



A Probabilistic Future for Neuromorphic Computing

Brad Aimone

Center for Computing Research

Sandia National Laboratories





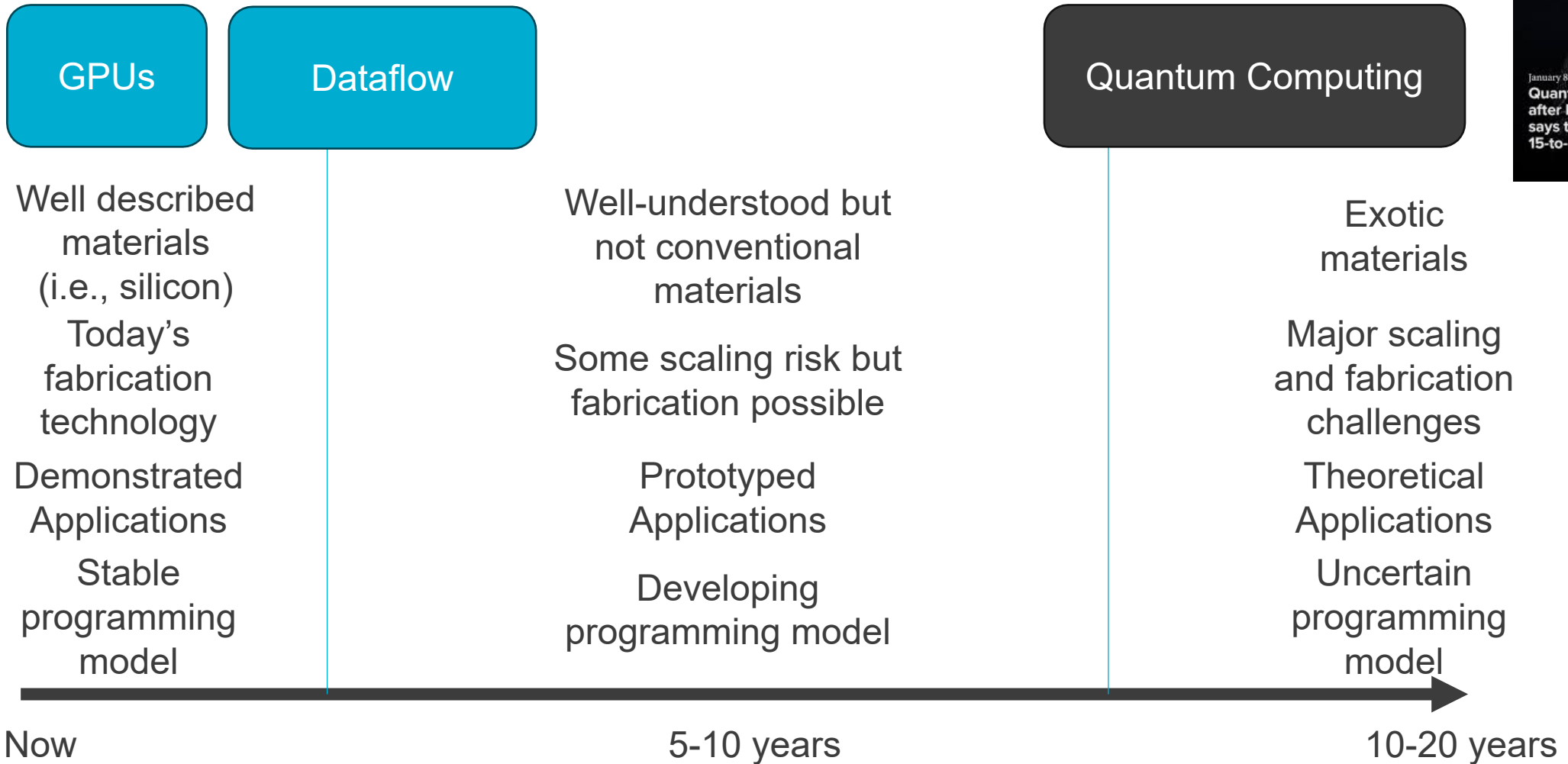
There is not a single roadmap for neuromorphic computing

- ❑ Active research across the technology stack
 - ❑ New post-CMOS materials (memristors, ECRAM, MTJs, quantum materials, ...)
 - ❑ Non-digital devices (analog, stochastic, optical, ...)
 - ❑ Bio-inspired circuits (reconfigurable, dendrites, learning, ...)
 - ❑ Neuromorphic architectures (spiking, event-driven sensors, ANN accelerators, ...)
 - ❑ Software paradigms (compilers, intermediate representations, ...)
 - ❑ Neuromorphic algorithms

*When is it ready for prime time?
What is needed to get there?*

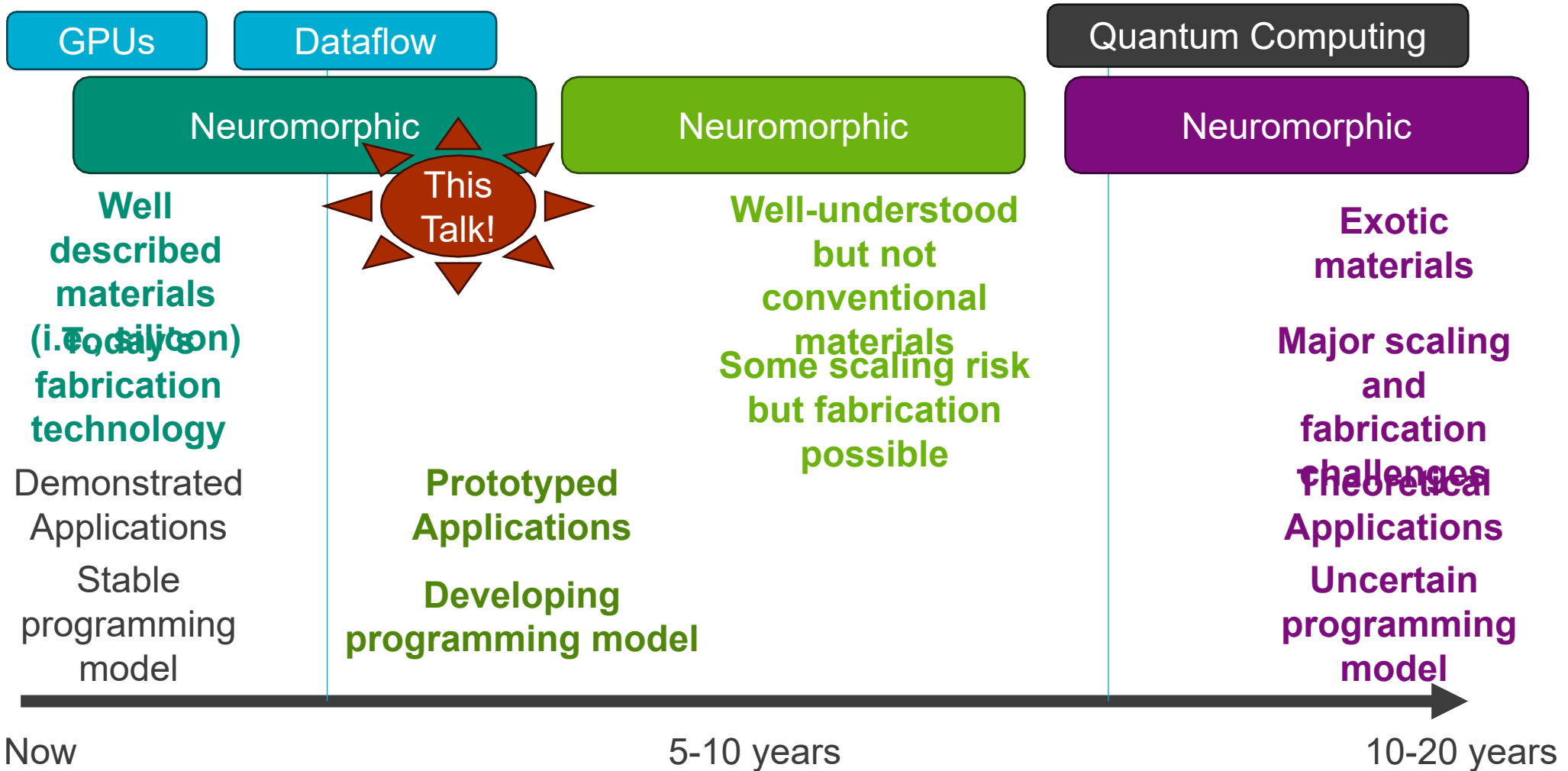


When will neuromorphic computing be a reality?





Neuromorphic computing has promise at different time scales

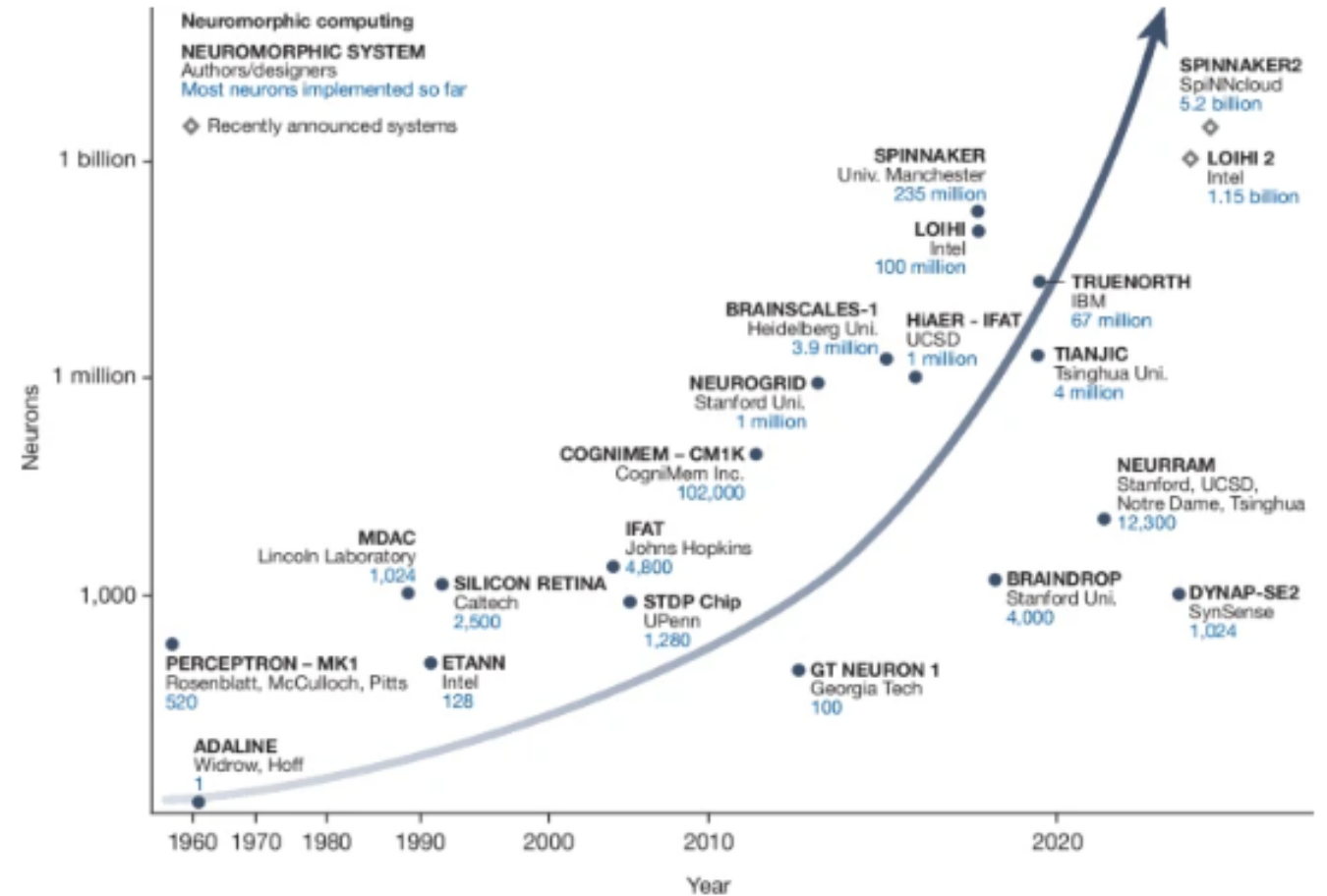




Today's digital neuromorphic systems are approaching brain-scale

- ❑ Systems like Intel's Loihi 2 and SpiNNaker2's SpiNNaker 2 can surpass 1 billion neurons
- ❑ Individual chips are ~1 million neurons and ~1 Watt
- ❑ Fully CMOS (little fabrication risk)
- ❑ Digital or Digital + Analog hybrid
- ❑ Future devices and novel materials can amplify potential impact

Fig. 1: Progression of neuromorphic computing systems.

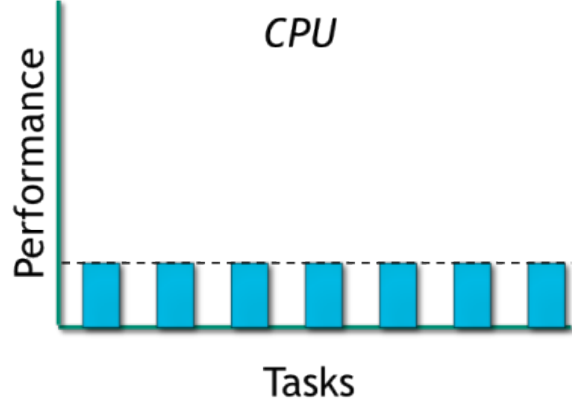


Kudithipudi et al., Nature 2025

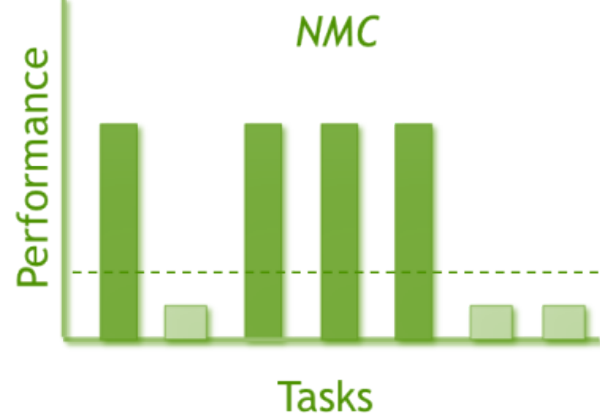
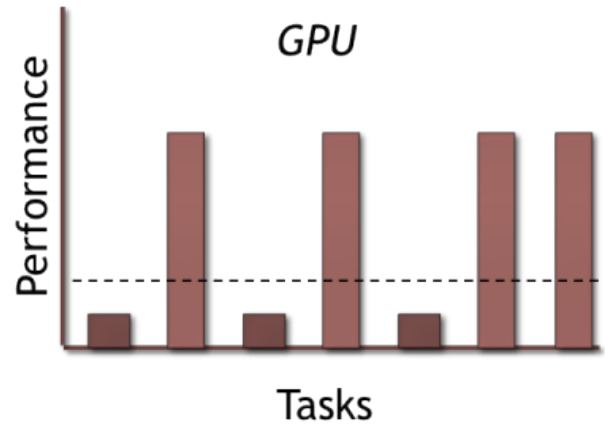


Neuromorphic is likely similar to GPUs in degree of specialization

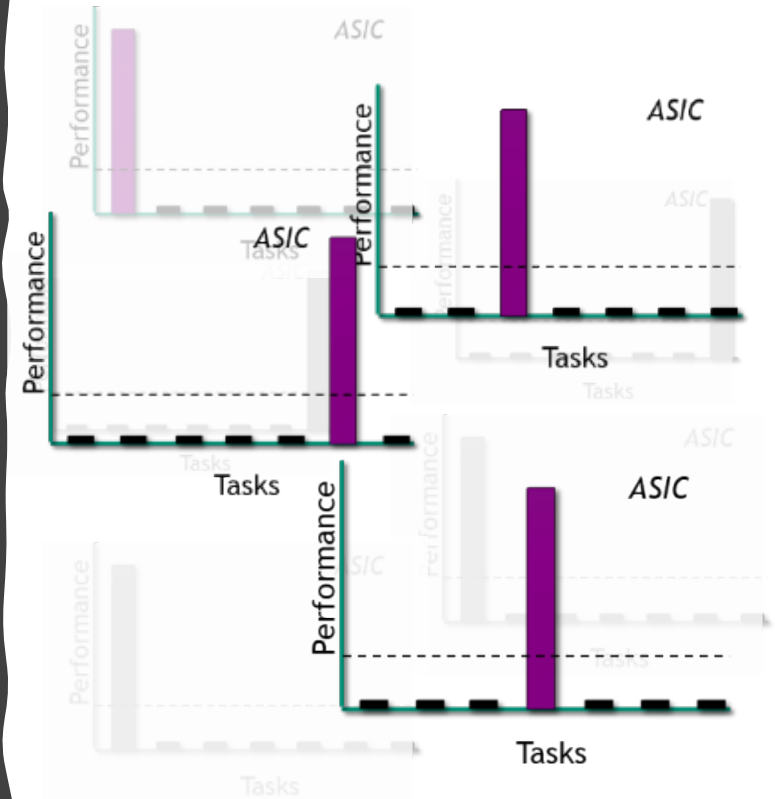
Truly General Purpose



Specialized General Purpose

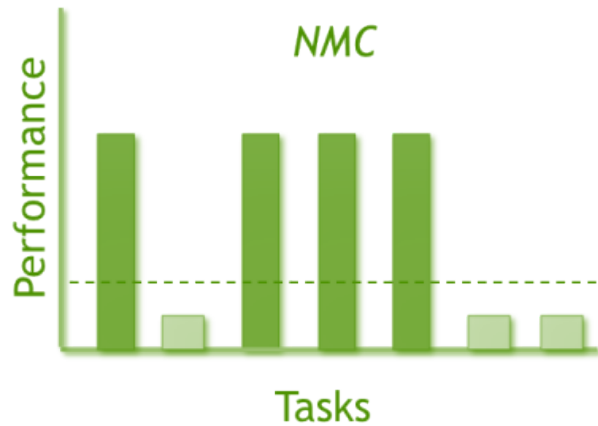
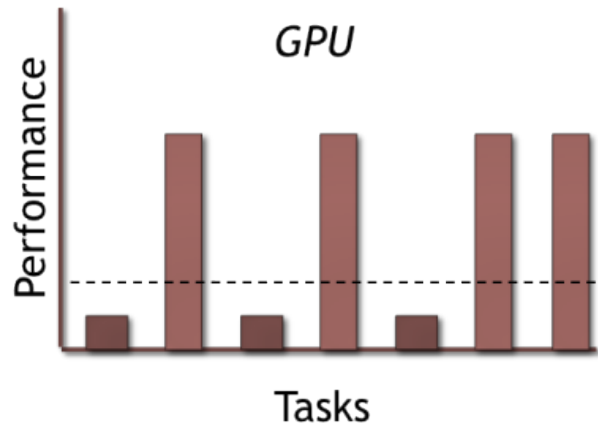


Application Specific





Claim: the major challenge to neuromorphic computing today is the algorithm impact



Identifying neuromorphic advantages today will

- ❑ Communicate the fundamental value proposition of neuromorphic specific
- ❑ Compare the scope of neuromorphic impact to alternatives (GPUs, accelerators, etc)
- ❑ Clarify what aspects of today's neuromorphic architectures need to be improved
- ❑ Justify cost of moving to new non-CMOS materials



So is there actually a neuromorphic advantage?

Neuromorphic advantage: an algorithm that surpasses conventional architectures on one performance axis (energy, time, accuracy) while demonstrating equivalent (or better) performance on all other axes

Machine Learning



Scientific Computing



Sensing and Autonomy





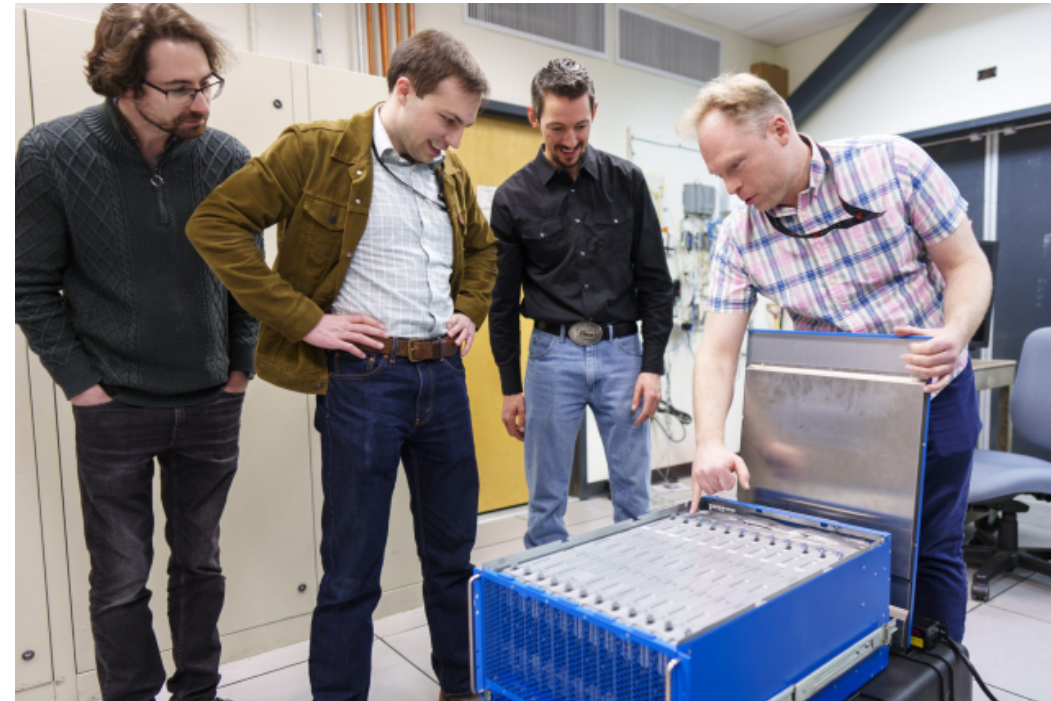
What is neuromorphic computing today?

In digital silicon (CMOS) technology

- Over 1 billion neuron system at Sandia
 - Roughly the number in a parrot or small primate brain
- Neurons are “simulated” in an efficient way
- Generally a leaky integrate-and-fire model

Analog systems

- Wide range of technologies, but far smaller
- Emulate the brain’s biophysics in different materials
- Wide range of neural dynamics emulated



Left to right: William Chapman, Brad Theilman,
Craig Vineyard, Mark Plagge



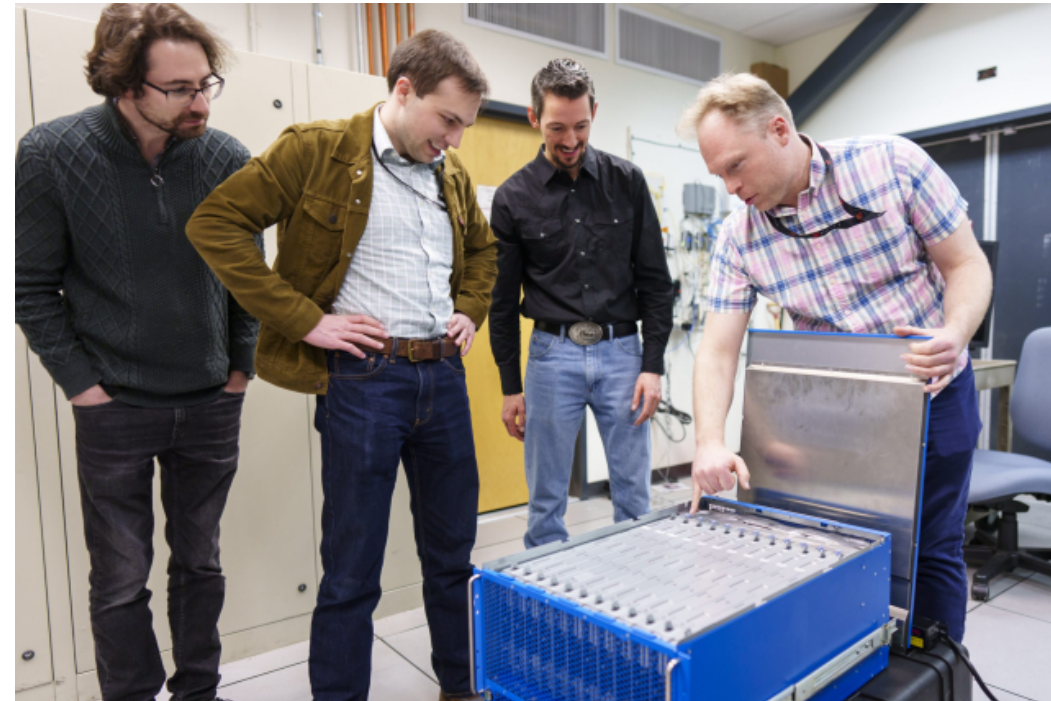
What is neuromorphic computing today?

In digital silicon (CMOS) technology

- Over 1 billion neuron system at Sandia
 - Roughly the number in a parrot or small primate brain
- Neurons are “simulated” in an efficient way
- Generally a leaky integrate-and-fire model

Analog systems

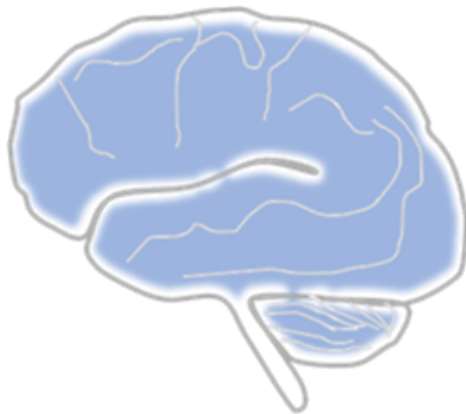
- Wide range of technologies, but far smaller
- Emulate the brain’s biophysics in different materials
- Wide range of neural dynamics emulated



Left to right: William Chapman, Brad Theilman,
Craig Vineyard, Mark Plagge

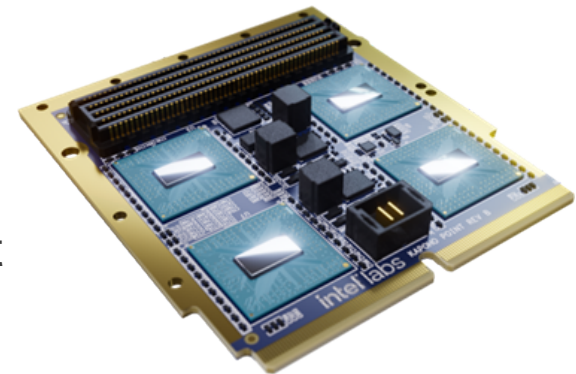


Rationale: Efficiency protects us from being trapped by two trivial conclusions



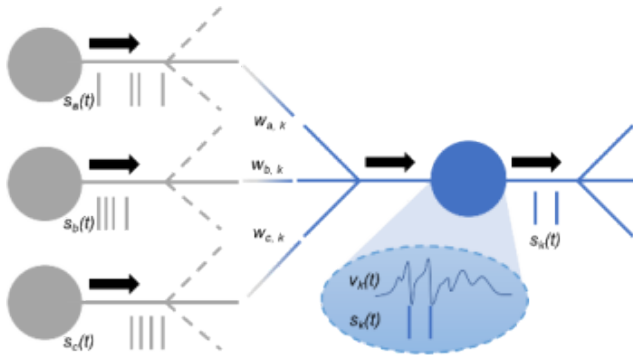
The brain (very) likely only computes what the brain is efficient at computing

Neuromorphic should have many of the same restrictions





Spiking neuromorphic today: Overview



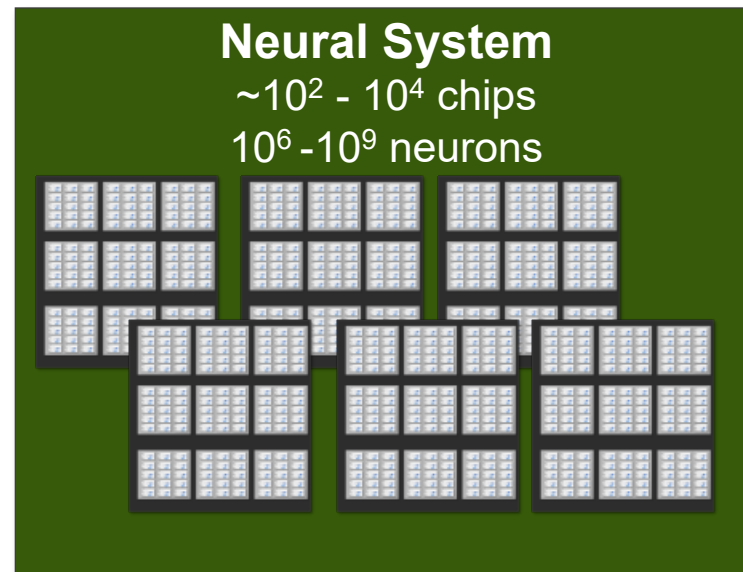
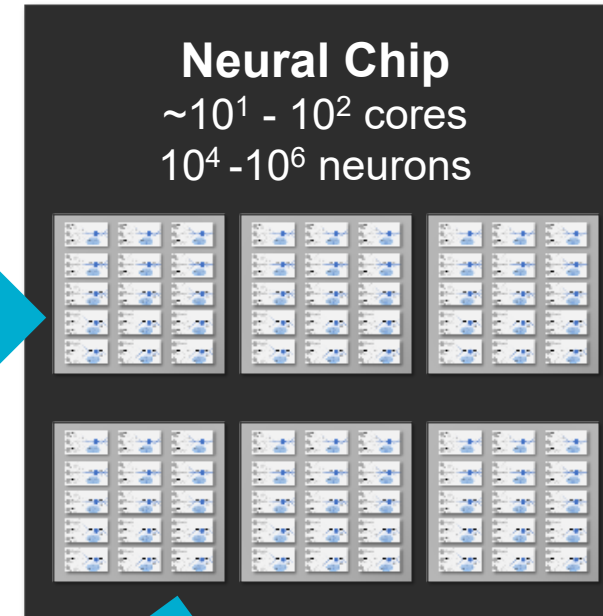
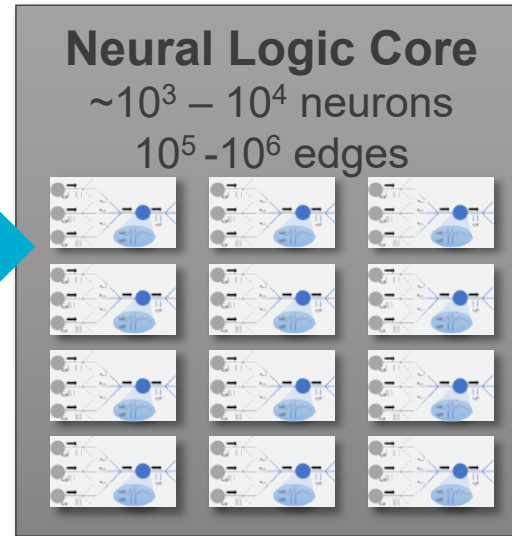
Computational Primitives:

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



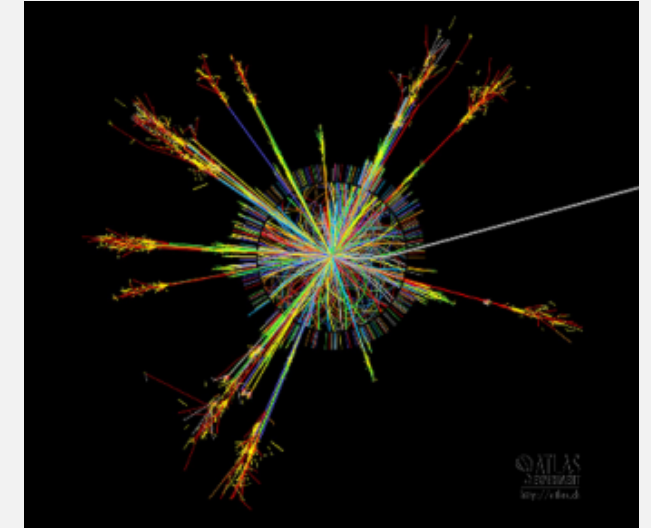
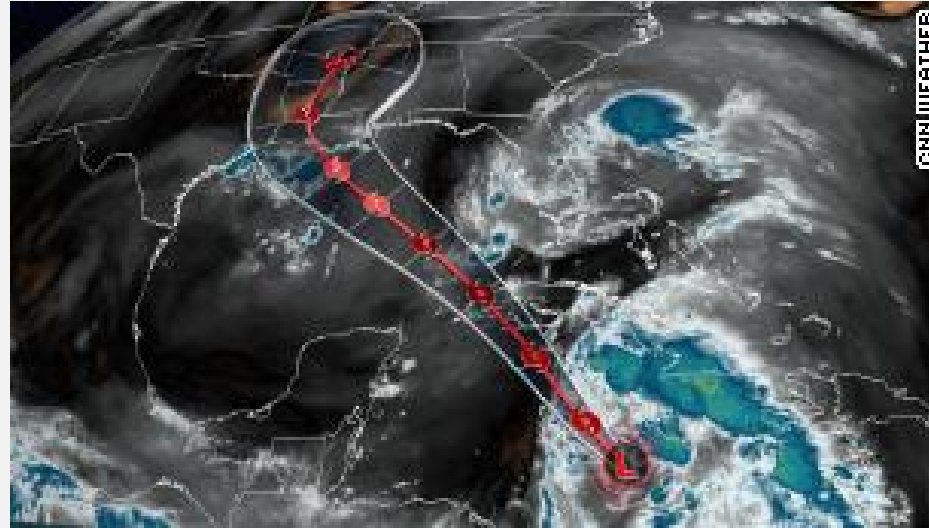


Computing applications face challenges in uncertainty



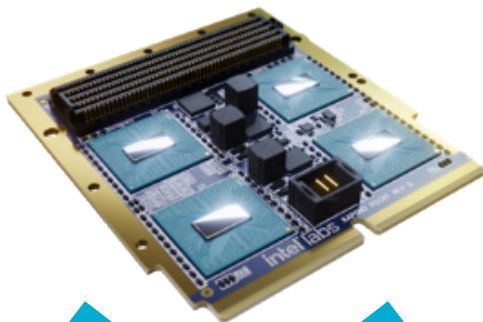
Artificial Intelligence

- Bayesian neural networks are appealing yet often computationally intractable

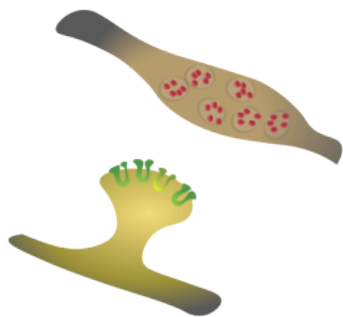


Modeling and Simulation

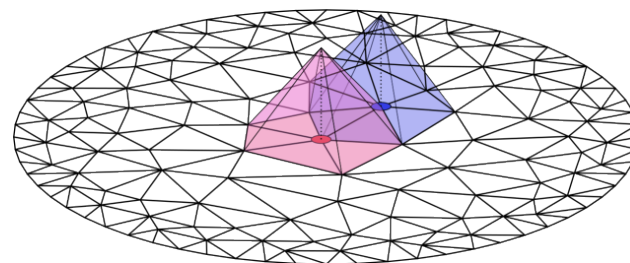
- Modeling uncertainties is critical in the use of even fully deterministic simulations
- Many applications are inherently stochastic in their physics and are best modeled using probabilistic methods



Problems that require
widespread sampling



Problems that are
naturally spatially
distributed





Diffusion

Diffusion can be modeled either as a deterministic PDE or a stochastic process

- For an initial distribution of particles, P_0 , what is distribution of particles at time t ?
- PDE solves the following equation

$$\frac{\partial C(x,t)}{\partial t} = D \frac{\partial^2 C(x,t)}{\partial x^2}$$

- Stochastic process implements many random walkers to statistically approximate a solution
 - Mean position of N walkers approaches expected mean of deterministic solution at rate of $1/\sqrt{N}$



“Naïve” way to run diffusion on GPUs

```
void rw_mix(curandState_t* states, int n, int *x, int *y)
```

```
{
```

```
    int i = blockIdx.x*blockDim.x+threadIdx.x;
```

```
    int rand_1;
```

```
    int rand_2;
```

```
    if (i < n){
```

```
        for (int t=0; t<100000; t++){
```

```
            rand_1 = curand(&states[i]) % 2;
```

```
            rand_2 = curand(&states[i]) % 2;
```

```
            if (rand_1 == 1){
```

```
                if(rand_2 == 1) x[i]++;
```

```
                else x[i]-=1;
```

```
            }
```

```
            else{
```

```
                if (rand_2 == 1) y[i]++;
```

```
                else y[i]-=1;
```

```
            }
```

```
        }
```

```
        y[i] = y[i]%21;
```

```
        x[i] = x[i]%21;
```

```
    }
```

```
}
```



For every particle “owned” by this GPU



... loop through 100,000 time steps



... draw two random numbers



50% chance to pick up or down...



50% chance to pick left or right...



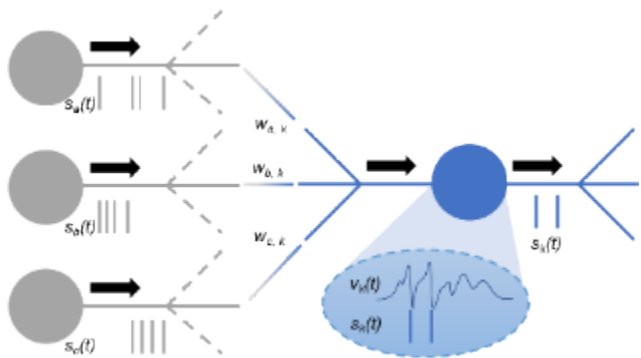
... mod back to torus

- ✓ Easy to do
- ✓ Embarassingly parallel over particles
- ✓ Mostly independent of size of state space

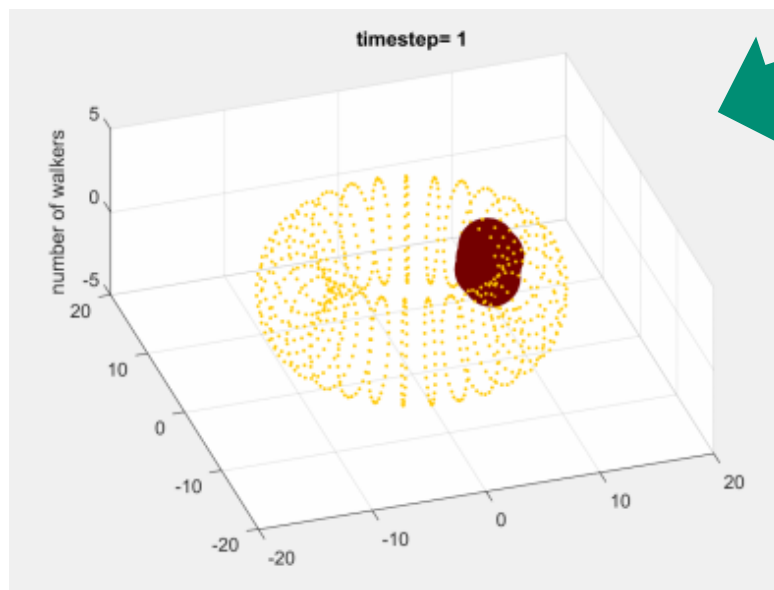
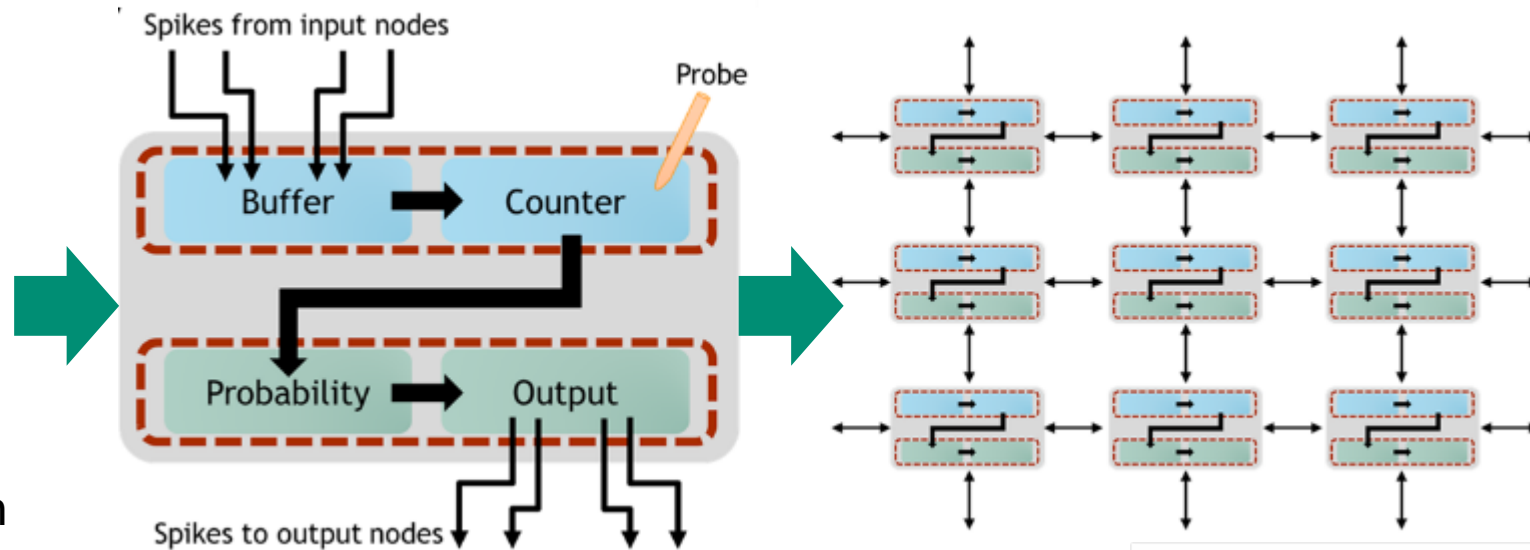
- ✗ Particle location in computer is independent from location in model
- ✗ High cost of inhomogeneity
- ✗ Lots of random number draws



Flip the approach! Use neurons to represent *state space* of Monte Carlo and use spikes to represent particles



Leaky integrate-and-fire neuron



nature electronics ARTICLES
<https://doi.org/10.1038/s41945-021-00765-7>

Neuromorphic scaling advantages for energy-efficient random walk computations

J. Darby Smith¹, Aaron J. Hill, Leah E. Reeder, Brian C. Franke, Richard B. Lehoucq, Ojas Parekh, William Severa and James B. Aumanez²

Neuromorphic computing, which aims to replicate the computational structure and architecture of the brain in synthetic hardware, has typically focused on artificial intelligence applications. What is less explored is whether such brain-inspired hardware can provide value beyond cognitive tasks. Here we show that the high degree of parallelism and configurability of spiking neuromorphic architectures makes them well suited to implement random walks via discrete-time Markov chains. These random walks are useful in Monte Carlo methods, which represent a fundamental computational tool for solving a wide range of numerical computing tasks. Using IBM's TrueNorth and Intel's Loihi neuromorphic computing platforms, we show that our neuromorphic computing algorithm for generating random walk approximations of diffusion offers advantages in energy-efficient computation compared with conventional approaches. We also show that our neuromorphic computing algorithm can be extended to more sophisticated jump-diffusion processes that are useful in a range of applications, including financial economics, particle physics and machine learning.

Despite the increasing ability to develop large-scale neural hardware, the theoretical value of neuromorphic hardware remains unclear—unlike quantum computing that offers clear fundamental advantages at scale¹. Nevertheless, there are several architectural features of most nervous systems that could yield advantages including the high degree of connectivity between neurons, the collocation of processing and memory, and the use of action potentials (referred to as spikes) to communicate^{2,3}. Algorithm research for spiking neuromorphic hardware has primarily focused on its suitability for deep learning and other emerging artificial intelligence (AI) algorithms^{4–7}. Such applications are straightforward, given the alignment of neural architectures with neural networks, and it can be expected that the value of neuromorphic computing will grow as AI algorithms derive further inspiration from the brain⁸. However, the impact of neuromorphic computing beyond cognitive applications is less certain.

Quantum computing has shown how emerging hardware can have an impact beyond its original inspiration: it was conceived as a means for efficient chemistry simulations^{9,10}, but is now recognized as useful in a much broader range of applications^{11–13}. Unlike quantum computing, which faces technical challenges in scaling up¹⁴,

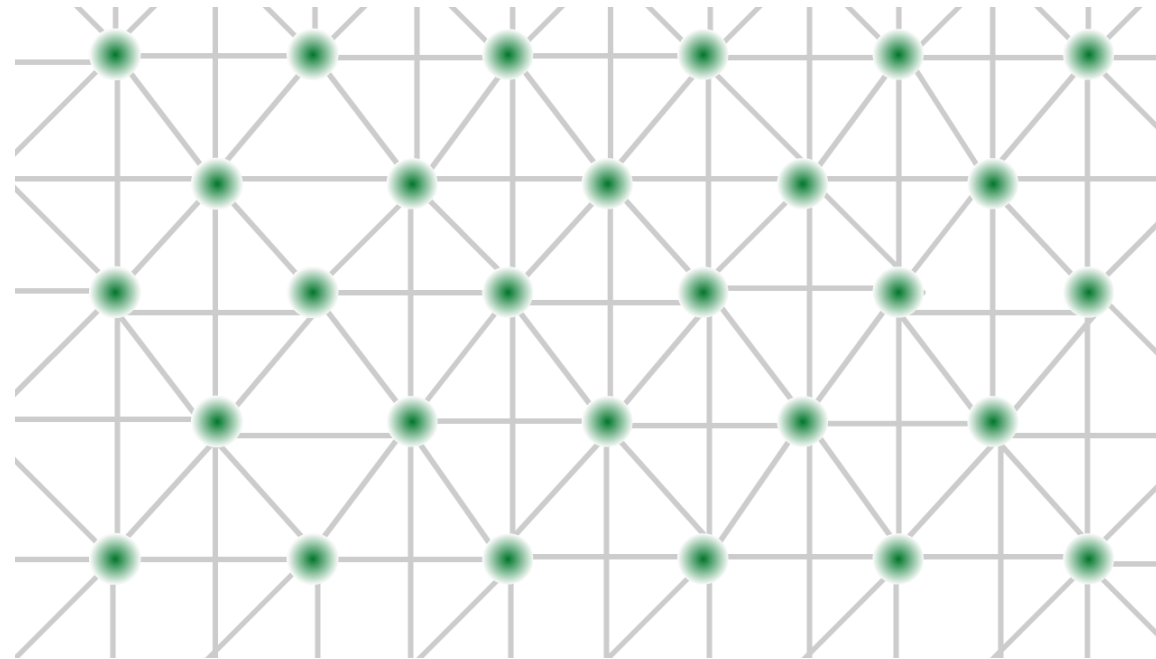
neuro scaling compared with the von Neumann architecture and still requiring less total energy to perform the same computation.

Observing a neuromorphic advantage for non-cognitive applications should not be taken as a given since the specialization of computer architectures to improve performance on a subset of tasks will likely result in degraded performance in other tasks¹⁵. Therefore, observing a neuromorphic advantage on non-cognitive applications would demonstrate that neuromorphic computing can have a broader impact than previously assumed and provide a concrete framework by which to develop the technology. Although a definitive neuromorphic advantage (as defined here) has not yet been demonstrated for non-cognitive applications, there are three categories of such computing tasks that appear well suited for neuromorphic computing: linear algebra, in which the high fan-in of neurons can be used to realize known theoretical advantages of threshold gate (TG) logic¹⁶; graph statistical tasks that can leverage the configurability and parallelization of neural circuits^{17–19}; and sampling steady state distributions for a wide range of potential applications using stochastic neural circuits^{20–22}.

In this Article, we show that large-scale neuromorphic hardware can offer a neuromorphic advantage on a fundamental



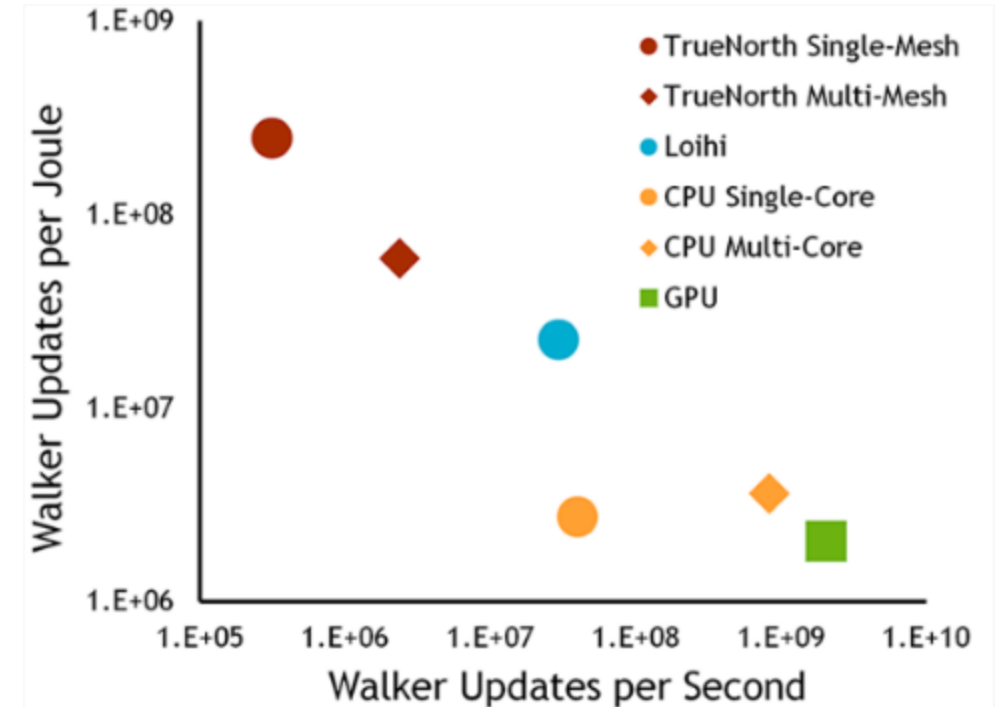
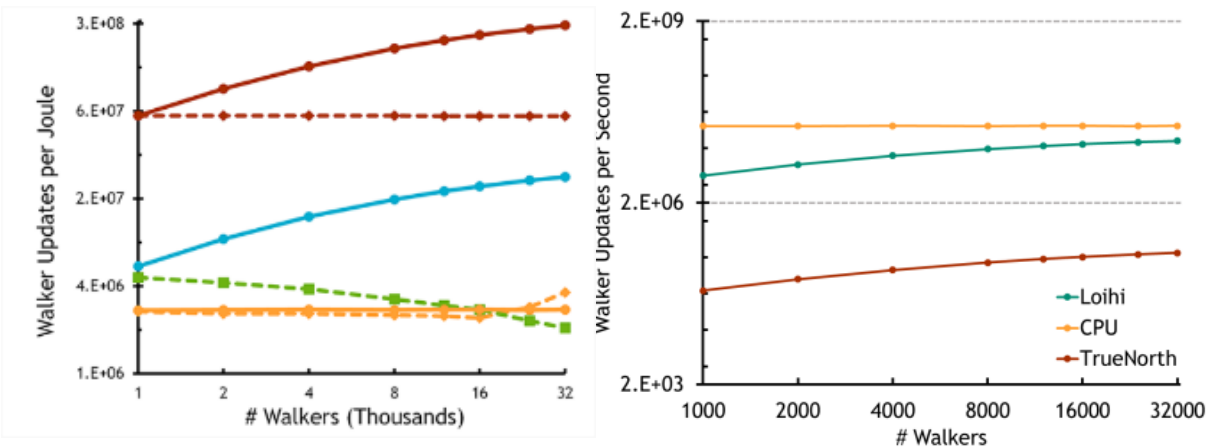
Neuromorphic computing advantage appears to be when an algorithm can split the task across computational graph with sparse communication





We can identify a neuromorphic advantage for simulating random walks

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).





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

Class of Partial Integro-Differential Equations:

$$\begin{aligned} \frac{\partial}{\partial t} u(t, \mathbf{x}) = & \frac{1}{2} \sum_{i,j} (\mathbf{a}\mathbf{a}^\top)_{i,j}(t, \mathbf{x}) \frac{\partial^2}{\partial x_i \partial x_j} u(t, \mathbf{x}) + \sum_i b_i(t, \mathbf{x}) \frac{\partial}{\partial x_i} u(t, \mathbf{x}) \\ & + \lambda(t, \mathbf{x}) u(t, \mathbf{x}) + \int \mathbf{h}(t, \mathbf{x}, q) - u(t, \mathbf{x}) \phi_Q(q; t, \mathbf{x}) dq \\ & + c(t, \mathbf{x})u(t, \mathbf{x}) + f(t, \mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^d, t \in [0, \infty). \end{aligned}$$

Stochastic Process:

$$d\mathbf{X}(t) = \mathbf{b}(t, \mathbf{X}(t)) dt + \mathbf{a}(t, \mathbf{X}(t)) d\mathbf{W}(t) + \int \mathbf{h}(t, \mathbf{X}(t), q) dP(t; Q, \mathbf{X}(t)).$$

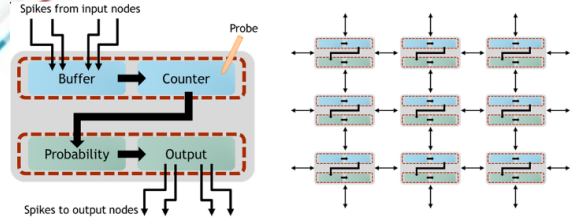
NMC Hardware Simulates This Stochastic Process



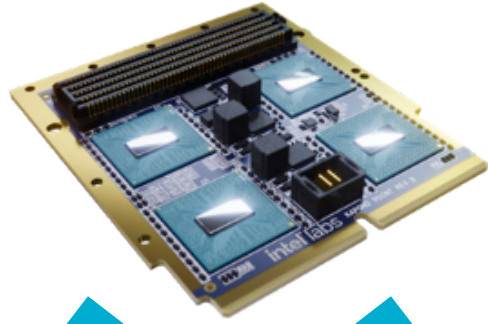
Monte Carlo Approximates This Expectation

Solution to initial value problem $(u(0, \mathbf{x}) = g(\mathbf{x}))$

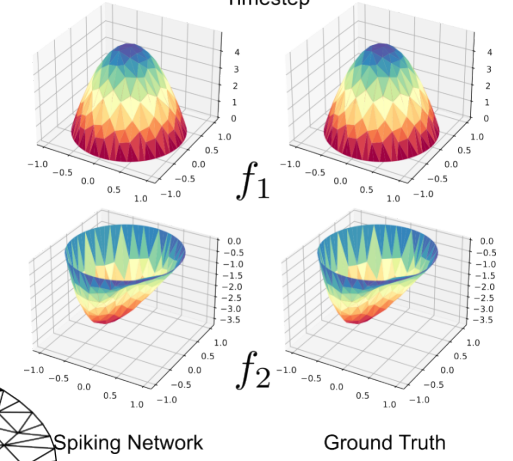
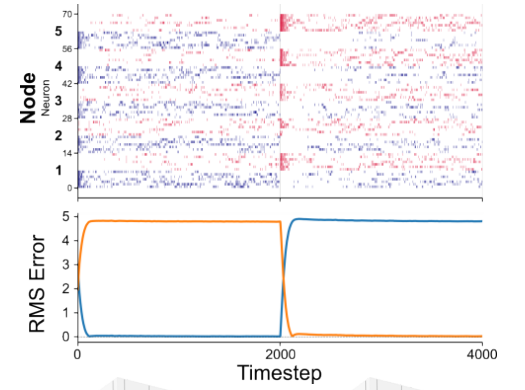
$$u(t, \mathbf{x}) = \mathbb{E} \left[g(\mathbf{X}(t)) \exp\left(-\int_0^t c(s, \mathbf{X}(s)) ds\right) + \int_0^t f(s, \mathbf{X}(s)) \exp\left(-\int_0^s c(\ell, \mathbf{X}(\ell)) d\ell\right) ds \mid \mathbf{X}(0) = \mathbf{x} \right].$$



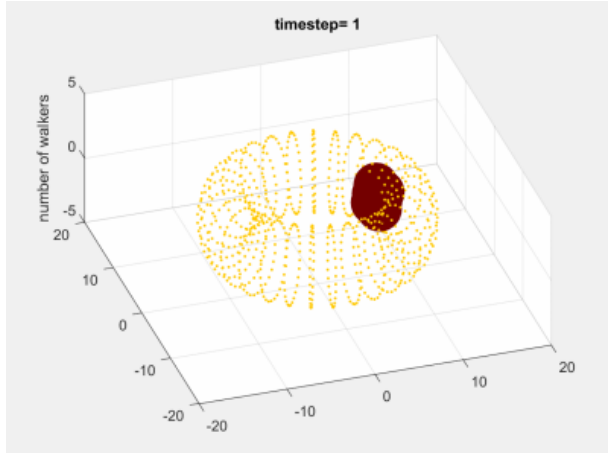
Neuromorphic hardware can *efficiently* solve stochastic Monte Carlo simulations



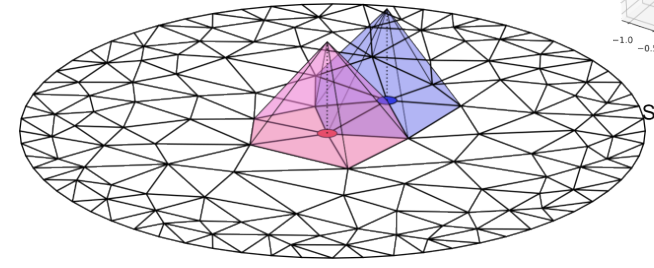
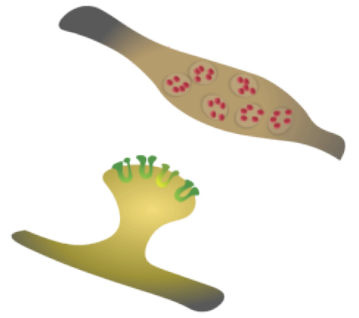
Neuromorphic hardware can *efficiently* solve finite element method simulations



Theilman et al., in review



Smith et al., Nature Electronics, 2022

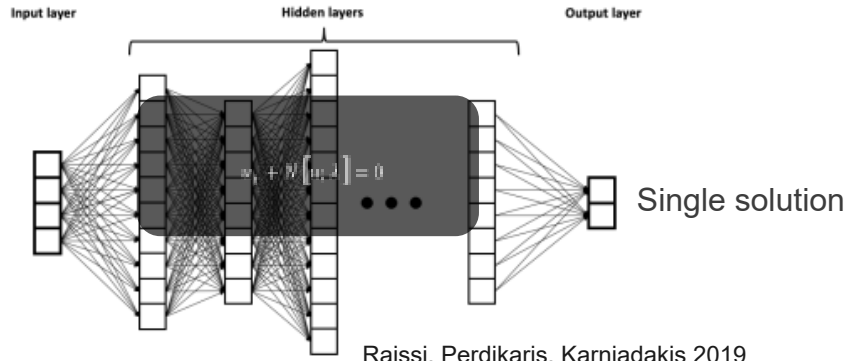




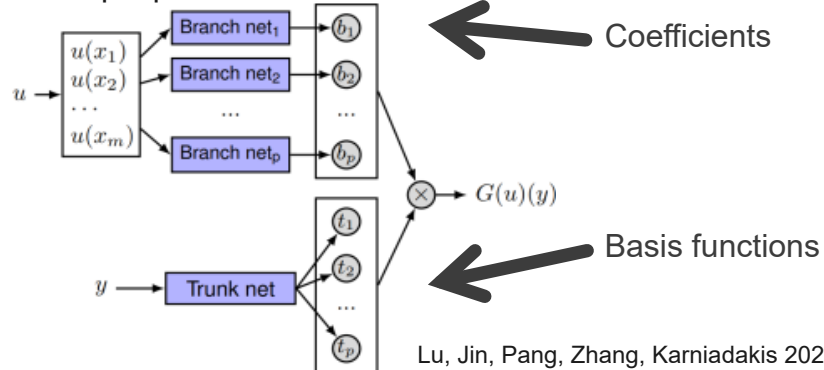
Solving partial differential equations is a major challenge to emerging computing platforms

Scientific Machine Learning

Physics-Informed Neural Networks

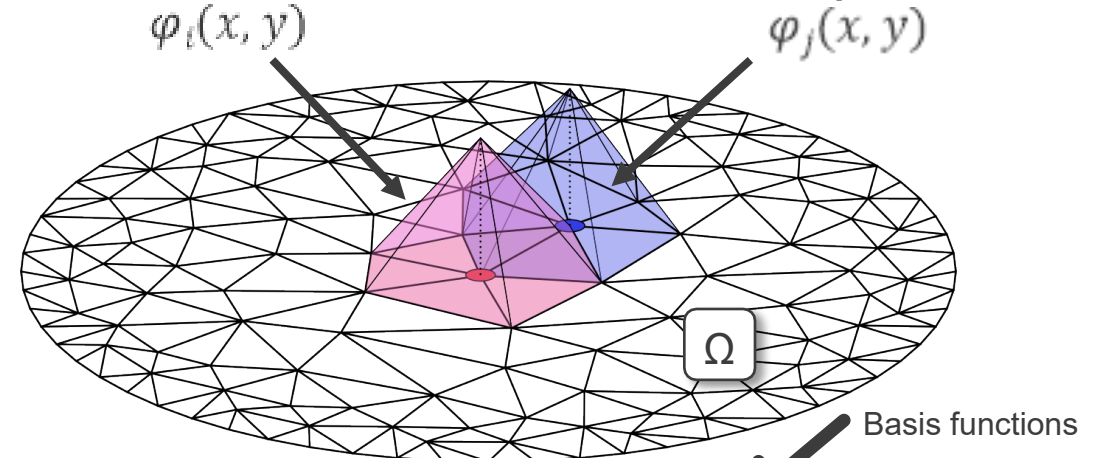


Deep Operator Networks



$$\nabla^2 u = f$$

Finite Element Methods – Gold Standard but expensive



$$u(x, y) \approx \sum_i u_i \phi_i(x, y)$$

Coefficients

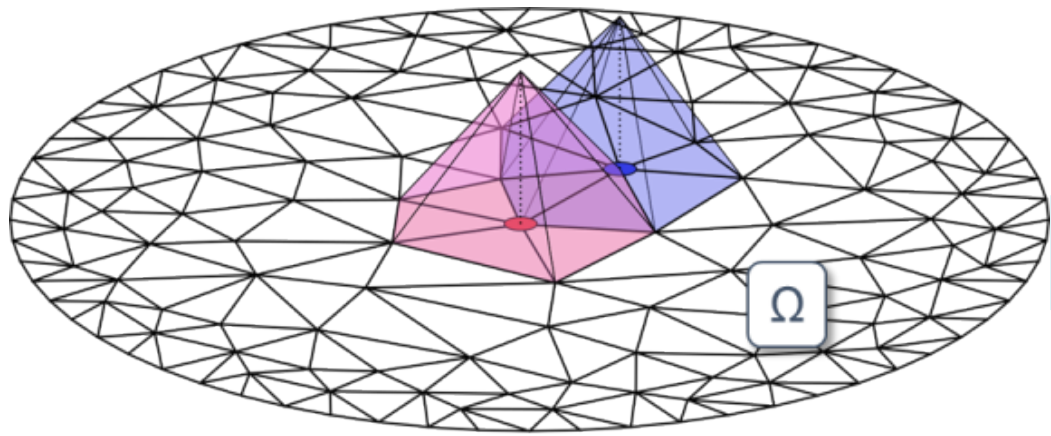
Sparse, linear system

$$A\vec{u} = \vec{b}$$

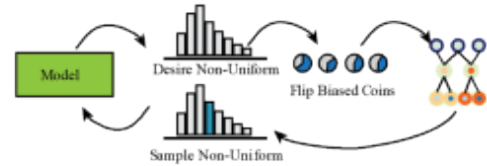
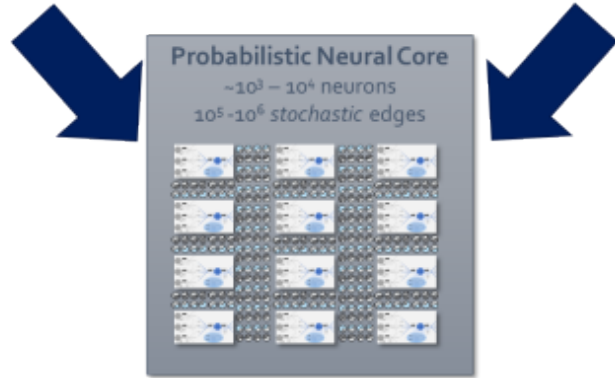
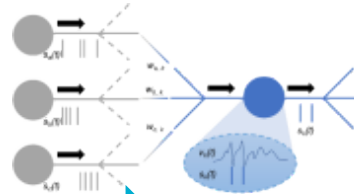
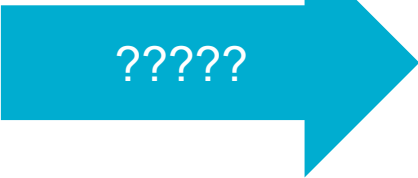
NEUROMORPHIC FINITE ELEMENT METHODS?



Can we tackle FEM with probabilistic neural hardware?

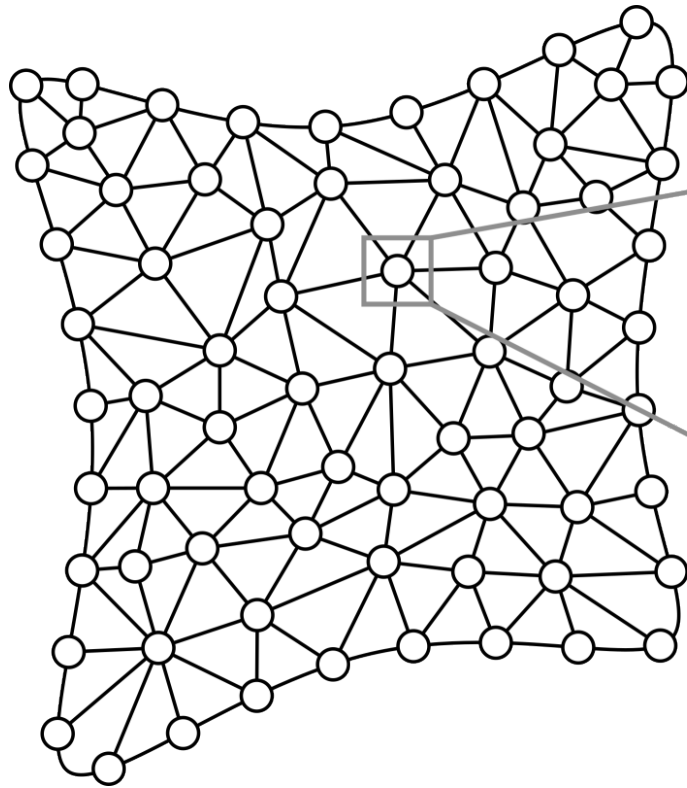


$$u(x, y) \approx \sum_i u_i \varphi_i(x, y)$$

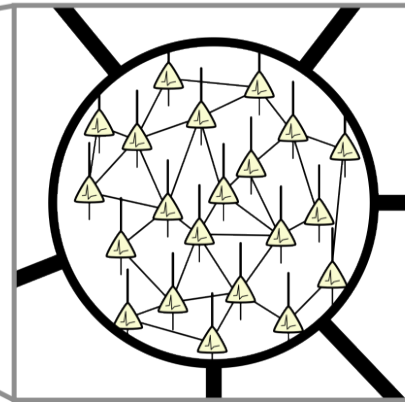




FEM – SNN MAPPING

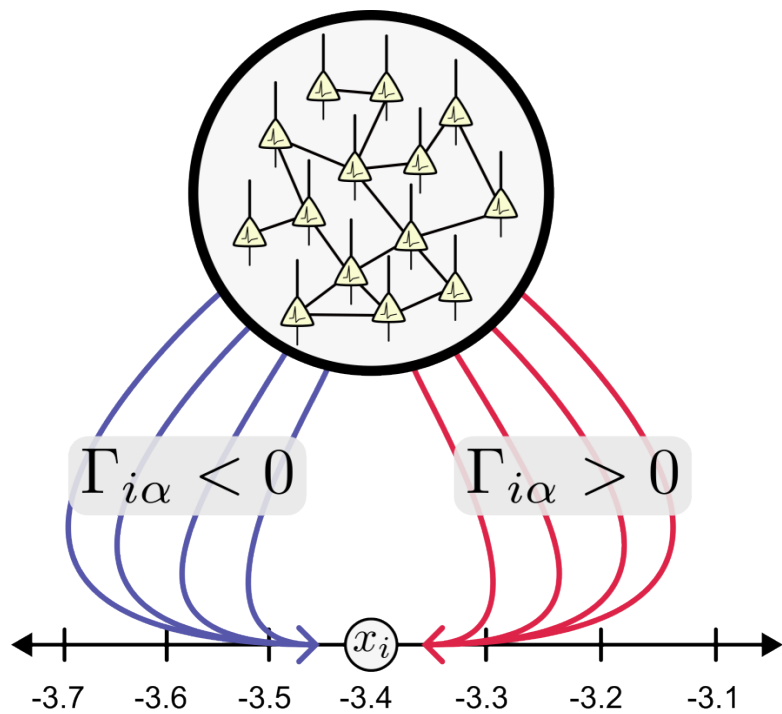


Each node “contains” a population of neurons
Neurons within a node connect locally within the node and to neighboring nodes

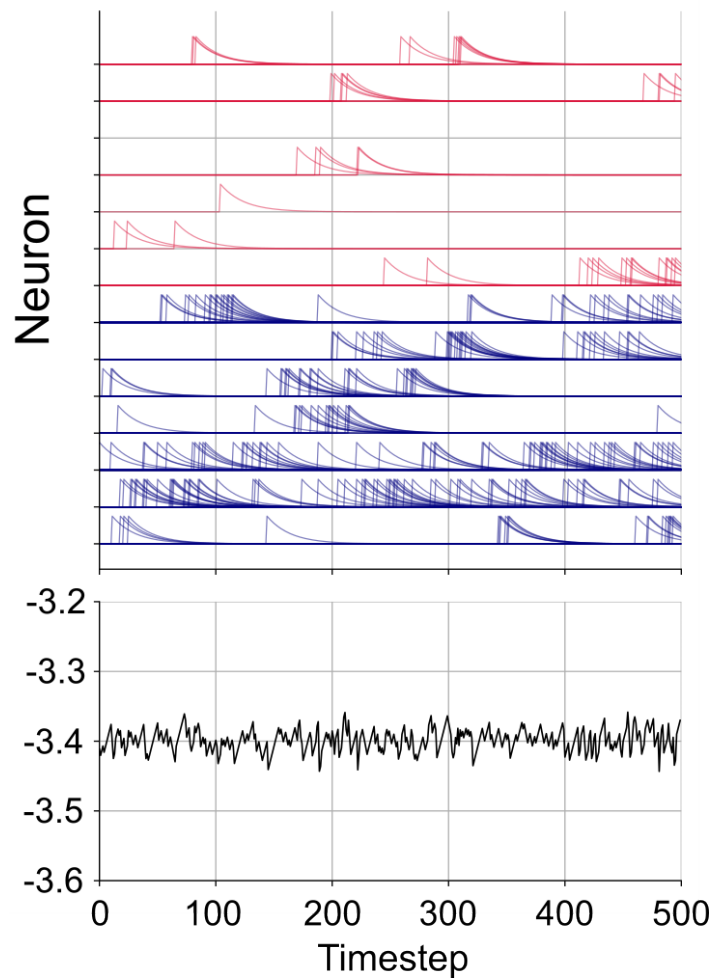


$$\dot{x}_i(t) = -\frac{x_i}{\tau} + \vec{\Gamma}_{in} S_n$$

Neurons within a node project to the solution variable associated to that node



$$\frac{dx_i}{dt} = -\lambda_d x_i + \Gamma_{i\alpha} s_\alpha$$





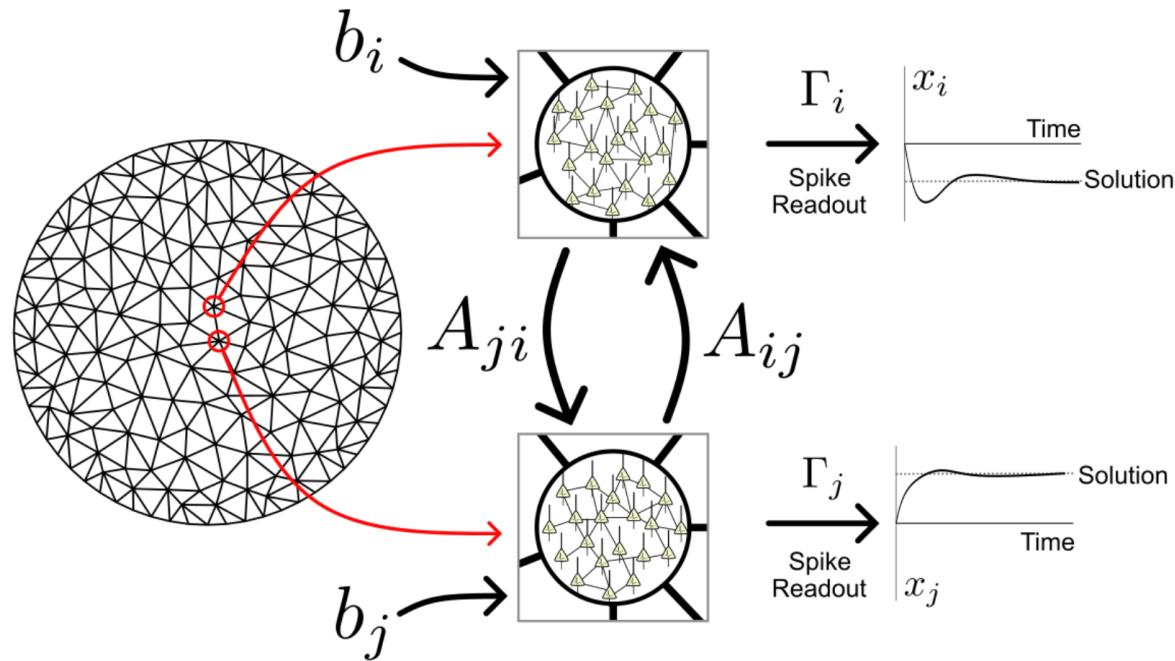
Poisson Equation

$$\nabla^2 u = f$$

Discretization

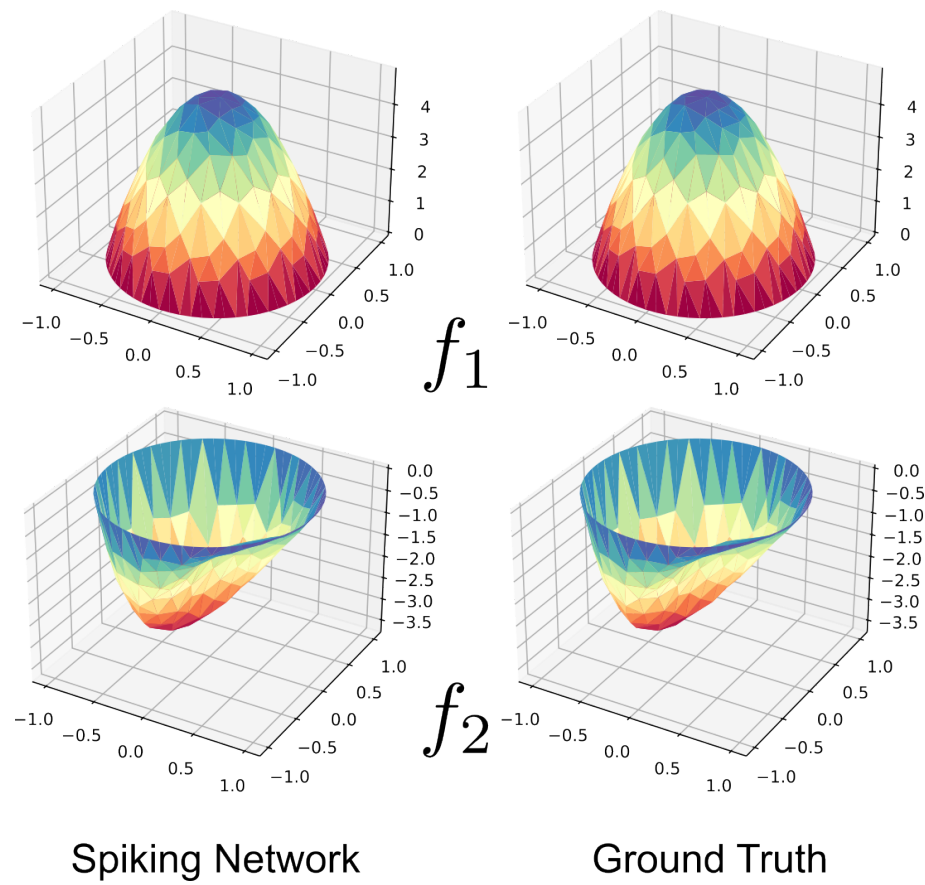
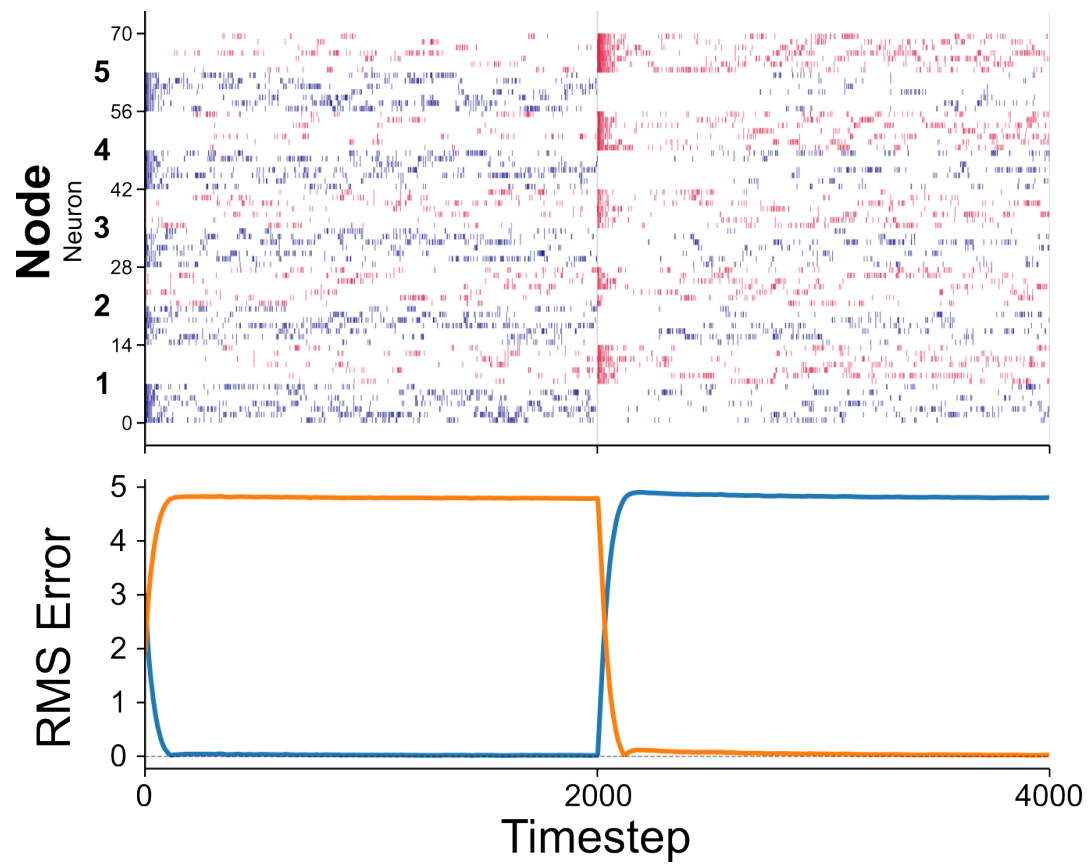
Sparse Linear System

$$A\vec{x} = \vec{b}$$



Neuromorphic virtues:

- Locally dense, globally sparse connectivity
- Scalable: neighbors $\sim O(1)$
- Sparse spiking activity



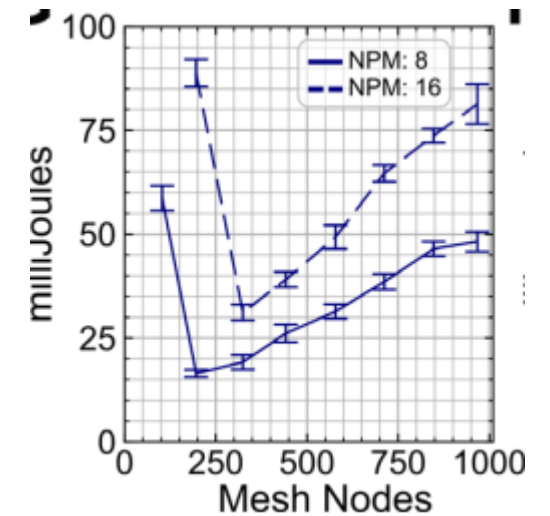
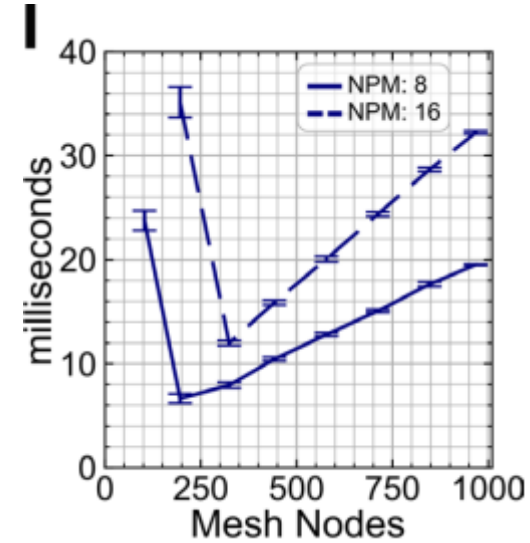
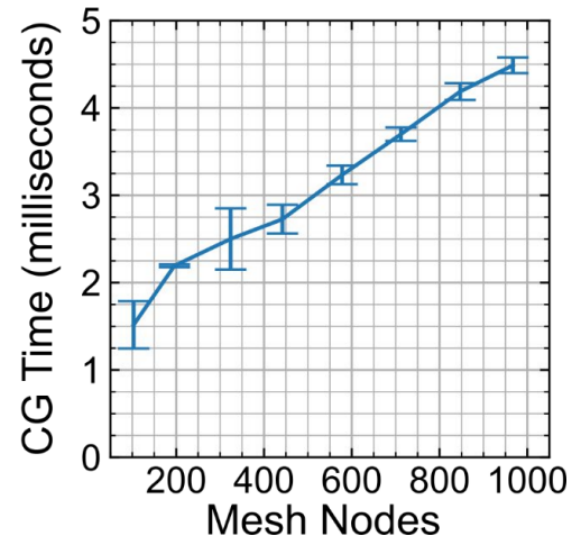
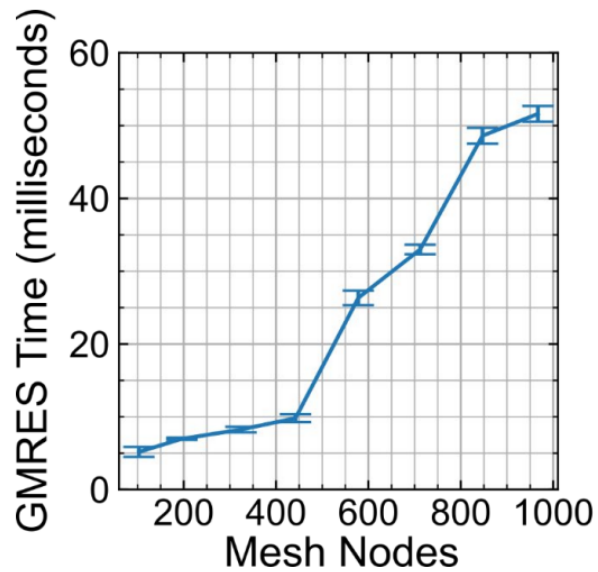
Spiking Network

Ground Truth



NeuroFEM is similar in speed to GMRES and Conjugate Gradient

Neuromorphic solution is general



Conjugate gradient is faster at small scales, but can only be used for symmetric matrices

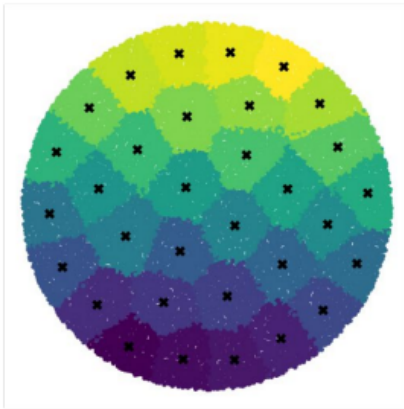


NeuroFEM shows compelling strong and weak scaling

Strong Scaling

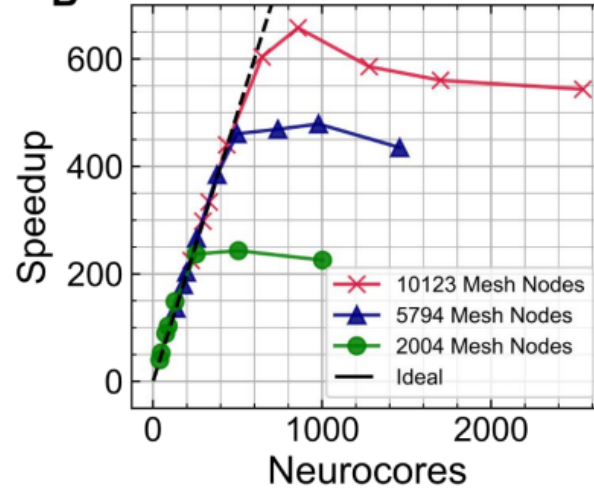
Weak Scaling

A

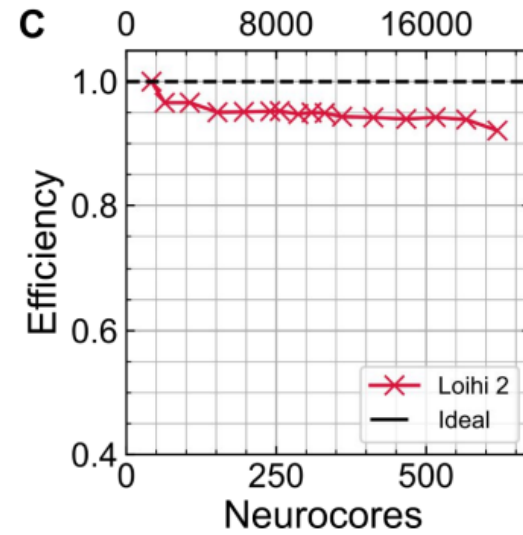


Mesh Nodes

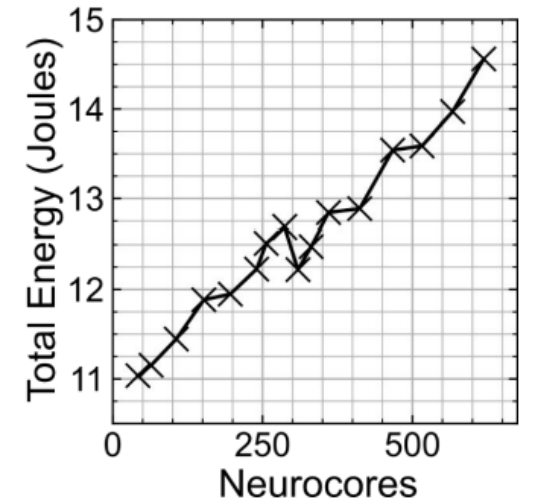
B



C

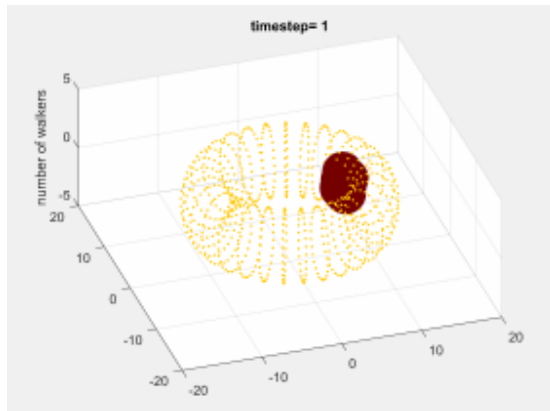


D

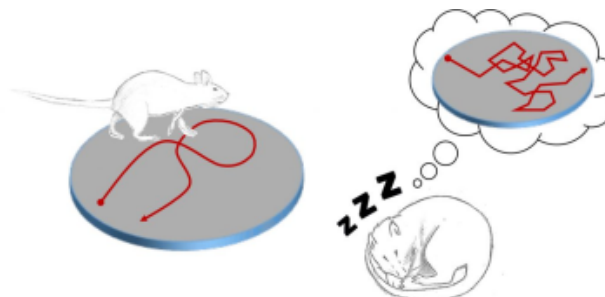




These neuromorphic advantages are perhaps closer to biology than people may realize



Widespread sampling by neural circuits



Neuron Article

Hippocampal Reactivation of Random Trajectories Resembling Brownian Diffusion

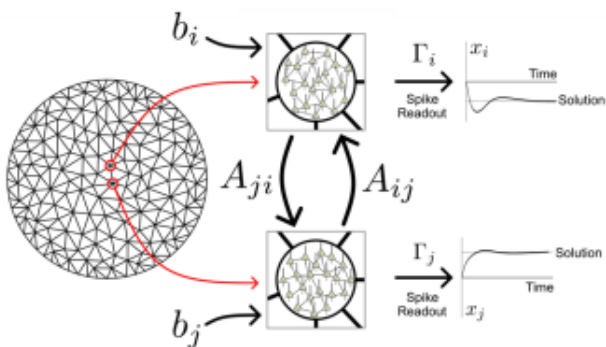
Federico Stella,^{1,2} Peter Barszelay,^{1,2} Joseph O'Neill,^{1,2} and Jozsef Csicsvari^{1,2,3*}

¹Institute of Science and Technology Austria (IST Austria) Am Campus 1, 3400 Klosterneuburg, Austria
²Present address: Department of Biochemistry, Institute of Biology, Eötvös Loránd University, Budapest 117, Hungary
³Present address: School of Psychology, Cardiff University, Park Place, Cardiff CF10 3AT, UK
 *Lead Contact
 *Correspondence: federico.stella@ist.ac.at (F.S.), jozsef.csicsvari@ist.ac.at (J.C.)
<https://doi.org/10.1016/j.neuron.2019.03.022>

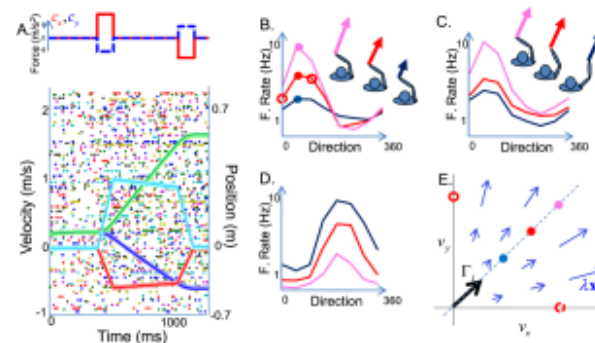
SUMMARY
 Hippocampal activity patterns representing movement trajectories are reactivated in immobility and sleep periods, a process associated with memory recall, consolidation, and decision making. It is thought that only fixed, behaviorally relevant patterns can be reactivated, which are stored across hippocampal synaptic connections. To test whether some generalized rules govern reactivation, we examined trajectory reactivation following non-stereotypical exploration of familiar open-field environments. We found that random trajectories of varying lengths and timescales were reactivated, resembling that of Brownian motion of particles. The animals' behavioral trajectory did not follow Brownian diffusion demonstrating that the exact behavioral experience is not reactivated. Therefore, hippocampal circuits are able to generate random trajectories of any recently active map by following diffusion dynamics. This ability of hippocampal circuits to generate representations of all behavioral outcome combinations, experienced or not, may underlie a wide variety of hippocampal-dependent cognitive functions such as learning, generalization, and planning.

INTRODUCTION

Poisson Equation $\nabla^2 u = f$ $\xrightarrow{\text{Discretization}}$ Sparse Linear System $A\vec{x} = \vec{b}$



Distributed control dynamics to describe action space



OPEN ACCESS | PLOS ONE

Predictive Coding of Dynamical Variables in Balanced Spiking Networks

Martin Boerlin¹, Christian K. Machens¹, Sophie Denier^{1*}

¹Department of Neurobiology, Max Planck Institute of Psychiatry, Munich, Germany

Abstract
 Free observation of the cortex has puzzled neuroscientists for a long time. First, neural responses are highly variable. Second, the level of excitation and inhibition received by each neuron is tightly balanced at all times. Here, we demonstrate that both properties are necessary consequences of neural networks that represent dynamical variables. Our approach is based on two assumptions: we assume that information about dynamical variables can be read out directly from neural spike rasters, and we assume that neurons only fire a spike if that improves the representation of the dynamical variable. Based on these assumptions, we derive a network of noisy integrate-and-fire neurons that is able to implement arbitrary linear dynamical systems. We show that the membrane voltage of the neurons is equivalent to a prediction error about a continuous resolution-level signal, among other things, our approach allows us to interpret the integration network of spiking neurons that is robust against many perturbations. Most importantly, neural variability in our networks cannot be equated to noise. Despite exhibiting the same single-unit properties as widely used population codes models (e.g. tuning curves, precise distributed spike trains, balanced networks) our networks are orders of magnitude more reliable. Our approach suggests that spikes do matter when considering how the brain computes, and that the reliability of neural representations could have been strongly underestimated.

Introduction
 While these studies explain how relatively deterministic single units can generate stochastic neural responses as they integrate information in time, they generally do not clearly how particular computations can be realized, not do they fundamentally answer why the brain would be organized in such a way.
 Here we show that the properties of balanced networks can be derived from a single efficiency principle, which is more obvious



Thank You!

Neuromorphic Monte Carlo Team

- Darby Smith, Michael Krygier, William Severa, Ojas Parekh, Rich Lehoucq, Aaron Hill

Neuromorphic Testbed Team

- Craig Vineyard, Suma Cardwell, Aaron Hill, Srideep Musuvathy, Felix Wang, Fred Rothganger, Frances Chance, Corinne Teeter, Mark Plagge, Ryan Dellana

Neuromorphic FEM

- Brad Theilman

COINFLIPS team

- 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

