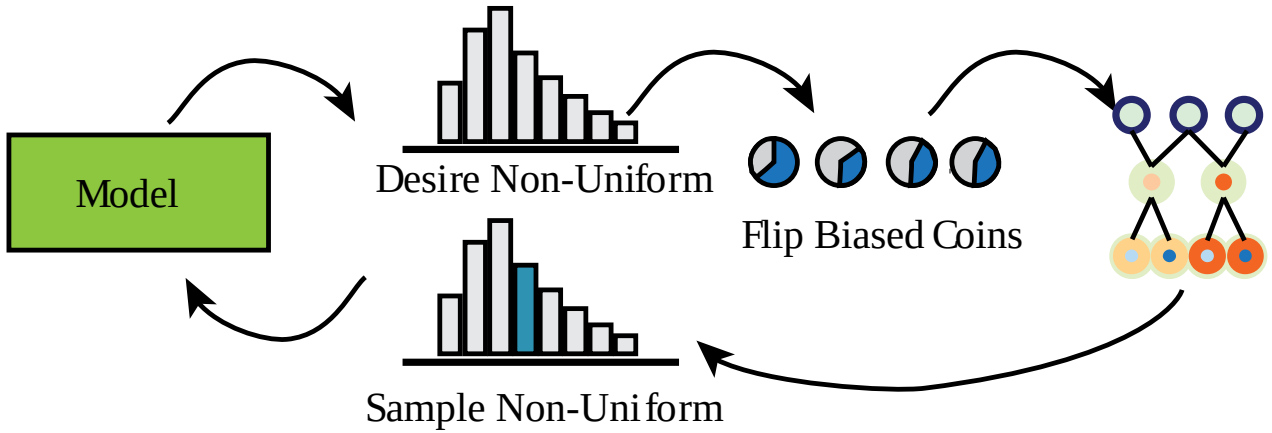
COINFLIPS

Today



Future?





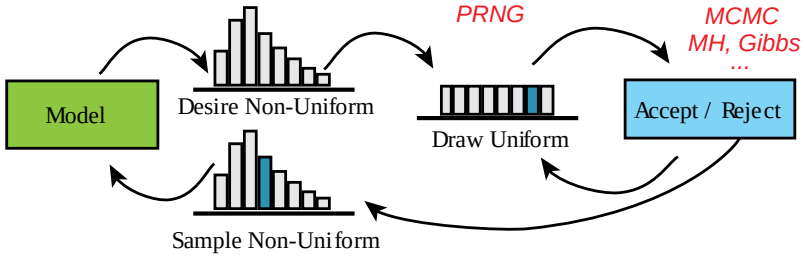
Step 2: Draw suitable model-specific RNs from hardware

COINFLIPS

Evaluate opportunity of a probabilistic computing paradigm

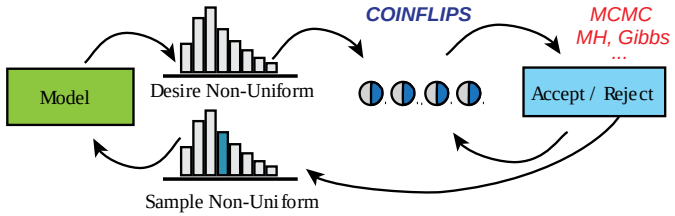
COINFLIPS

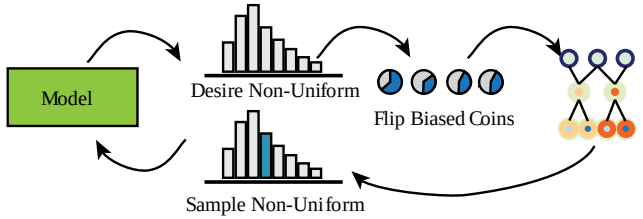
Today

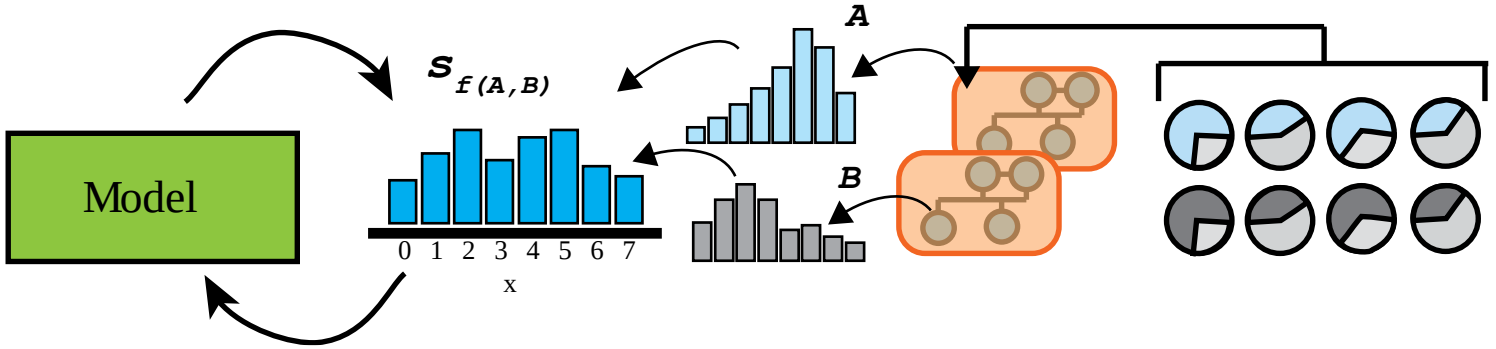


COINFLIPS

Future?

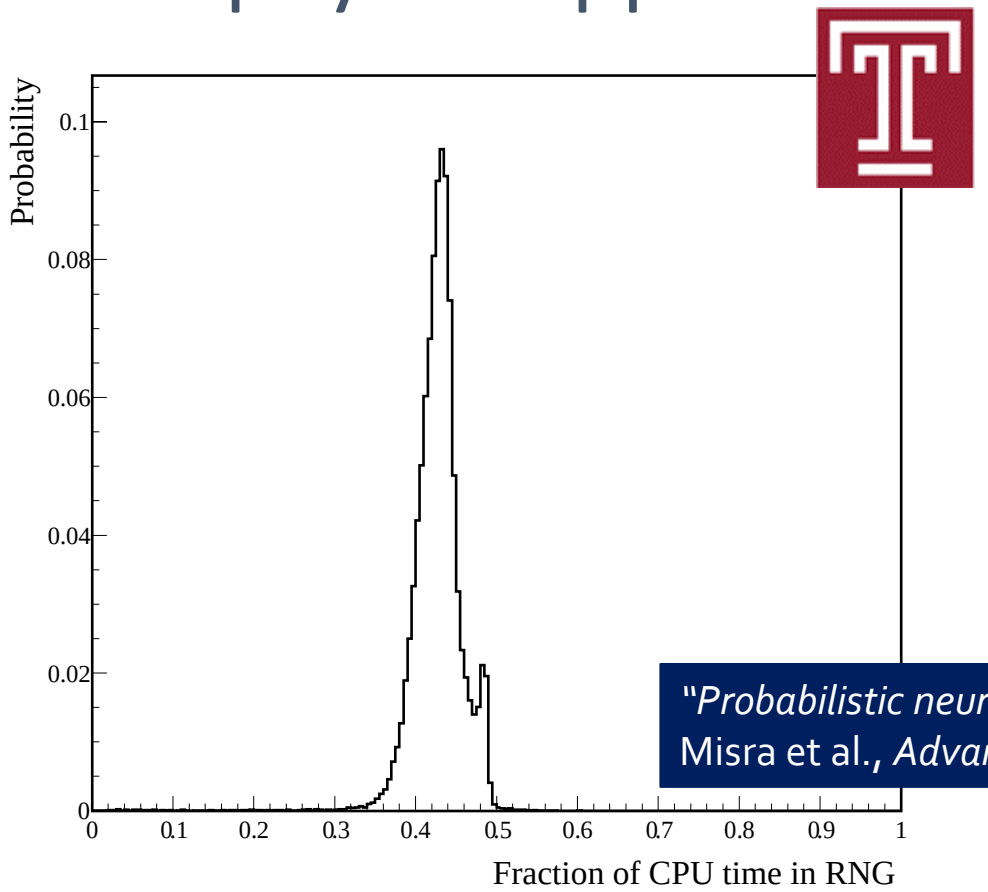


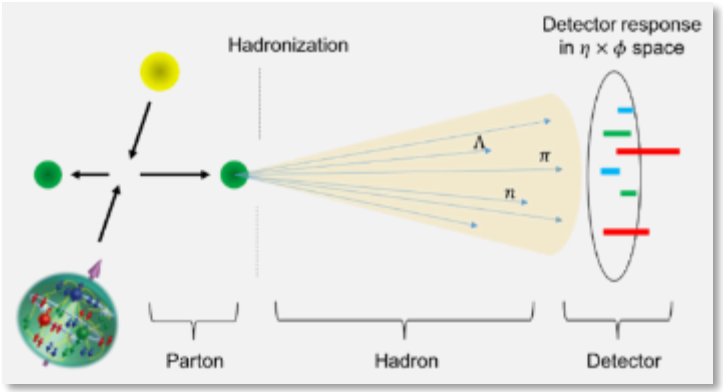




Step 3: Integrate hardware-enabled random sampling into computation

Random numbers are a limiting computational cost for some nuclear physics applications





"Probabilistic neural computing with stochastic devices"
Misra et al., *Advanced Materials*. 2023

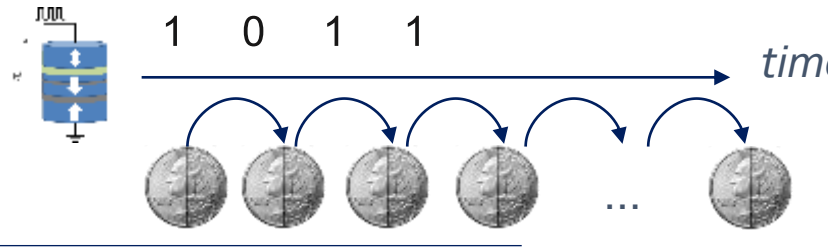
Particle Physics Demonstration

Half of computational cost is generating a uniform random number, which then must be transformed

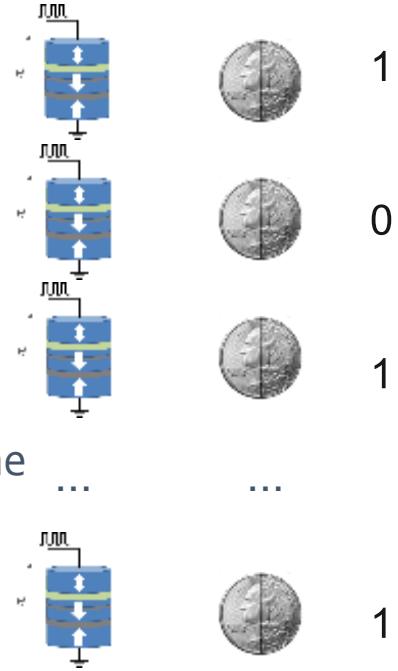
Sampling a uniform distribution (generate random number 0-1)

COINFLIPS

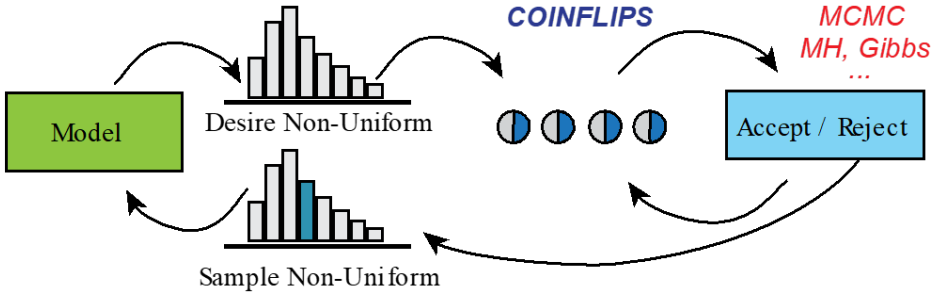
Two options for true RNG



One device flipping over time



Many devices flipping one time

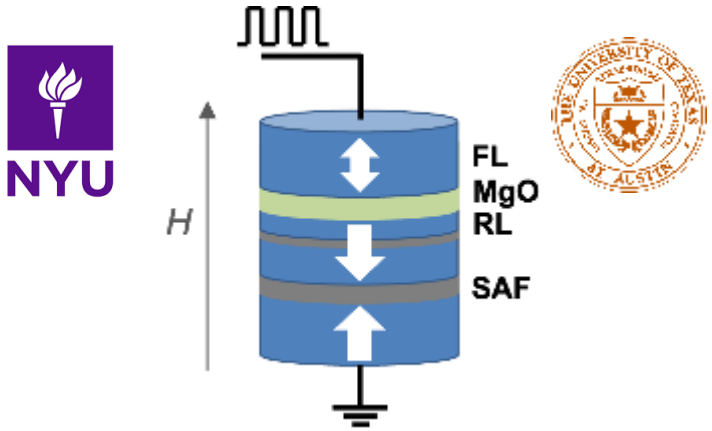


Tunable Stochastic Devices

Fair coinflip device example – Magnetic Tunnel Junction (MTJ)

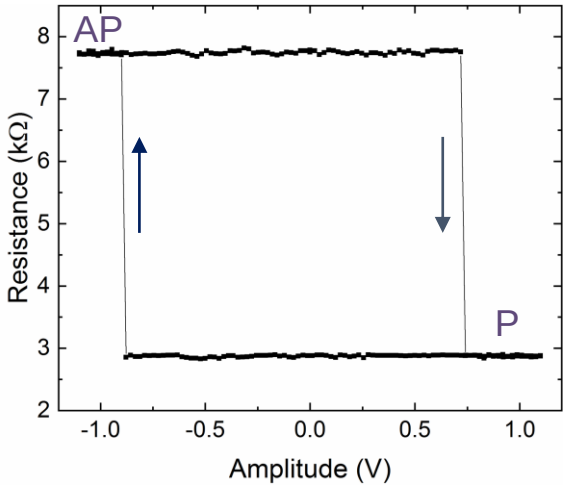
COINFLIPS

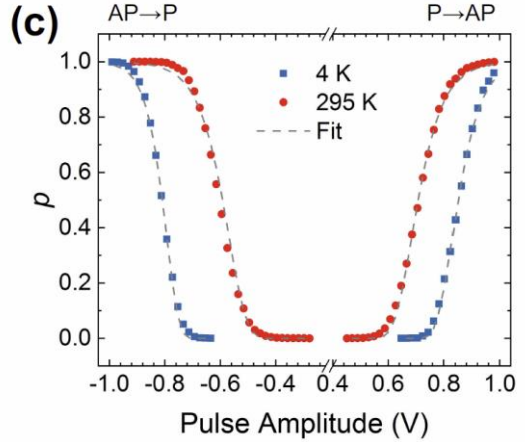
MTJ Coinflip device

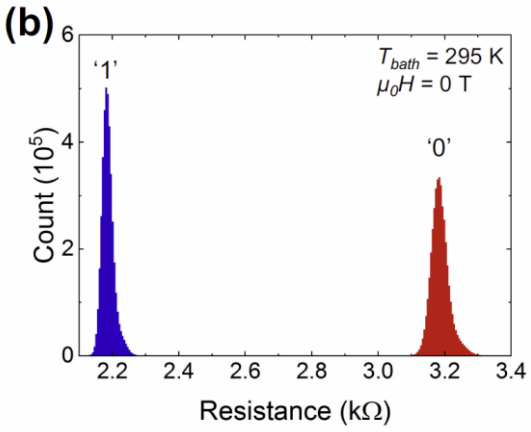


40 nm circular pMTJ with CoFeB/W/CoFeB composite free layer

Reset – set metastable state – read







"*Spin hall effect magnetic tunnel junction coinflips*"
Reim et al., *Submitted arXiv 2209.01480*

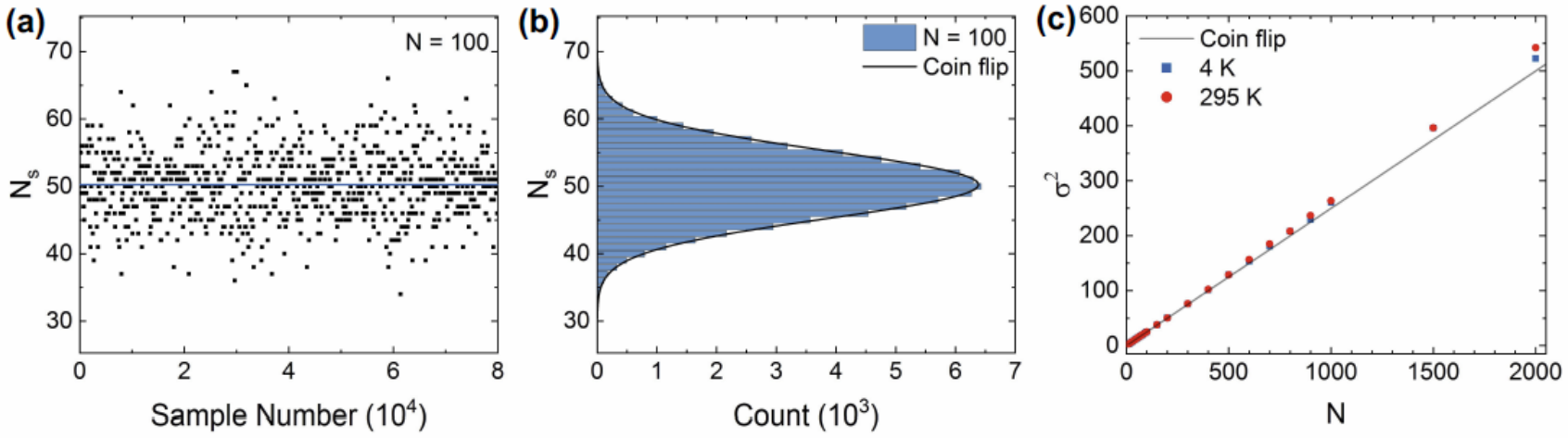
Tunable Stochastic Devices

What makes one coinflip device better than another?

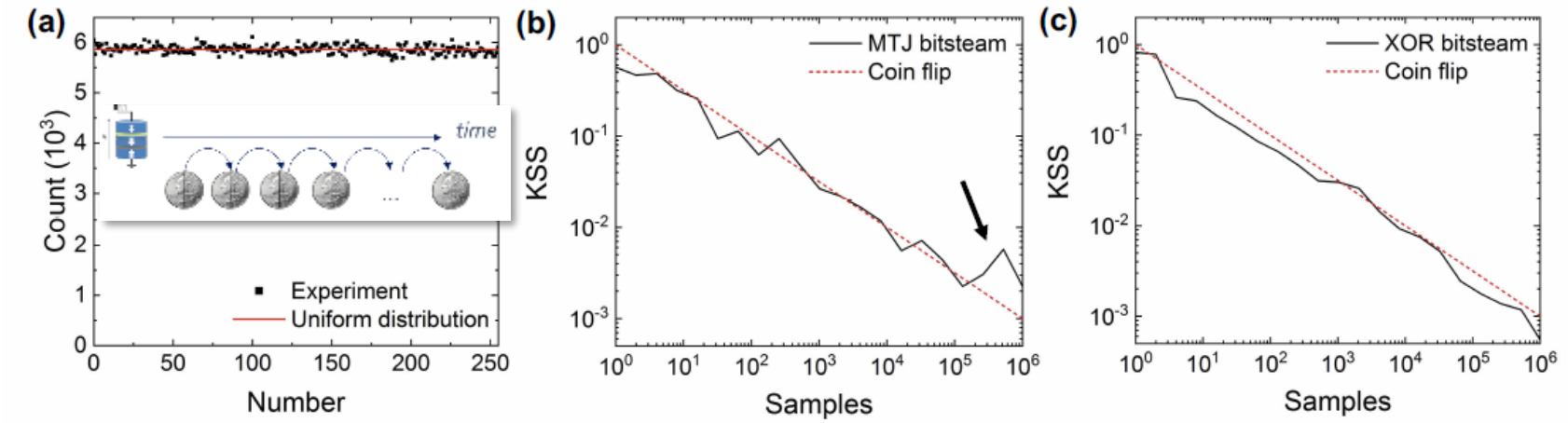
Quality of coinflip directly tied to quality of sample

COINFLIPS

Blocks of 100 random coinflips show expected distribution of random samples

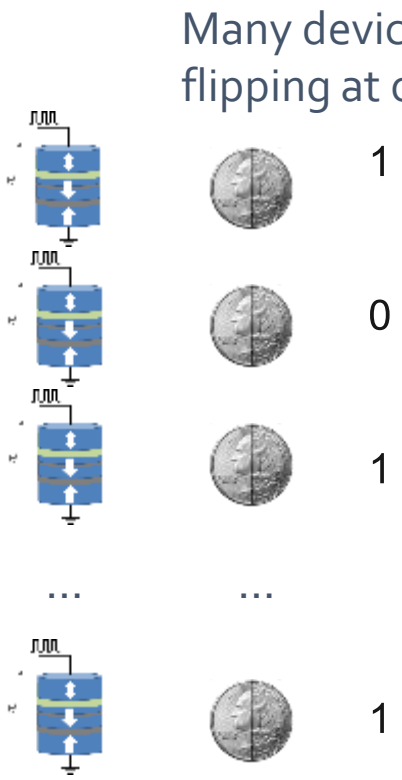


Generating 8-bit (integers from 0 – 255) from coinflips produces good random samples



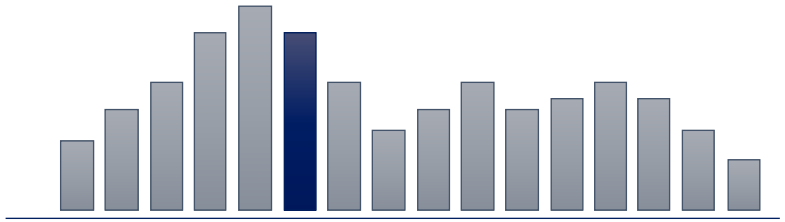
Tunable Stochastic Devices

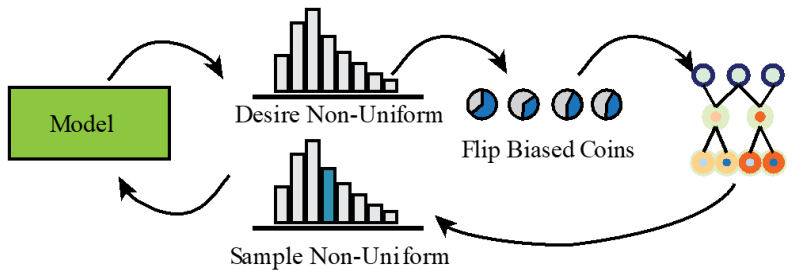
AI-guided design of neuromorphic circuits – arbitrary distribution



Fair coins for *uniform* distribution

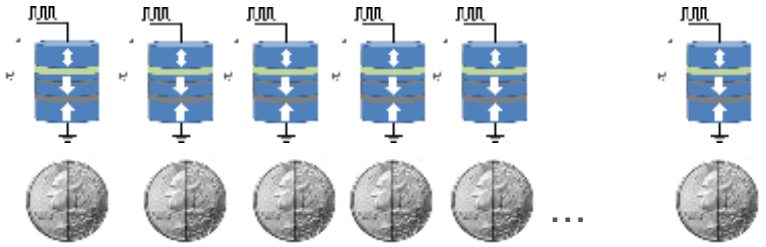
Biased coins for *non-uniform* distribution?





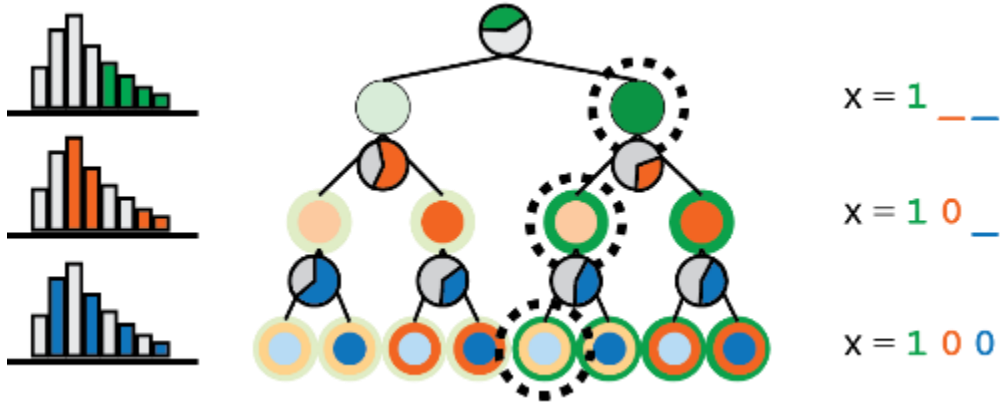
Probabilistic Circuits and Architectures

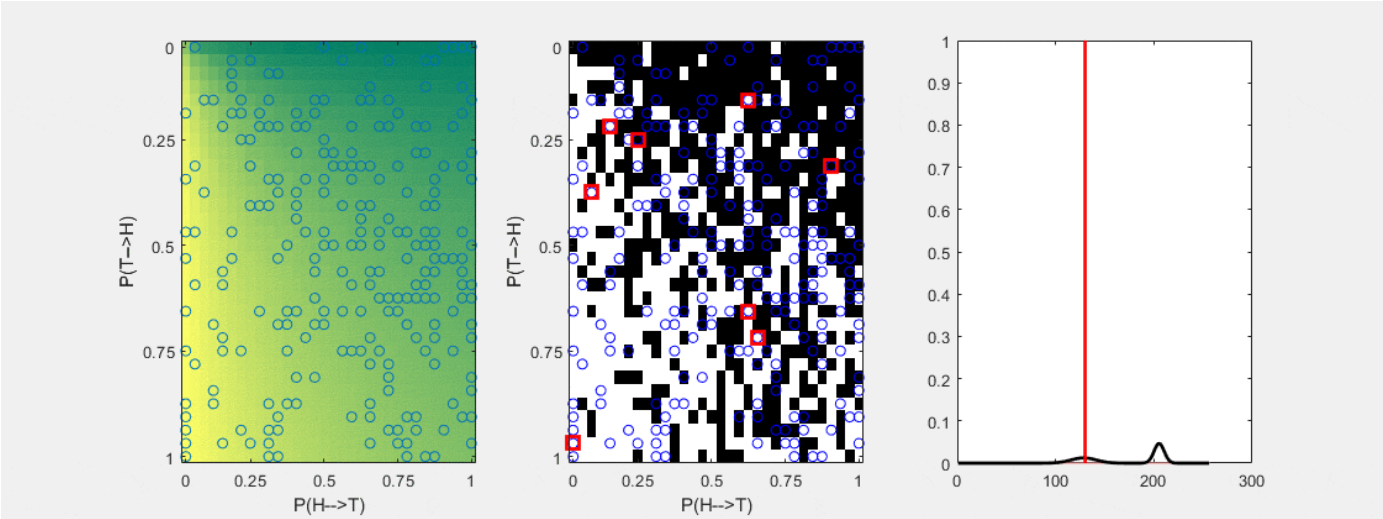
Mapping Coinflips to Arbitrary Distributions



Many devices
flipping at one time

Naively, we can expand a binary tree with probabilities to describe any distribution



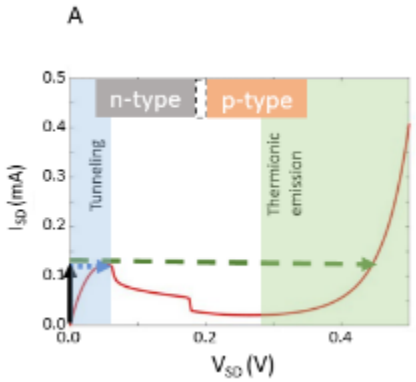


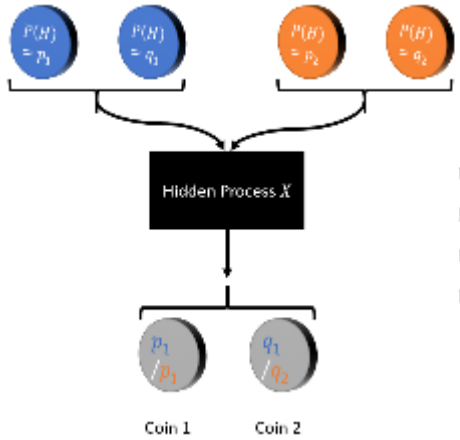
Simulation...

Probabilistic Neural Theory and Algorithms

AI-guided design of neuromorphic circuits – making arbitrary distributions efficient

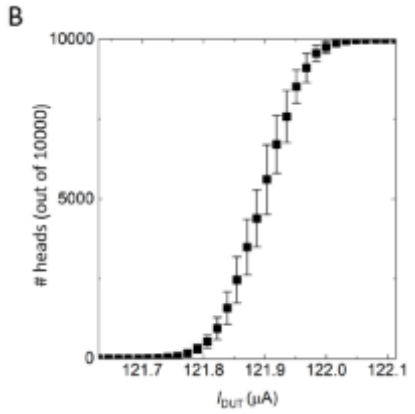
COINFLIPS

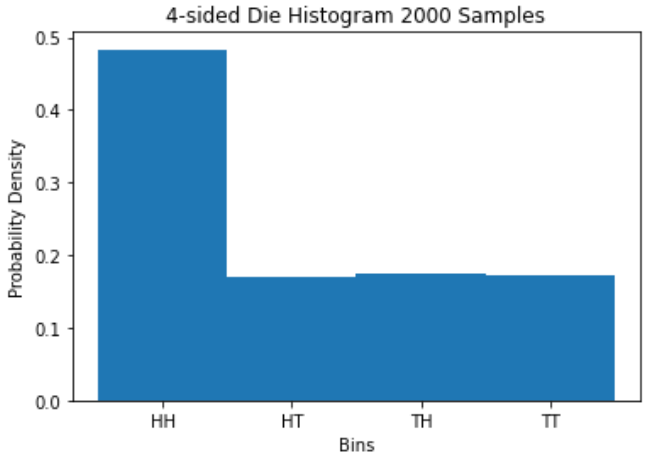




Target Distribution

$$\mathbf{P}[\text{Coin 1} = H \text{ and Coin 2} = H] = \frac{1}{2}$$
$$\mathbf{P}[\text{Coin 1} = H \text{ and Coin 2} = T] = \frac{1}{6}$$
$$\mathbf{P}[\text{Coin 1} = T \text{ and Coin 2} = H] = \frac{1}{6}$$
$$\mathbf{P}[\text{Coin 1} = T \text{ and Coin 2} = H] = \frac{1}{6}$$





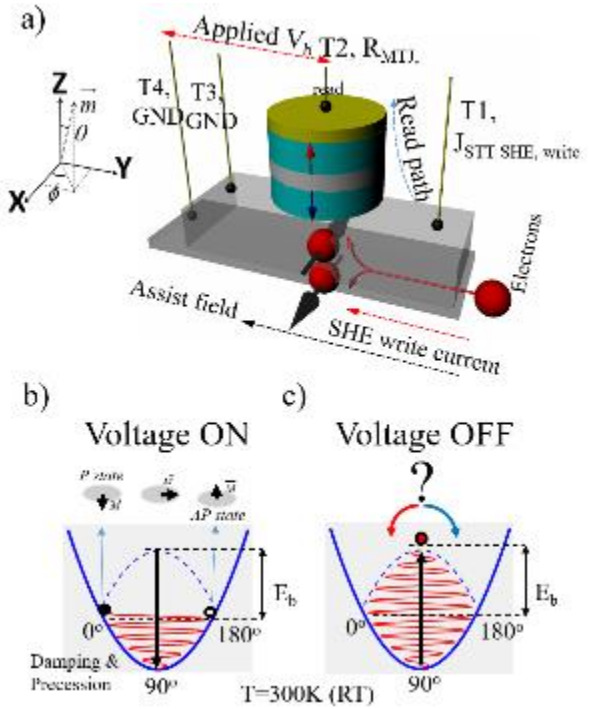
"AI-enhanced Codesign for Probabilistic Neural Circuits"
Cardwell SG et al. *2022 International Conference on Rebooting Computing*

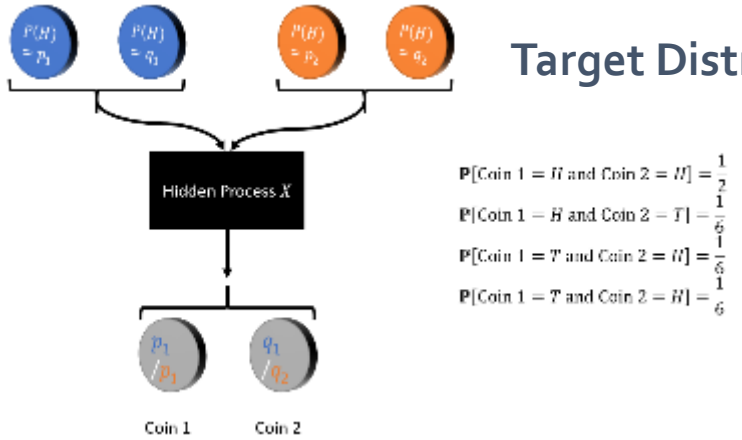
Probabilistic Circuits and Architectures

Sampling arbitrary distributions needs *weighted* coinflip devices

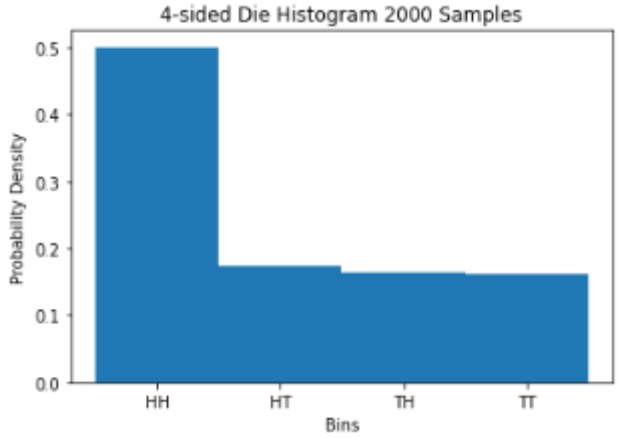
AI-guided design of neuromorphic circuits – making arbitrary distributions efficient

COINFLIPS





Target Distribution



"AI-enhanced Codesign for Probabilistic Neural Circuits"
Cardwell SG et al. *2022 International Conference on Rebooting Computing*

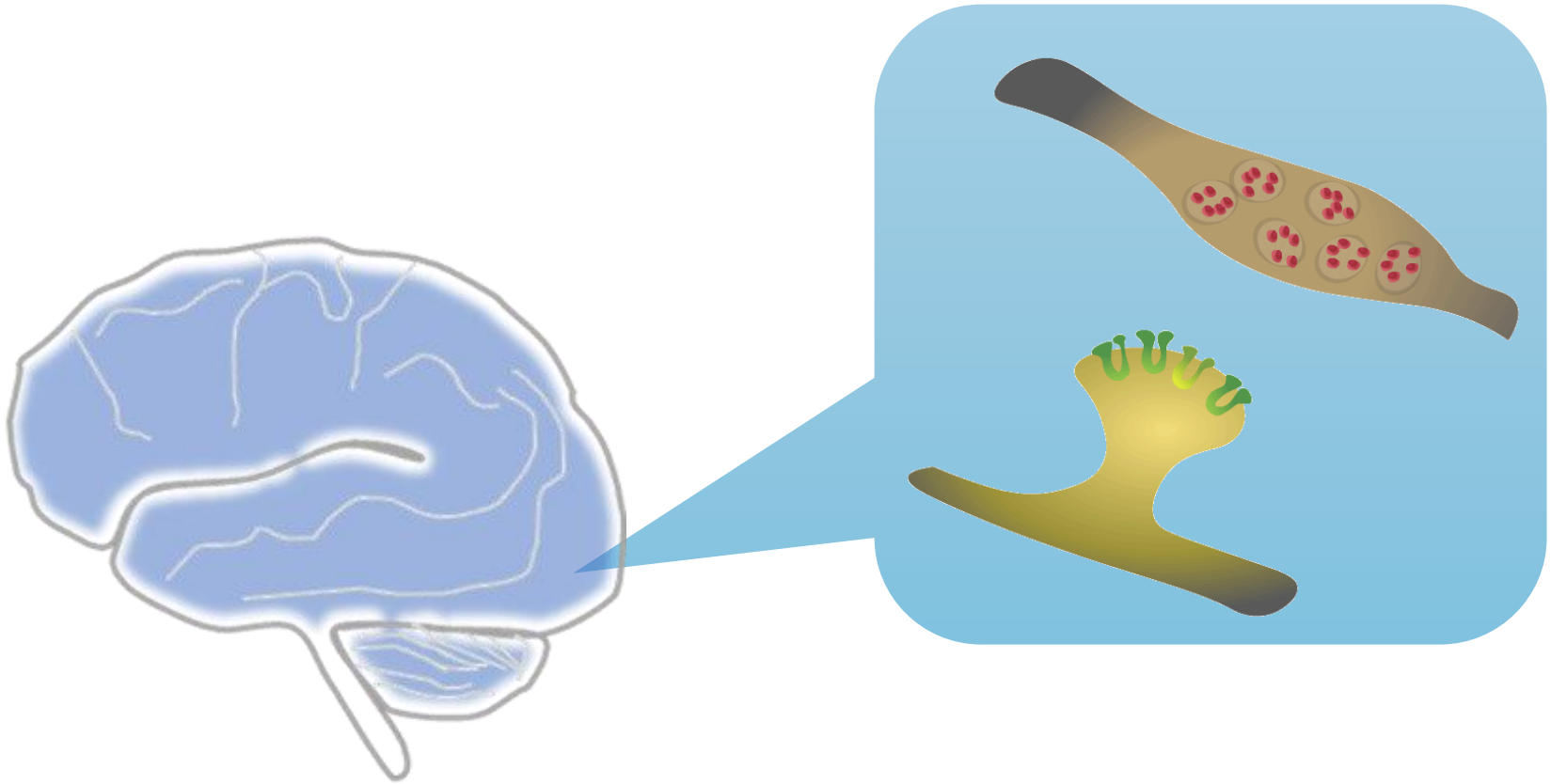
Probabilistic Circuits and Architectures

Sampling arbitrary distributions needs *weighted* coinflip devices

Probabilistic Neuromorphic Algorithms

COINFLIPS

So what happens if we put stochastic devices with neurons?

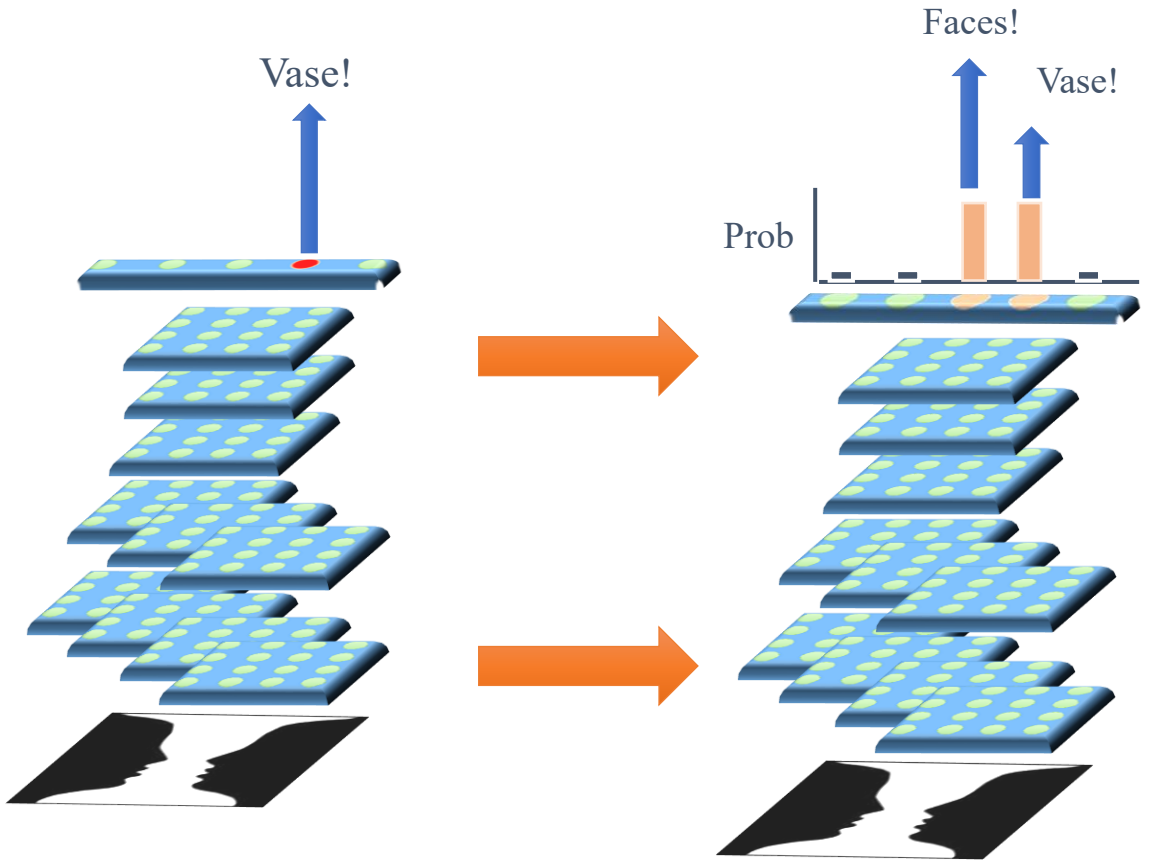


Probabilistic Neural Theory and Algorithms

Probabilistic Neuromorphic Algorithms

COINFLIPS

So what happens if we put stochastic devices with neurons?



We are going to do this! Need to advance stochastic devices, probabilistic circuits, and Bayesian algorithms

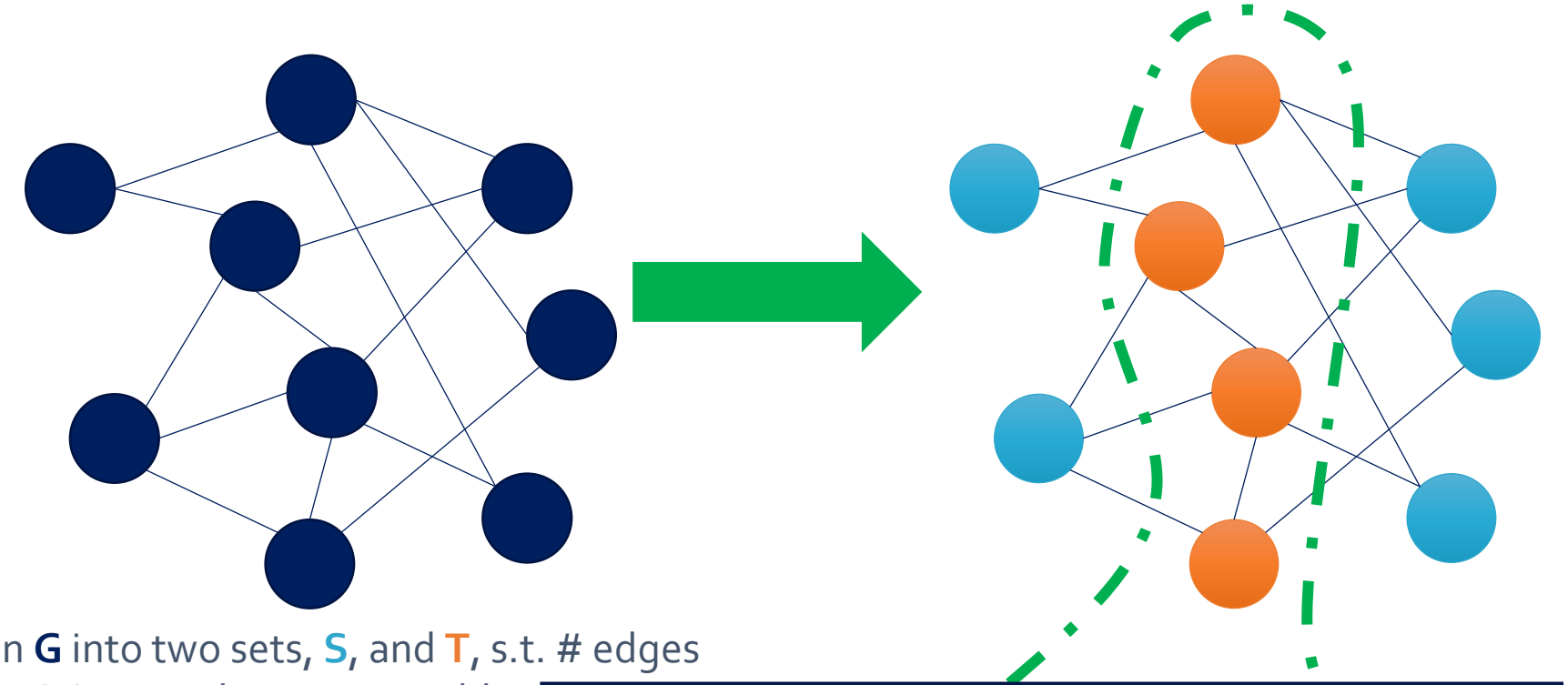
Probabilistic Neural Theory and Algorithms

Probabilistic Neuromorphic Algorithms

COINFLIPS

So what happens if we put stochastic devices with neurons?

Approximate Algorithms for "Maximum Cut" of Graphs



Probabilistic Neural Theory and Algorithms

Partition **G** into two sets, **S**, and **T**, s.t. # edges between S & T is as large as possible

"Stochastic Neuromorphic Circuits for Solving MAXCUT" Theilman et al., IPDPS 2023; Arxiv 2210.02588

WHY MAXCUT?

- NP-hard
- Central theoretical testbed in discrete optimization (Commander 2008)
- Practical applications
  - VLSI design (Pinter 1984, Barahona et al. 1988)
- Stochastic approximation algorithms exist with practical performance guarantees
  - Led to stochastic approximations for graph coloring, satisfiability, etc.

Probabilistic Neural Theory and Algorithms

Goemans-Williamson maxcut approximation algorithm

Discrete optimization problem:

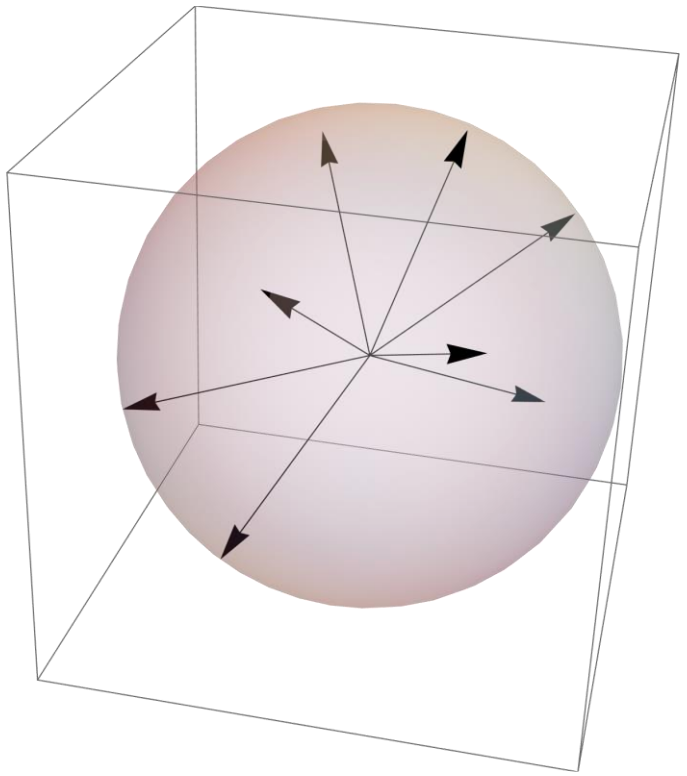
maximize $C = \frac{1}{4}\sum_{ij} A_{ij}(1 - y_i y_j)$

such that $y_i \in \{-1, 1\}$

Replace integer y_i with unit vectors:

maximize $\tilde{C} = \frac{1}{4}\sum_{ij} A_{ij}(1 - v_i \cdot v_j)$

such that $\|v_i\| = 1$



Probabilistic Neural Theory and Algorithms

Goemans and Williamson 1995

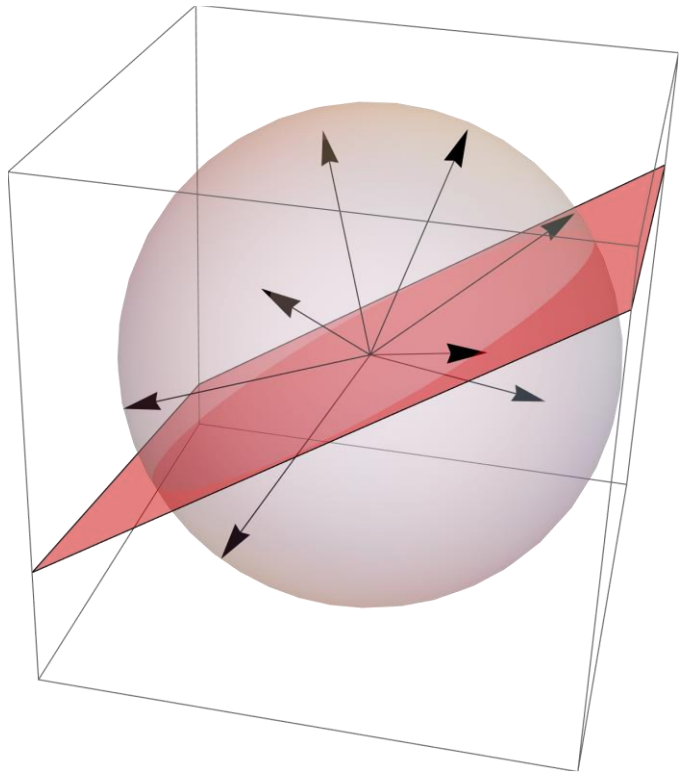
Goemans-Williamson maxcut approximation algorithm

COINFLIPS

Discrete optimization problem:

maximize $C = \frac{1}{4}\sum_{ij} A_{ij}(1 - y_i y_j)$

such that $y_i \in \{-1, 1\}$



Replace integer y_i with unit vectors:

maximize $\tilde{C} = \frac{1}{4}\sum_{ij} A_{ij}(1 - v_i \cdot v_j)$

such that $\|v_i\| = 1$

Choose random unit vector r, sample graph cut:

$$y_i = \mathrm{sgn}(r \cdot v_i)$$

Probabilistic Neural Theory and Algorithms

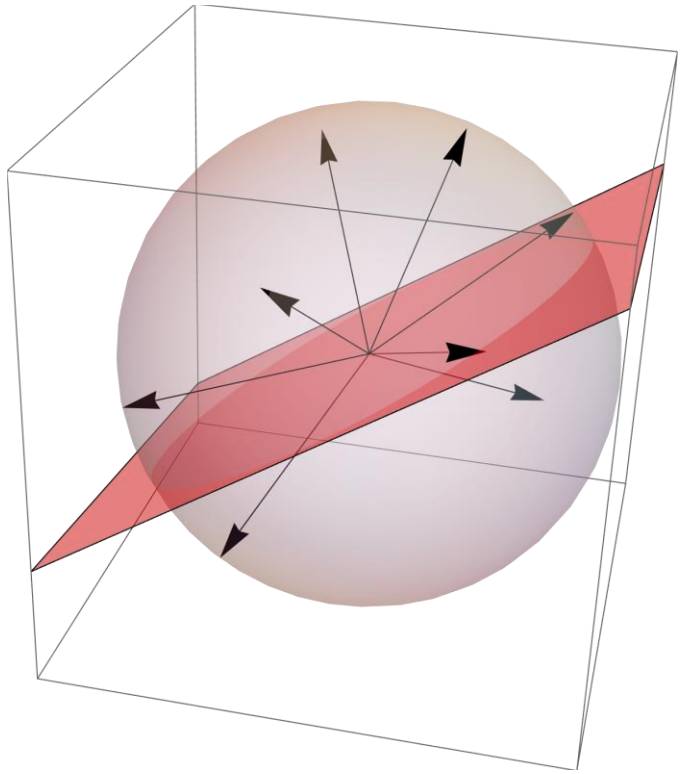
Approximation ratio:

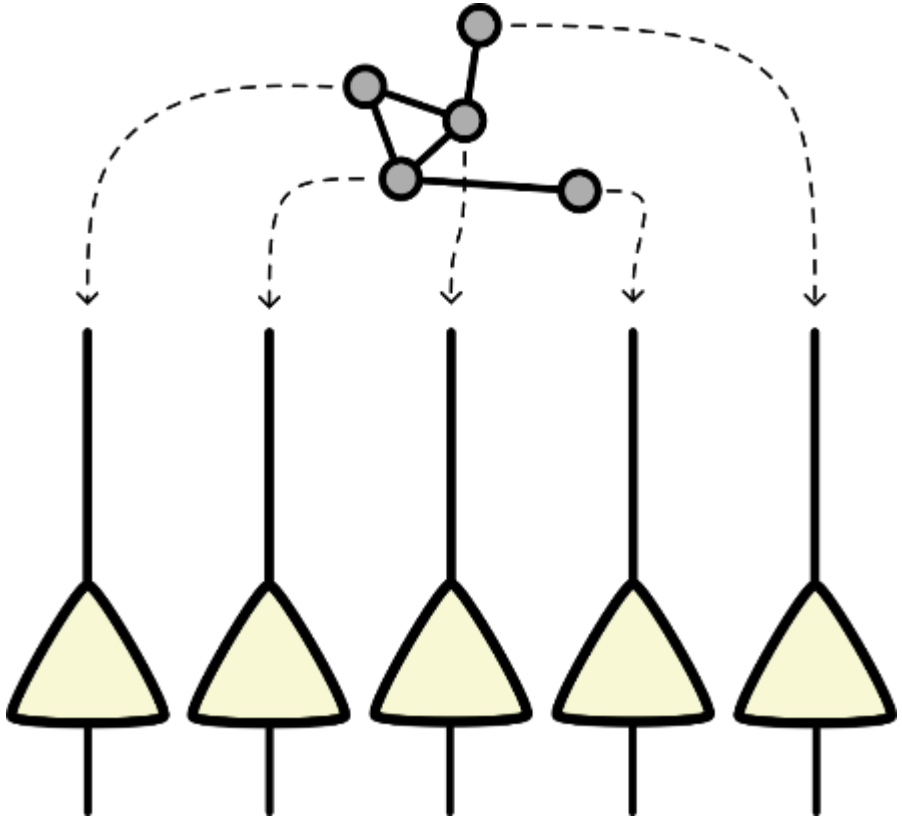
Expected cut weight vs. absolute maximum

$$\frac{\mathbb{E}[C]}{C_{max}} = 0.878$$

Goemans and Williamson 1995

Towards neuromorphic Goemans-Williamson





Probabilistic Neural Theory and Algorithms

Statistics of leaky integrate-and-fire neurons

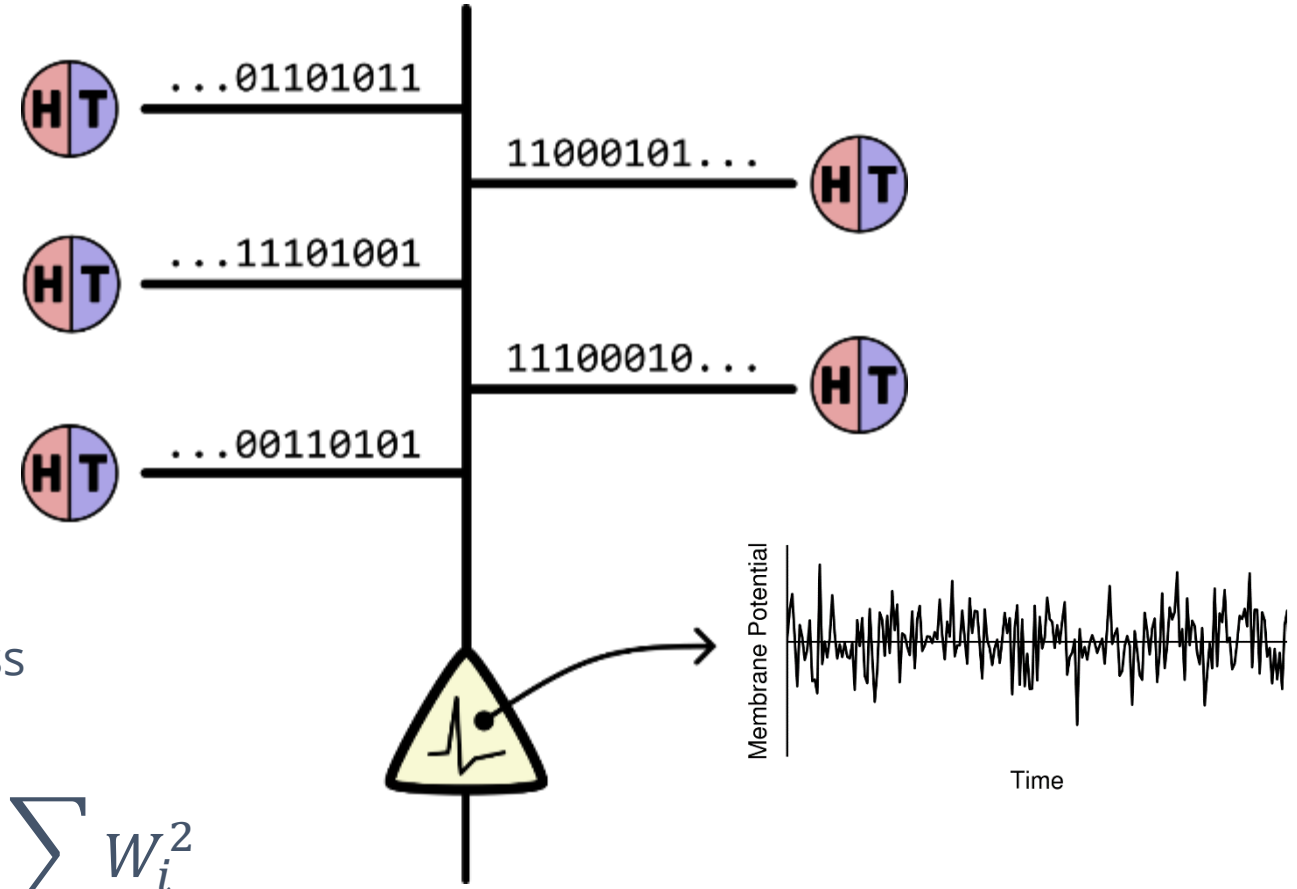
LIF membrane potential dynamics

$$C\frac{dV}{dt} = -\frac{V}{R} + \alpha\sum W_j s_j$$

- Leak current stabilizes mean membrane potential
- Central Limit Theorem guarantees V fluctuations approximate a Gaussian process

$$\mu(V) = \frac{\alpha R}{2}\sum_i W_i \qquad \sigma^2(V) = \frac{\alpha^2 R}{8C}\sum_i W_i^2$$

Probabilistic Neural Theory and Algorithms



Statistics of shared presynaptic input

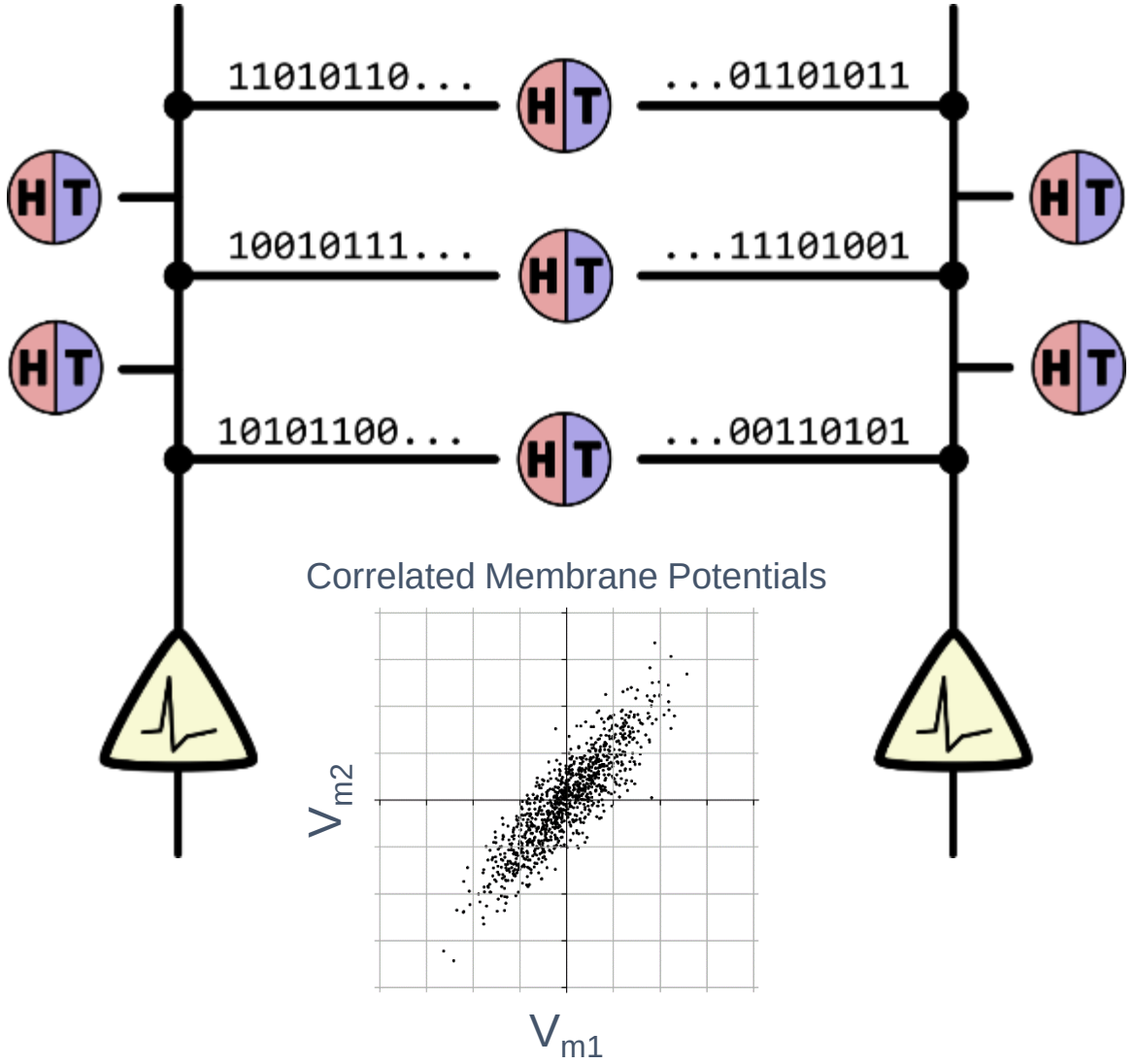
Shared synaptic input induces correlations between LIF membrane potentials

$$Cov(V_i, V_j) = \frac{\alpha^2 R}{2C} W_{ia} W_{jb} Cov(s_a, s_b)$$

$$Cov(s_a, s_b) = \frac{1}{4} \delta_{ab}$$

$$Cov(V_i, V_j) = \frac{\alpha^2 R}{8C} W_i \cdot W_j$$

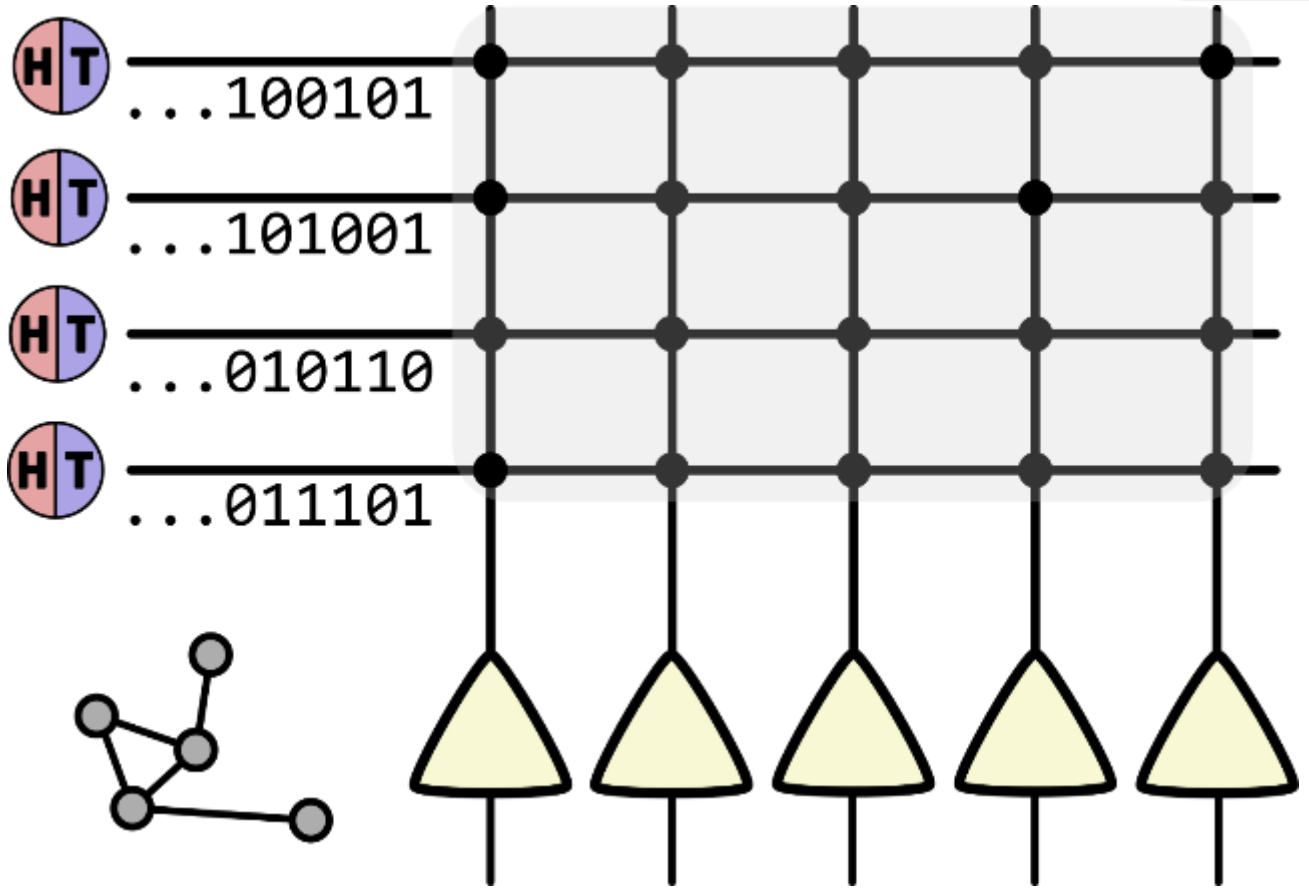
Probabilistic Neural Theory and Algorithms



Neuromorphic Goemans-Williamson sampling

COINFLIPS

Assign one LIF neuron to each graph vertex



Probabilistic Neural Theory and Algorithms

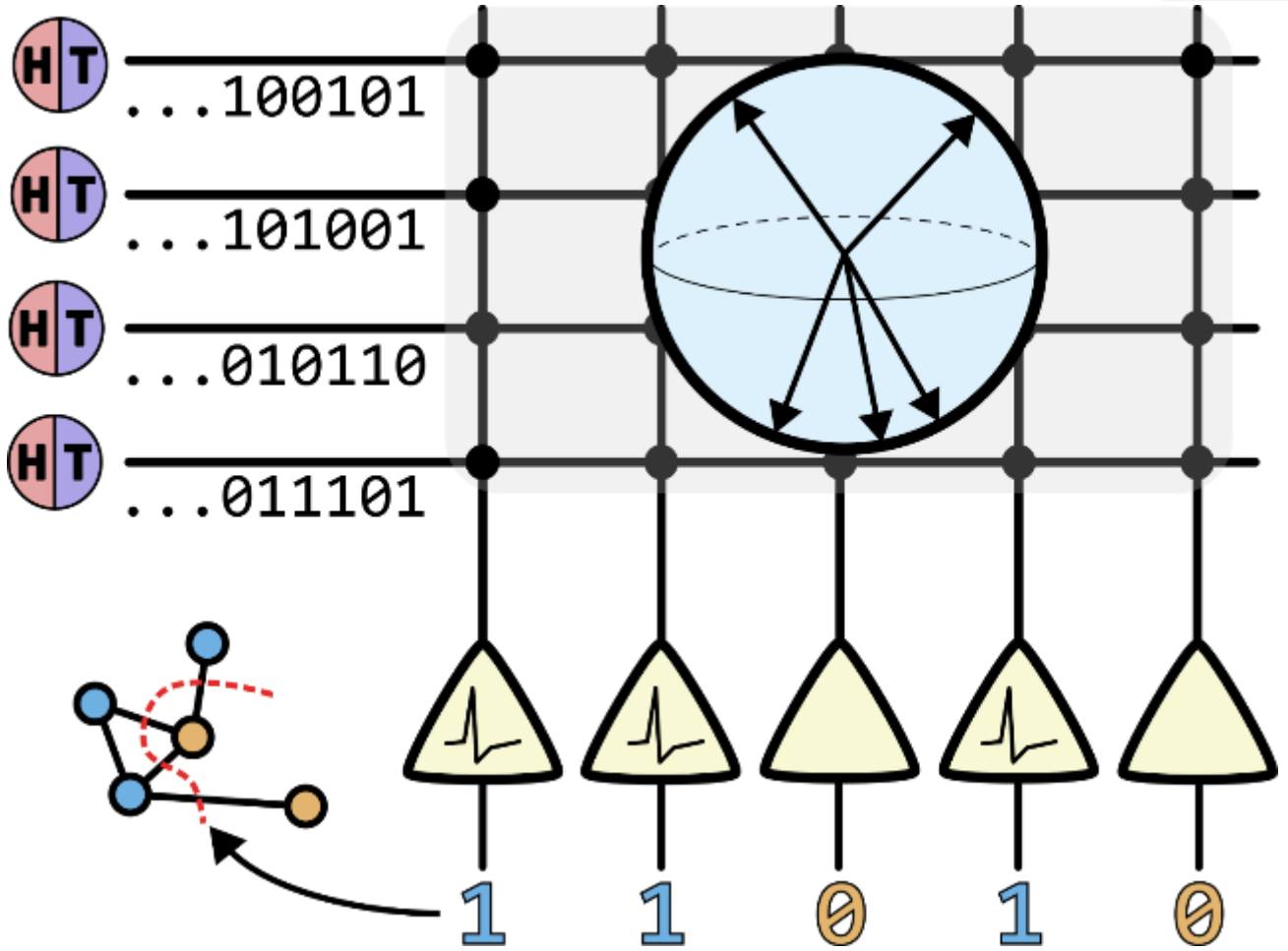
Neuromorphic Goemans-Williamson sampling

Assign one LIF neuron to each graph vertex

Set COINFLIPS -> LIF weights proportional to Goemans-Williamson vectors

$$Cov(V_i, V_j) = \frac{\alpha^2 R}{8C} W_i \cdot W_j$$

Probabilistic Neural Theory and Algorithms



"Spiking" threshold turns fluctuations into graph cuts

SPECTRAL METHODS FOR MAXCUT

COINFLIPS

- Trevisan's algorithm (with Soto's improvement): randomly threshold the minimum eigenvector of the normalized graph adjacency matrix

  Trevisan 2012, Soto 2015

  - Approximation ratio: 0.614

- Simplified spectral algorithm (Mirka and Williamson 2022): keep the threshold fixed at 0.

  - Approximation ratio unknown
  - Works well in practice

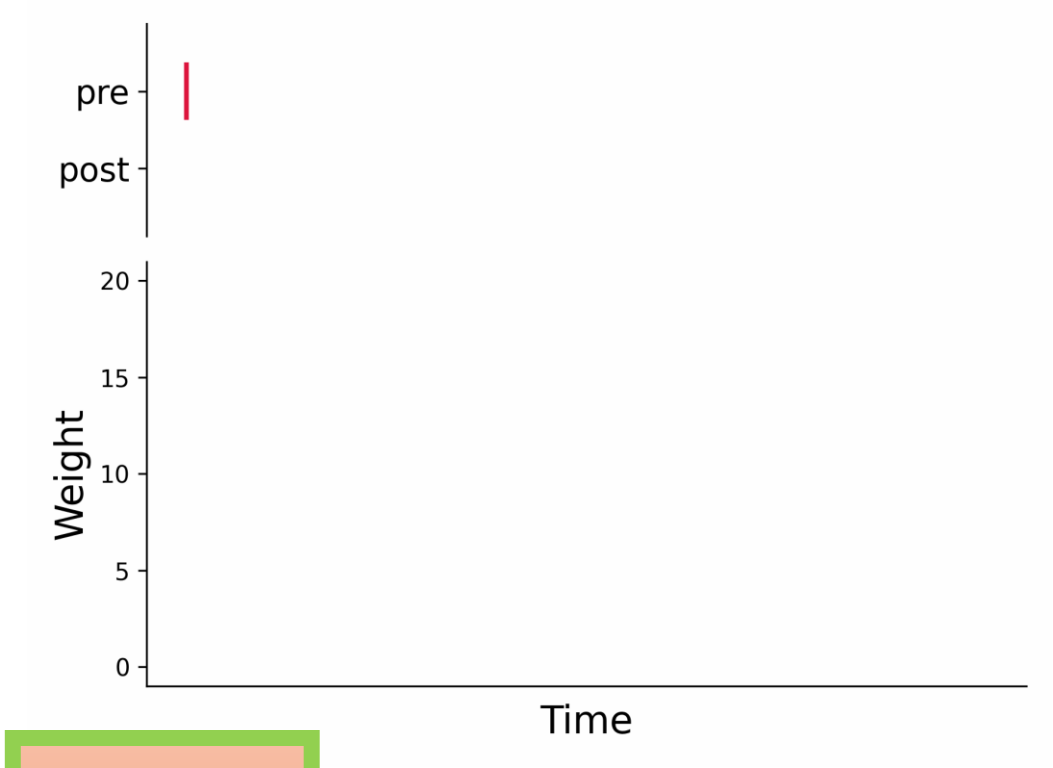
Neuromorphic approach:
Spectrally decompose LIF-generated covariance matrix

Probabilistic Neural Theory and Algorithms

Graph	Greedy	Trevisan	Simple Spectral	Sweep Cuts	SDP
G(50,0.1)	8.700×10^1	$\mathbf{9.600 \times 10^1}$	9.400×10^1	9.500×10^1	9.200×10^1
G(50,0.25)	1.970×10^2	2.060×10^2	2.060×10^2	2.080×10^2	$\mathbf{2.100 \times 10^2}$
G(50,0.5)	3.480×10^2	$\mathbf{3.600 \times 10^2}$	3.560×10^2	$\mathbf{3.600 \times 10^2}$	$\mathbf{3.600 \times 10^2}$
G(50,0.75)	5.140×10^2	5.140×10^2	4.990×10^2	5.190×10^2	$\mathbf{5.240 \times 10^2}$
G(100,0.1)	3.210×10^2	3.290×10^2	3.420×10^2	$\mathbf{3.430 \times 10^2}$	3.290×10^2
G(100,0.25)	7.640×10^2	7.830×10^2	7.850×10^2	$\mathbf{7.880 \times 10^2}$	7.860×10^2
G(100,0.5)	1.351×10^3	1.363×10^3	1.346×10^3	$\mathbf{1.375 \times 10^3}$	1.361×10^3
G(100,0.75)	2.019×10^3	2.024×10^3	2.020×10^3	$\mathbf{2.026 \times 10^3}$	2.016×10^3
G(200,0.1)	1.212×10^3	$\mathbf{1.250 \times 10^3}$	1.234×10^3	1.242×10^3	1.211×10^3
G(200,0.25)	2.795×10^3	2.859×10^3	2.847×10^3	$\mathbf{2.861 \times 10^3}$	2.778×10^3
G(200,0.5)	5.388×10^3	5.420×10^3	5.412×10^3	$\mathbf{5.423 \times 10^3}$	5.326×10^3
G(200,0.75)	7.784×10^3	7.855×10^3	7.831×10^3	$\mathbf{7.875 \times 10^3}$	7.815×10^3
G(350,0.1)	3.556×10^3	3.582×10^3	3.639×10^3	$\mathbf{3.651 \times 10^3}$	3.611×10^3
G(350,0.25)	8.378×10^3	8.544×10^3	8.583×10^3	$\mathbf{8.585 \times 10^3}$	8.236×10^3
G(350,0.5)	1.623×10^4	1.627×10^4	1.643×10^4	$\mathbf{1.649 \times 10^4}$	1.603×10^4
G(350,0.75)	2.356×10^4	$\mathbf{2.378 \times 10^4}$	2.374×10^4	2.374×10^4	2.353×10^4
G(500, .1)	7.155×10^3	7.155×10^3	7.303×10^3	$\mathbf{7.329 \times 10^3}$	7.097×10^3
G(500, .25)	1.673×10^4	1.697×10^4	1.712×10^4	$\mathbf{1.714 \times 10^4}$	1.652×10^4
G(500, .5)	3.272×10^4	3.275×10^4	3.313×10^4	$\mathbf{3.314 \times 10^4}$	3.311×10^4
G(500, .75)	4.820×10^4	$\mathbf{4.852 \times 10^4}$	4.847×10^4	4.849×10^4	4.813×10^4

Synaptic plasticity and spectral analysis: Oja's Rule

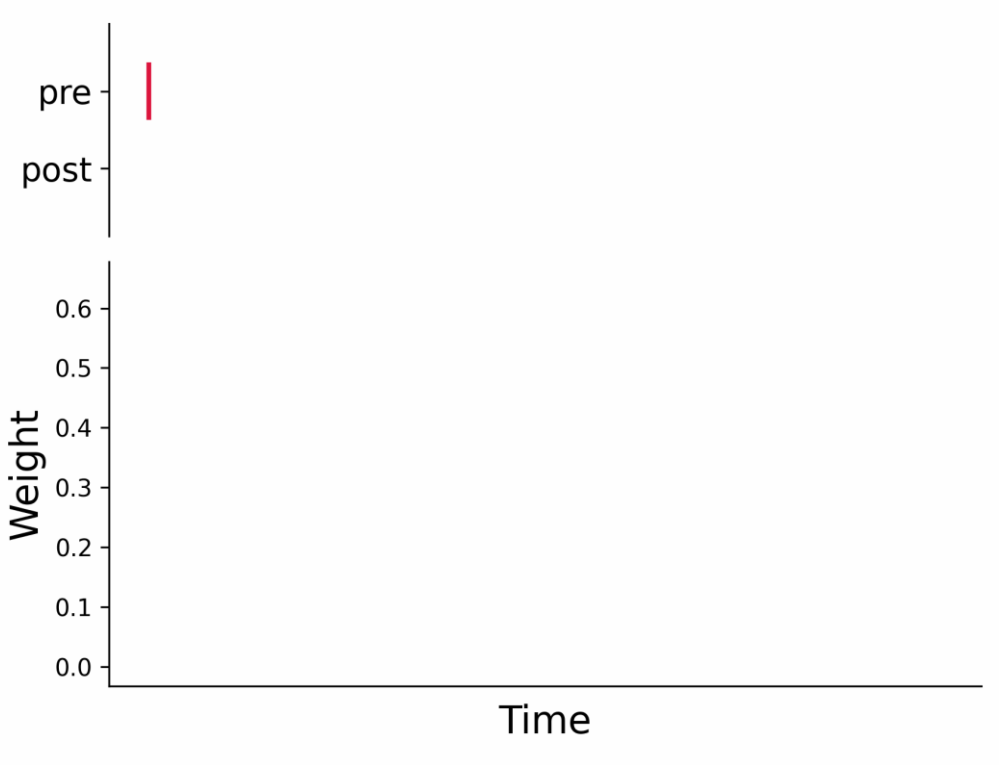
- Hebbian principle: neurons that fire together, wire together



Probabilistic Neural Theory and Algorithms

$$\Delta \boldsymbol{w} = y\boldsymbol{x}$$

- Oja's rule: stabilized Hebbian plasticity



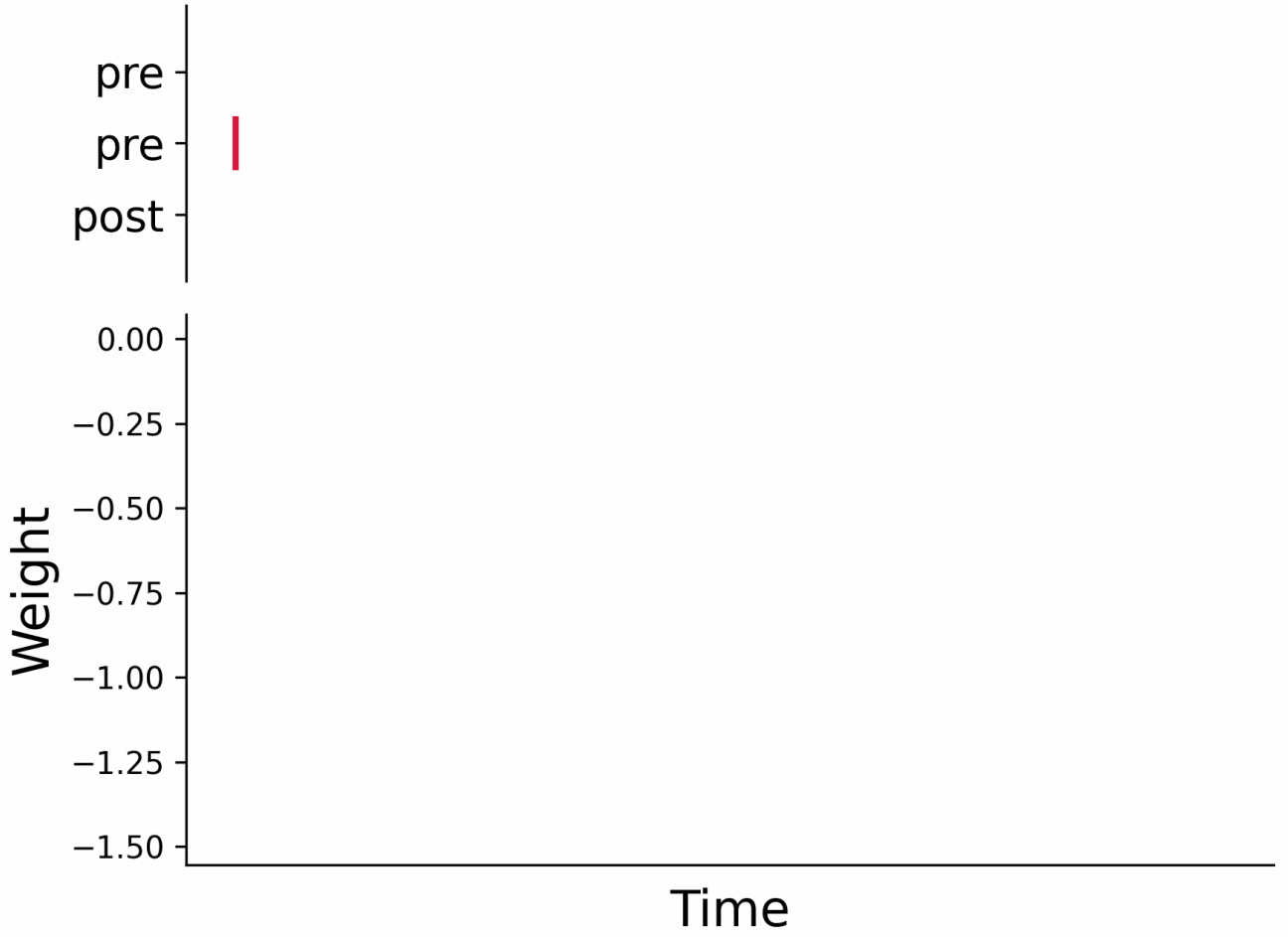
$$\Delta \boldsymbol{w} = y(\boldsymbol{x} - \boldsymbol{w}y)$$

Synaptic plasticity and spectral analysis: Oja's rule

- Oja's rule approximates principal component / maximum eigenvector

- Oja's *antihebbian* rule approximates minimum eigenvector:

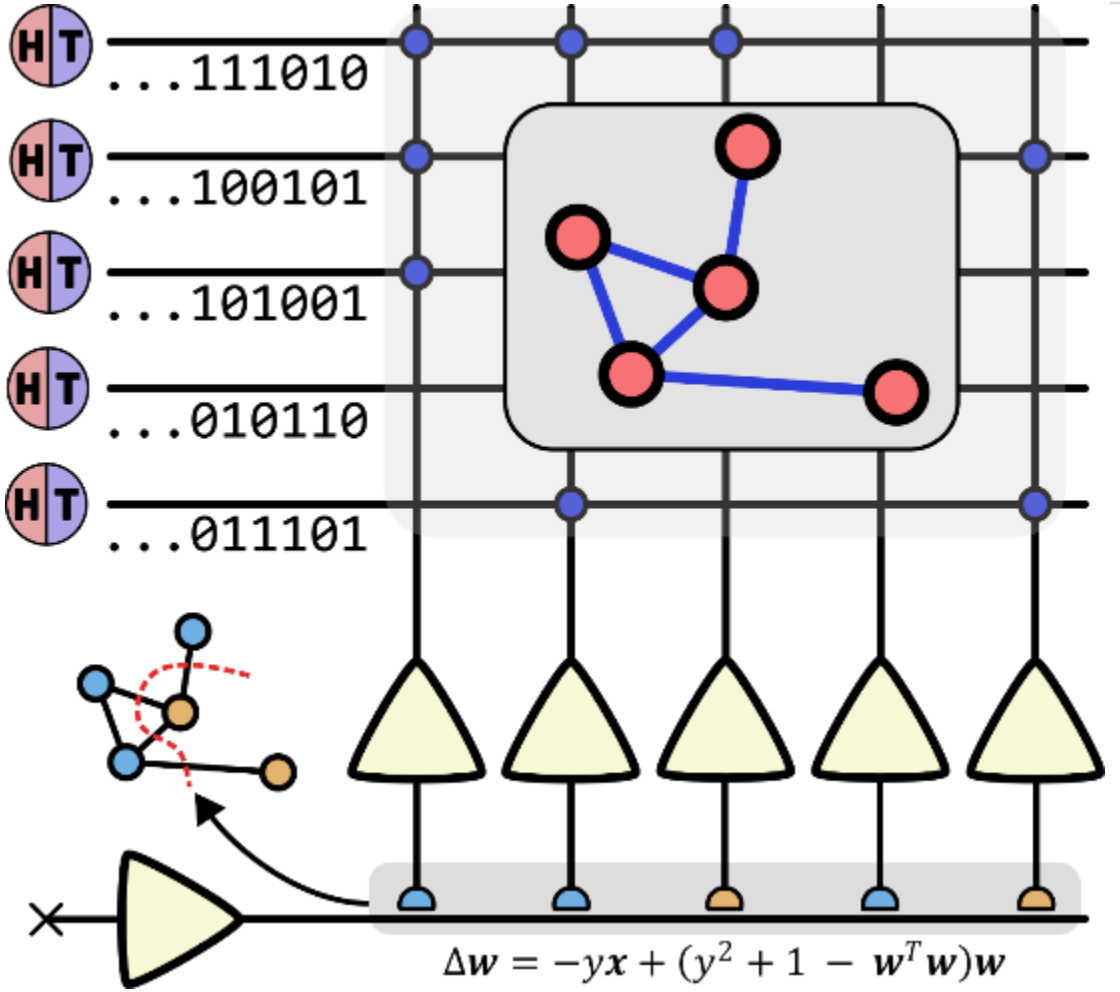
$$\Delta \boldsymbol{w} = -y\boldsymbol{x} + (y^2 + 1 - \boldsymbol{w}^T\boldsymbol{w})\mathbf{w}$$



Probabilistic Neural Theory and Algorithms

LIF-Trevisan circuit

- Correlation element generates correlated activity from random devices
- "Output" neuron computes minimum eigenvector via Oja's antihebbian rule

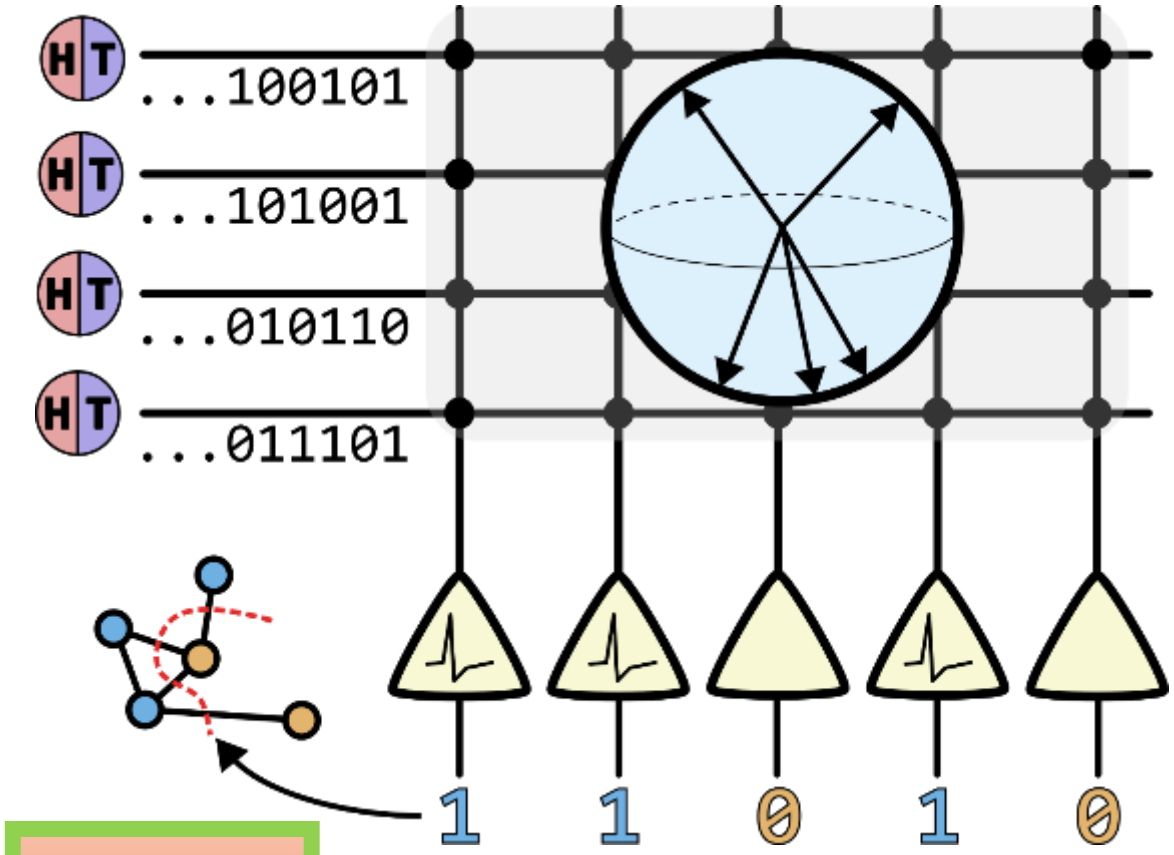


Probabilistic Neural Theory and Algorithms

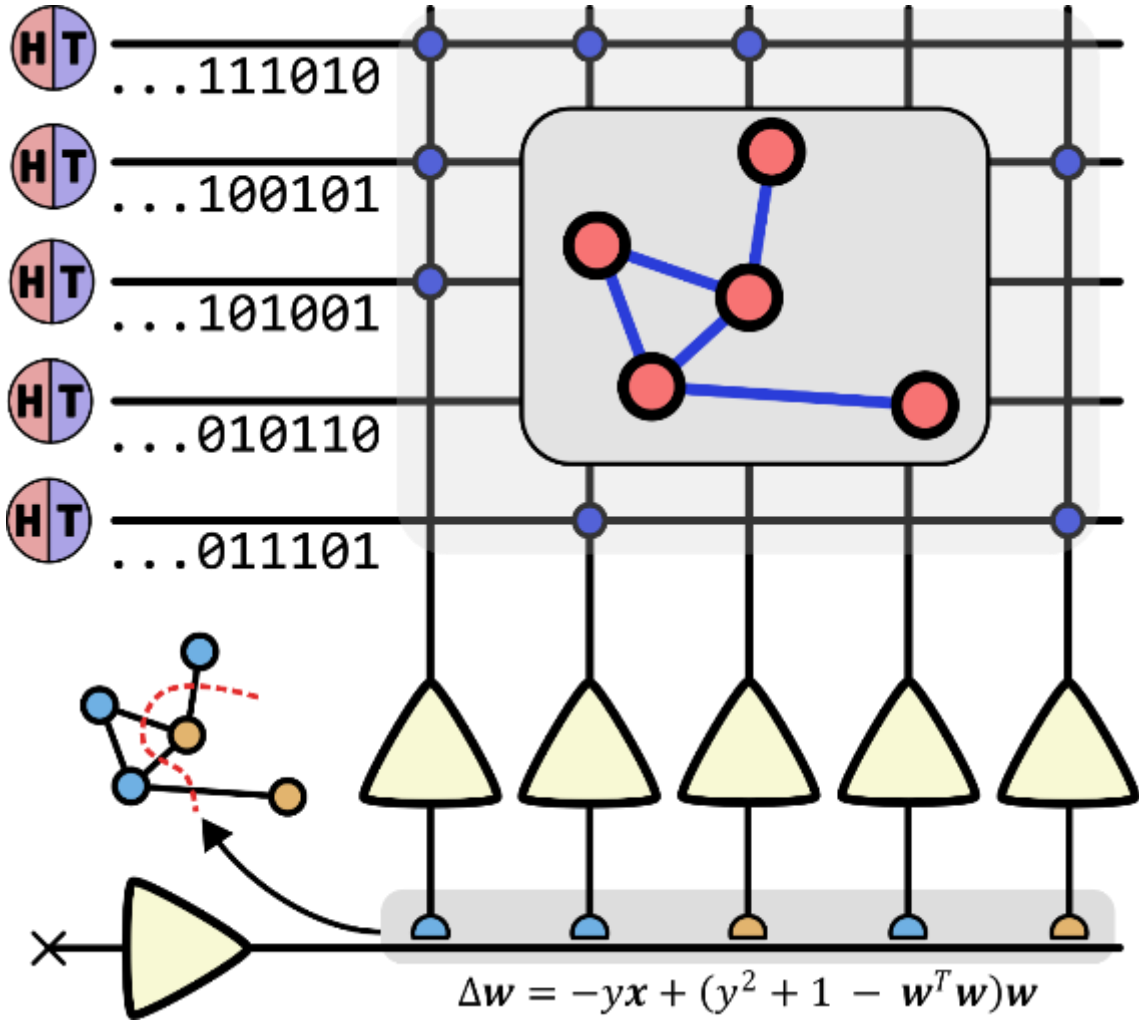
Neuromorphic maxcut circuits

COINFLIPS

LIF-GW



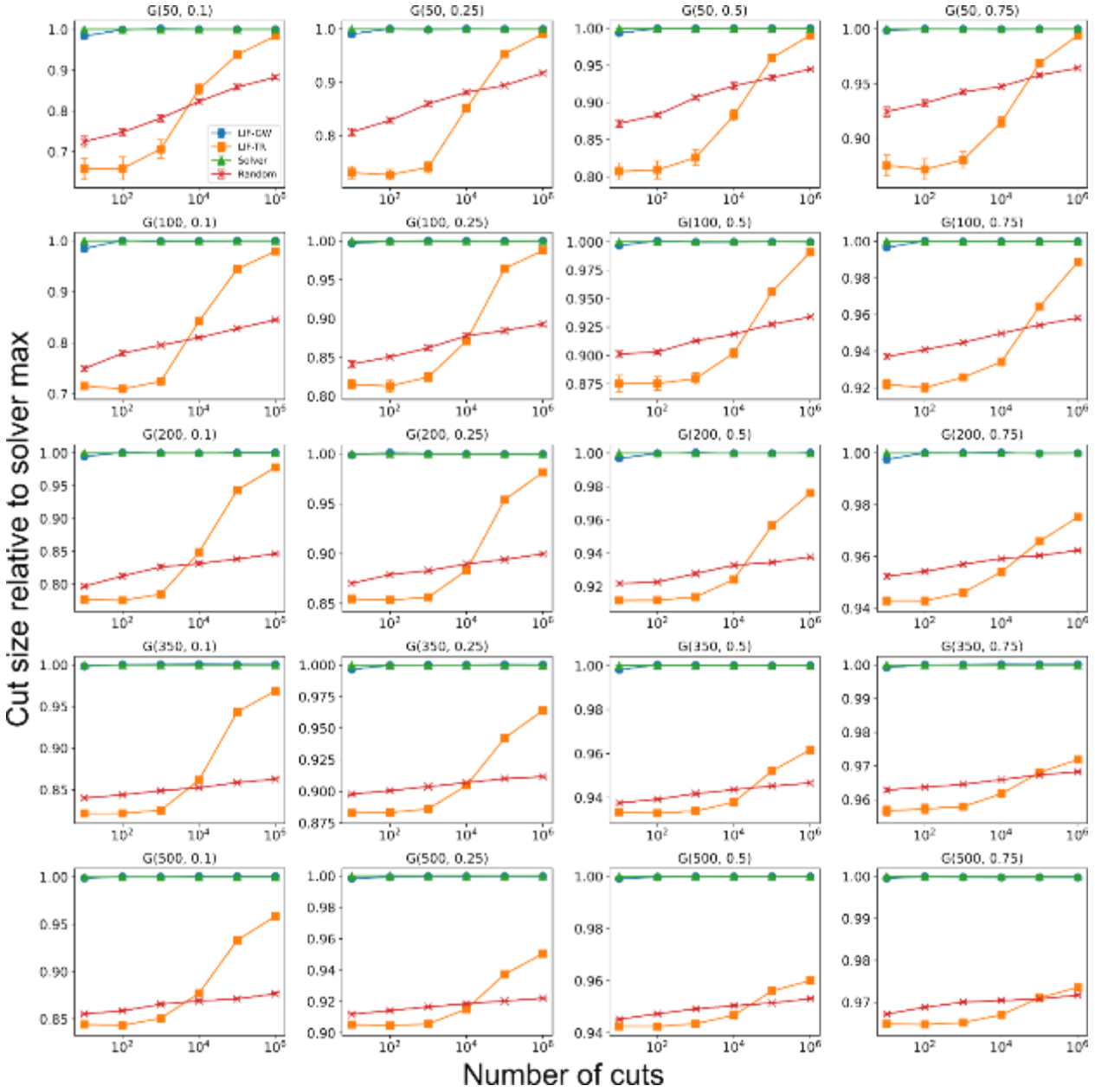
LIF-TR



Probabilistic Neural Theory and Algorithms

Maxcut Results

Erdos-Renyi random graphs



LIF-GW

LIF-TR

Solver

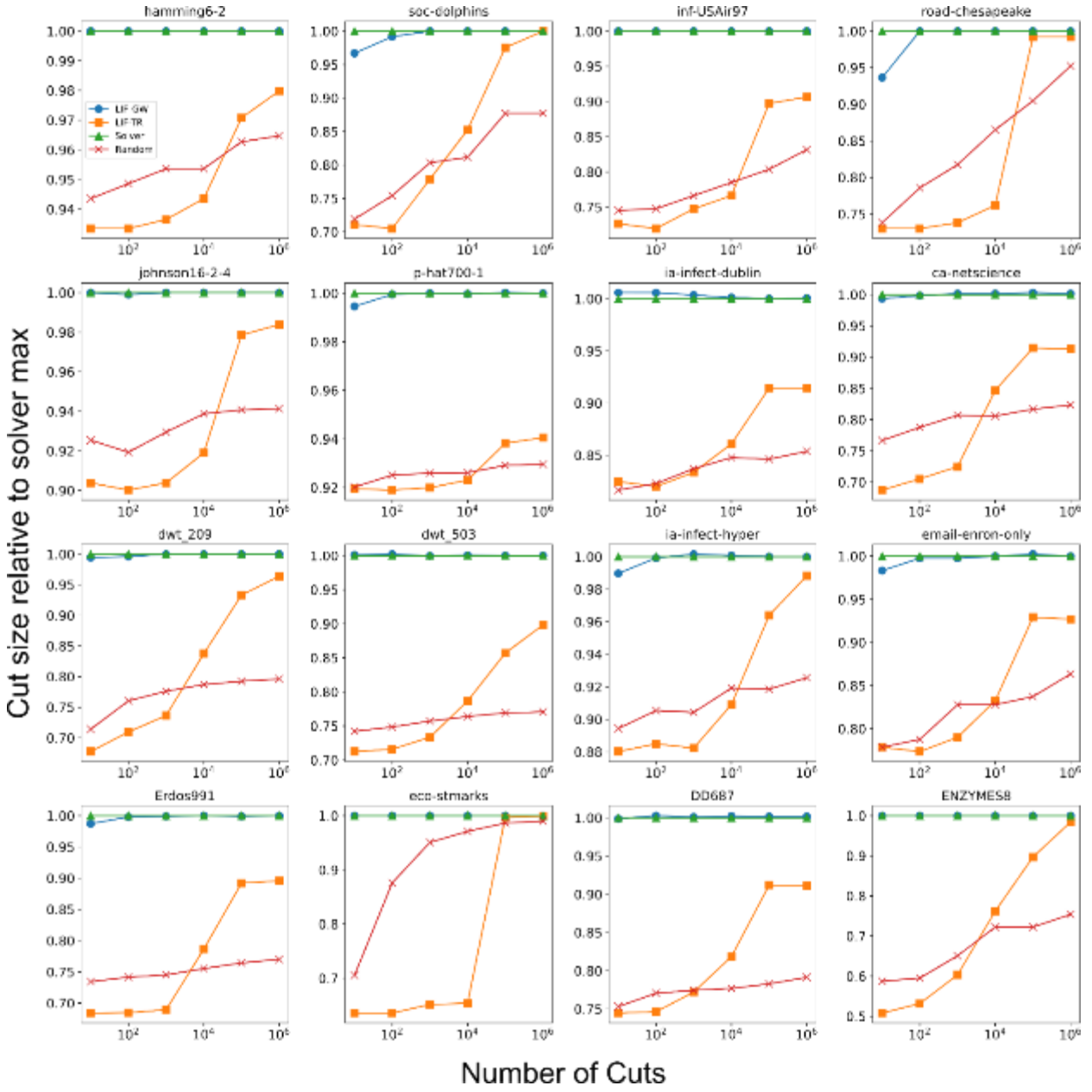
Random

Probabilistic Neural Theory and Algorithms

COINFLIPS

Maxcut Results

Empirical graphs (NRVIS)



LIF-GW

LIF-TR

Solver

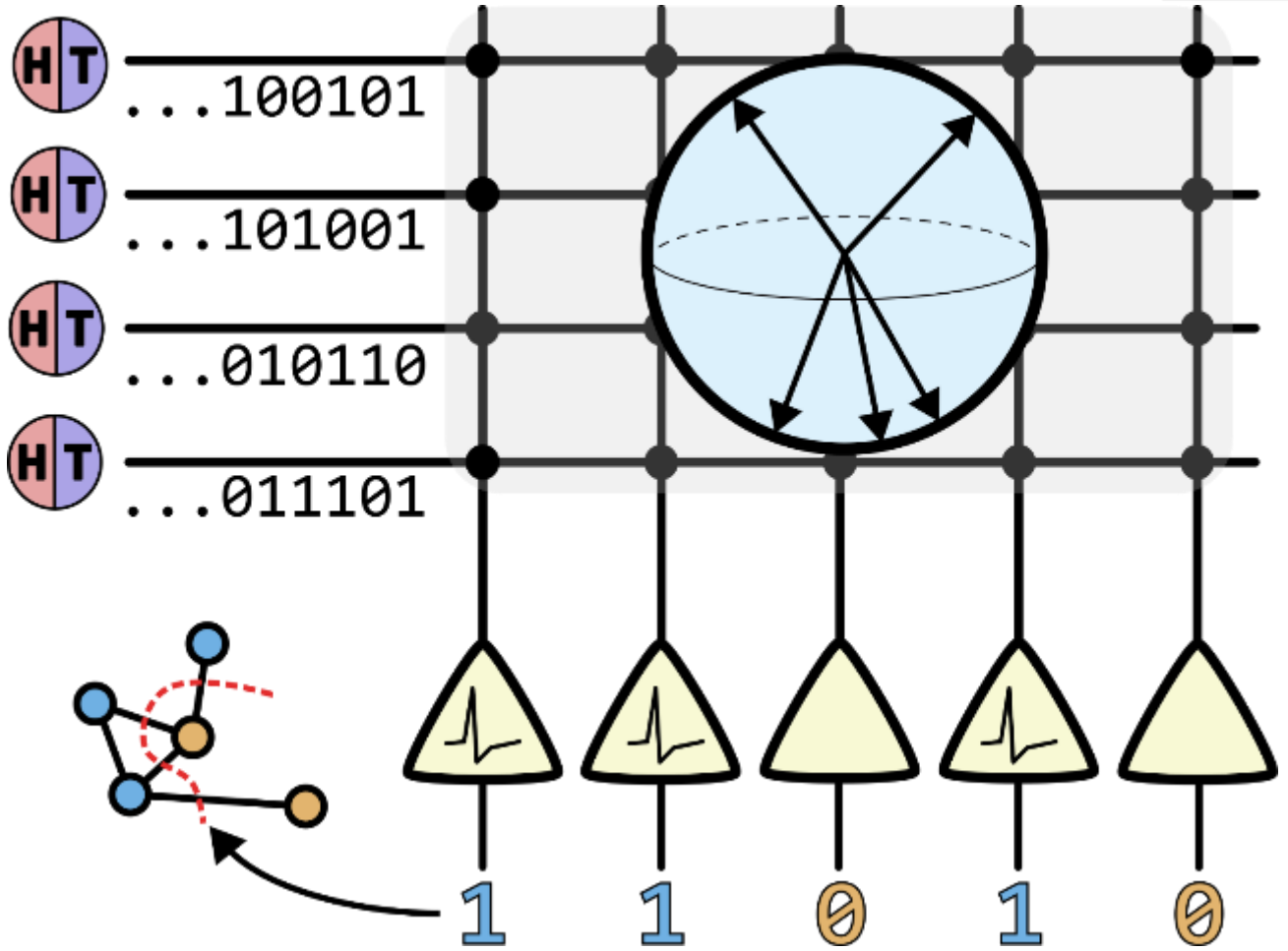
Random

Probabilistic Neural Theory and Algorithms

COINFLIPS

Loihi generated graph cuts

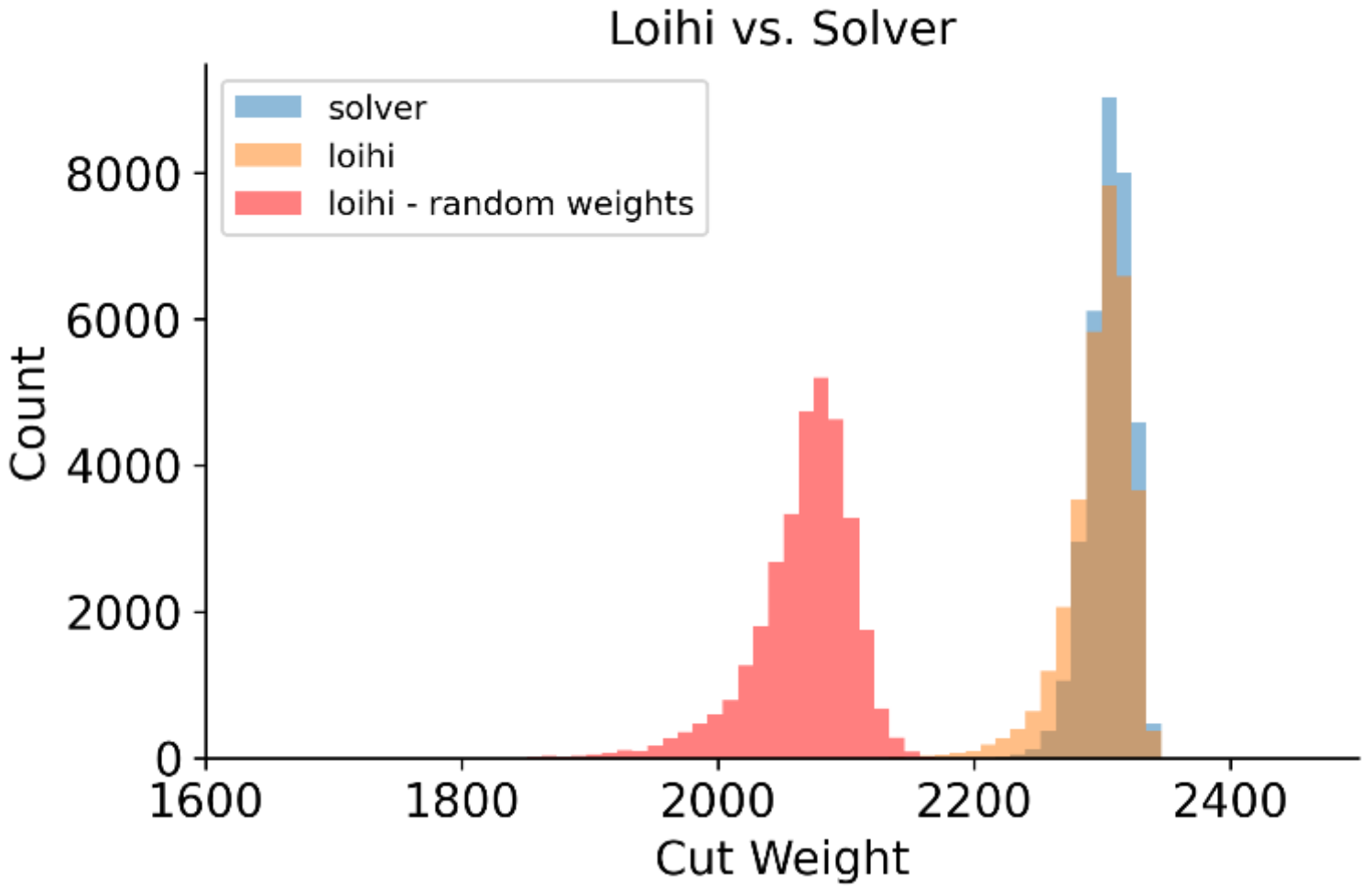
- Erdos-Renyi graph
  - 128 vertices
  - $p_{edge} = 0.5$
- GW vectors scaled to ± 255
- 2^{15} timesteps
- V_m time constant: 4 timesteps



Probabilistic Neural Theory and Algorithms

Loihi generated graph cut distribution

- Erdos-Renyi graph
  - 128 vertices
  - $p_{edge} = 0.5$
- GW vectors scaled to ± 255
- 2^{15} timesteps
- V_m time constant: 4 timesteps

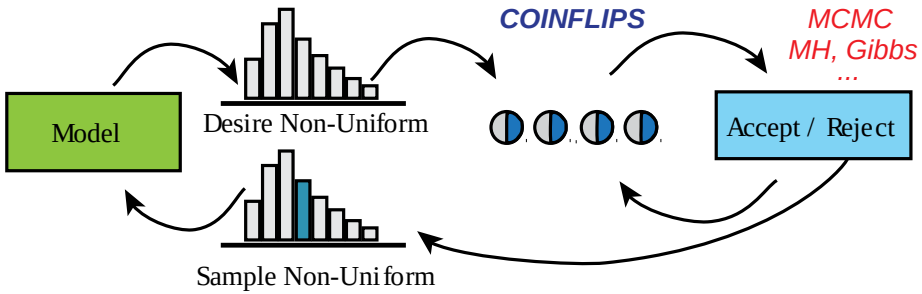


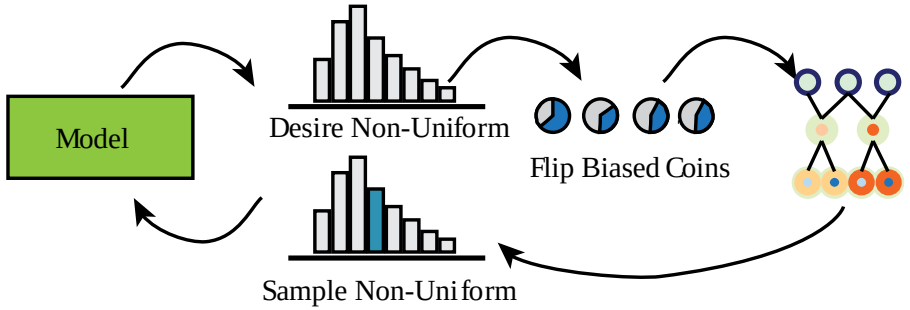
Probabilistic Neural Theory and Algorithms

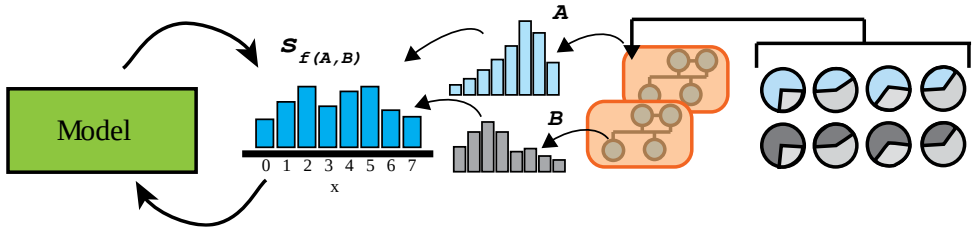
Evaluate opportunity of a probabilistic computing paradigm

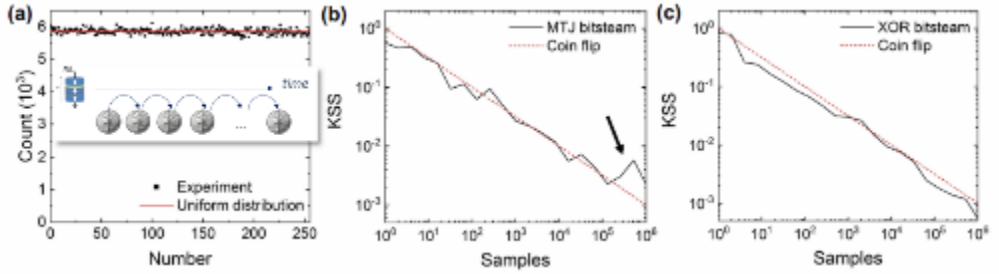
COINFLIPS

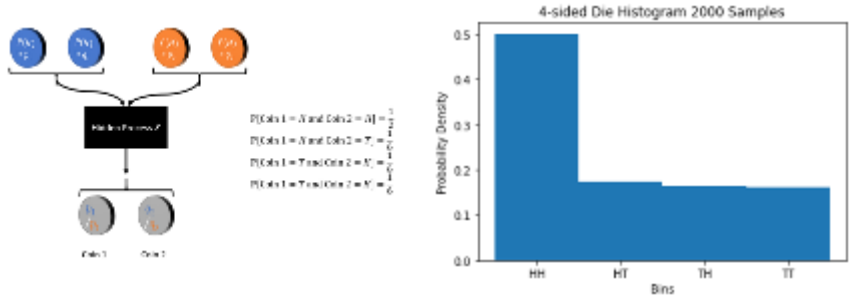
The COINFLIPS future may not be far away

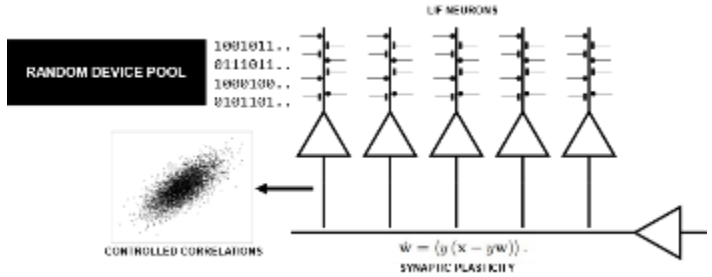




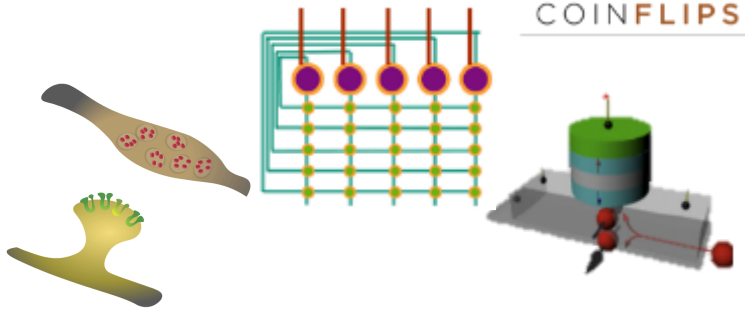




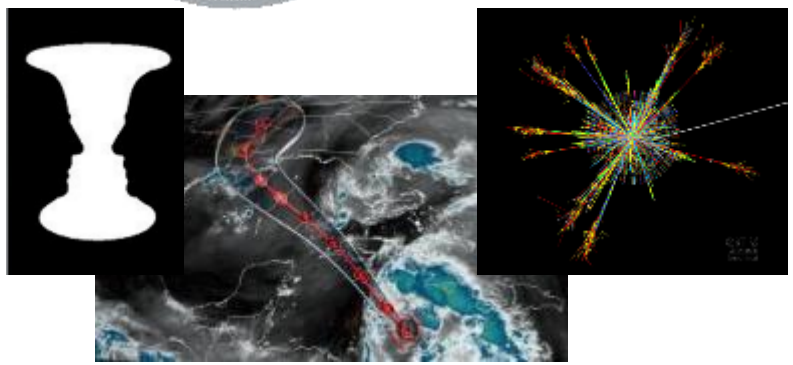




Summary: Probabilistic computing is perhaps an ideal target for exploring potential for future neuromorphic applications



- Brain is probabilistic exciting ways that have yet to be explored
- Stochastic devices
  + neuromorphic parallelism
  = broad application impact
  - *Both Mod-Sim and AI stand to benefit*
- Opportunity to consider important aspects of computing up front
  - *Address issues such as I/O, programmability, and theory from the onset, as opposed to after-the-fact*



Thank You!

- Neuromorphic testbed and Fugu
  - DOE Advanced Simulation and Computing (ASC)
  - Craig Vineyard, Suma Cardwell, Ryan Dellana, Fred Rothganger, William Severa, Srideep Musuvathy
- Neural PDE work
  - Sandia LDRD office
  - **Darby Smith**, William Severa, Rich Lehoucq, Ojas Parekh, Aaron Hill
- COINFLIPS / MAXCUT:
  - DOE Office of Science (BES, ASCR), Co-design in Microelectronics
  - Shashank Misra, Conrad James, Darby Smith, Suma Cardwell, **Brad Theilman**, Ojas Parekh, Yipu Wang, Chris Allemang, William Severa, Prasanna Date, Andy Kent, Laura Reim, Les Bland, Bernd Surrow, Jean Anne Incorvia, Jaesuk Kwon, Sam Liu, Katie Schuman, Karan Patel
- GNATs
  - DOE Office of Science (ASCR), CRCNS program
  - Katie Schuman, Seung-Hwan Lim, Felix Wang, **Brad Theilman**, Fred Rothganger, Shruti Kulkarni, Anika Tabassum

U.S. DEPARTMENT OF ENERGY | Office of Science

jbaimon@sandia.gov