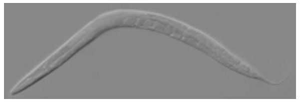
GPGPU ACCELERATED SIMULATION AND PARAMETER TUNING FOR NEUROMORPHIC APPLICATIONS

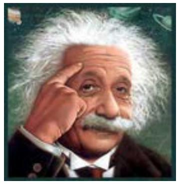
Kris Carlson, Michael Beyeler,
Nikil Dutt, Jeff Krichmar
Department of Cognitive Sciences
Department of Computer Science
University of California, Irvine, USA

Brains by the Numbers

Species	Neurons	Synapses
Nematode	302	10^3
Fruit Fly	100,000	10^7
Honeybee	960,000	10^9
Mouse	75,000,000	10^{11}
Cat	1,000,000,000	10^{13}
Human	85,000,000,000	10^{15}



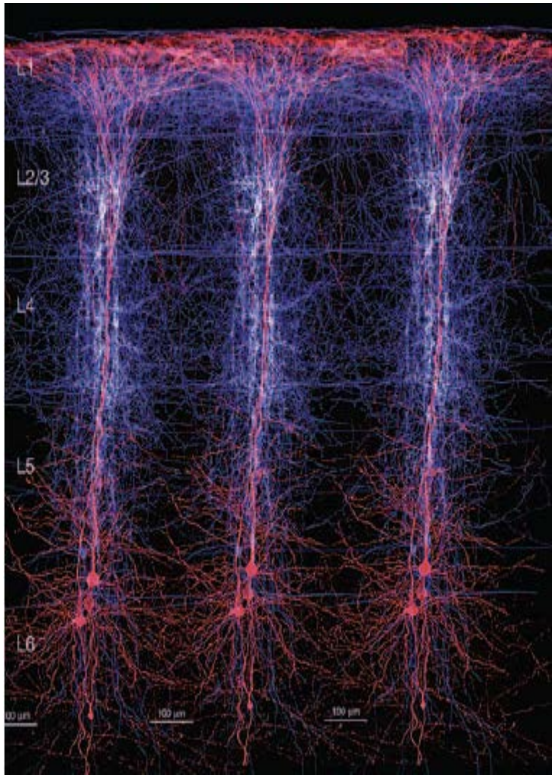


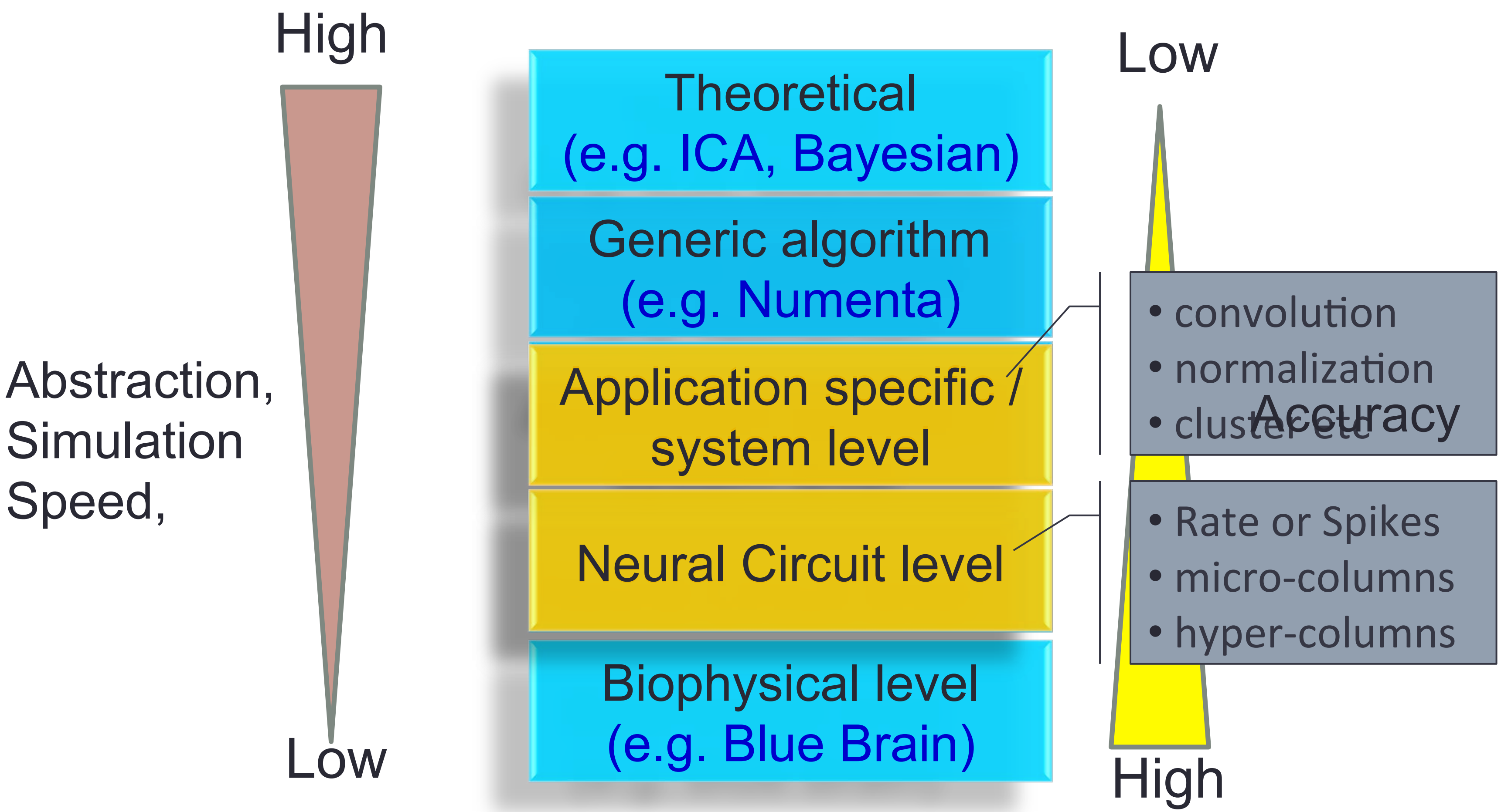
Source – http://en.wikipedia.org/wiki/List_of_animals_by_number_of_neurons

Brain Computations

- Massive parallelism (10^{11} neurons)
- Massive connectivity (10^{15} synapses)
- Excellent power-efficiency
  - ~ 20 W for 10^{16} flops
- Low-performance components (~100 Hz)
- Low-speed comm. (~meters/sec)
- Low-precision synaptic connections
- Probabilistic responses and fault-tolerant
- Autonomous learning



Hierarchy of Brain Model Abstractions



Outline

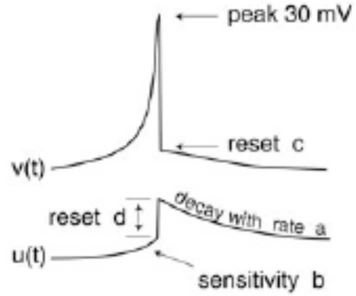
- **What are SNNs?**
- **Neuromorphic Hardware and Simulation Tools**
- **CARLSim SNN Simulator and Applications**
- **Parameter Tuning of Large-Scale SNNs**

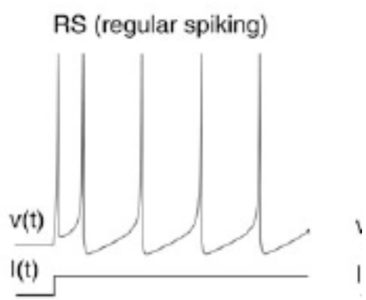
Spiking Neural Networks (SNNs)

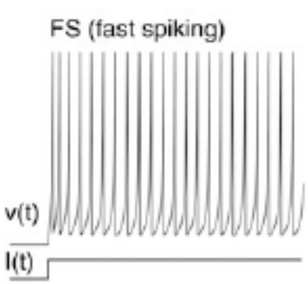
- **What are SNNs?**
  - Neural Networks that model neuronal/synaptic temporal dynamics
  - Spike only when the membrane voltage exceeds a threshold

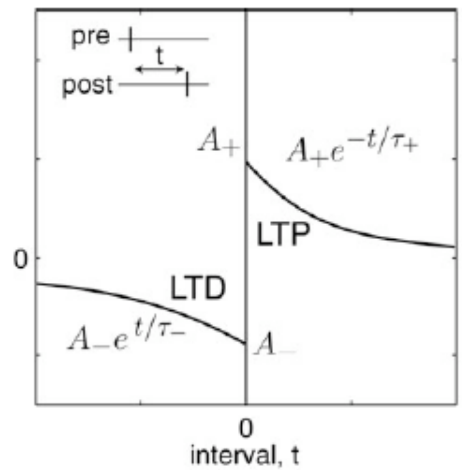
- **Why use SNNs?**
  - Speed of processing hypothesis
    - processing with a wave of spikes
  - Spikes are rare: average brain activity < 1Hz
    - "rates" are not energy efficient
  - SNNs use temporal coding but can still use rate coding
  - Event-driven nature of SNNs fits well with neuromorphic hardware
    - Use "**Address Event Representation**" (**AER**) to minimize communication
  - SNNs model important unsupervised learning algorithm: Spike Timing-Dependent Plasticity (STDP)
    - Precise spike firing has been found at almost all levels of the mammalian visual pathway
    - The exact timing of spike changes plasticity

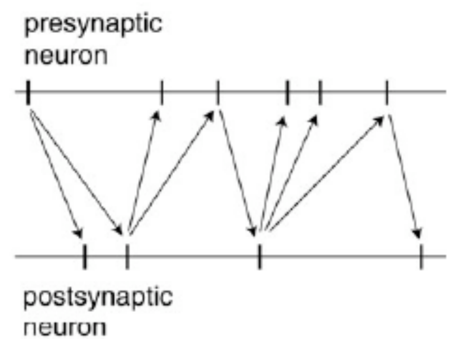
Modeling Components at the Neural Circuit Level

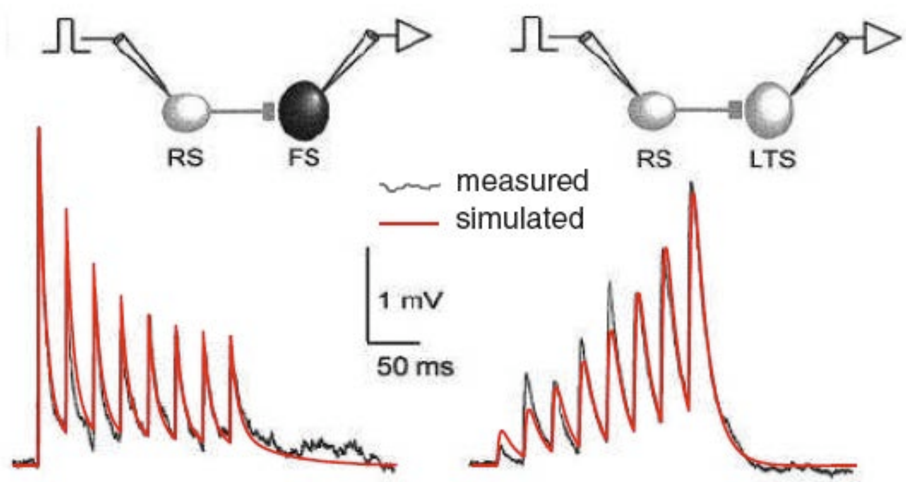












Many Parameters!

Outline

- What are SNNs?
- **Neuromorphic Hardware and Simulation Tools**
- CARLSim SNN Simulator and Applications
- Parameter Tuning of Large-Scale SNNs

Neuromorphic Hardware Devices

Hardware Project: Hardware Group	Hardware Description	Neuron Models	Synaptic Plasticity	Max Neurons	Max Synapses
SpiNNaker: Industry and UK universities	- Completely digital - Consists of array of nodes - Each node has 18 ARM9 cores - Final goal: 1,036,800 cores	Spiking: Izhikevich and non-spiking	Yes: STDP	1,000 neurons per ARM9 core*	10k synapses per ARM9 core*
Neurogrid: Stanford University	- Analog/digital hybrid - Full board has 16 neurochips - Operates on only 5 W	Spiking: Two-compartment neurons	No	65,536 neurons per neurochip	375M synapses per neurochip
True North Cog. Architecture: IBM SyNAPSE Team	- Completely digital - Consists of hierarchical design - Neurosynaptic core is basic building block	Spiking: many behaviors including LIF	No	256 neurons per neuro-synaptic core	256K binary synapses per neuro-synaptic core
HRL neural chip: HRL Labs, LLC SyNAPSE Team	- Analog/digital hybrid - Synaptic weights stored in memristors	Spiking: Izhikevich	Yes: STDP	576 neurons per chip	70k virtual synapses per chip
HiCANN: BrainScaleS Team	- Analog/digital hybrid - Each wafer has 384 chips - Neurons are analog - Synapses are digital	Spiking: AdExp and I&F	Yes: STDP	512 neurons per chip*	16k synapses per chip*

* Indicates the max. number of neurons are synapses are not independent

Neuromorphic Software Tools: SNN Simulators

Software Project	Features	Parallelized Implementations	Implementation Language	Parameter Tuning Tools
NENGO	- Uses neural engineering framework (NEF) - Set weights to perform specific computations - Uses both rate-based and spiking neurons - Uses neural plasticity rules (STDP) as well	None	- Core: Java - Python scripting	NEF
NEST	- Mature codebase for multiple platforms - Includes many neuron and plasticity models - Built-in simulation language interpreter - Module for creating complex networks	Parallelized MPI CPU implementation	- Core: C++ - Interface: Python - PyNN support	None
Brian	- Multiple integration methods - Multiple neuron and plasticity models - Uses Python plotting packages - Good documentation	Parallelized CPU support Parallelized GPU support only for tuning component	- Core: Python - PyNN support	Support for tuning neuron models
CARLsim	- Fast and efficient CUDA GPU implementation - Support for key ion channels - GPU parallelized general tuning framework - Includes highly optimized CUDA vision frontend	Parallelized CUDA GPU implementation	- Core: C++ and CUDA - Syntax similar to PyNN	General tuning framework using EAs

* Indicates the max. number of neurons are synapses are not independent

Outline

- What are SNNs?
- Neuromorphic Hardware and Simulation Tools
- **CARLSim SNN Simulator and Applications**
- Parameter Tuning of Large-Scale SNNs

Neuromorphic Software Tools: SNN Simulators

Software Project	Features	Parallelized Implementations	Implementation Language	Parameter Tuning Tools
NENGO	- Uses neural engineering framework (NEF) - Set weights to perform specific computations - Uses both rate-based and spiking neurons - Uses neural plasticity rules (STDP) as well	None	- Core: Java - Python scripting	NEF
NEST	- Mature codebase for multiple platforms - Includes many neuron and plasticity models - Built-in simulation language interpreter - Module for creating complex networks	Parallelized MPI CPU implementation	- Core: C++ - Interface: Python - PyNN support	None
Brian	- Multiple integration methods - Multiple neuron and plasticity models - Uses Python plotting packages - Good documentation	Parallelized CPU support Parallelized GPU support only for tuning component	- Core: Python - PyNN support	Support for tuning neuron models
CARLsim	**- Fast and efficient CUDA GPU impl.** **- Support for key ion channels** **- GPU parallelized general tuning framework** **- Includes highly optimized CUDA vision frontend**	**Parallelized CUDA GPU implementation**	**- Core: C++ and CUDA** **- Syntax similar to PyNN**	**General tuning framework using EAs**

CARLsim

frontiers in
NEUROINFORMATICS

ORIGINAL RESEARCH ARTICLE
published: 14 September 2011
doi: 10.3389/fninf.2011.00019

An efficient simulation environment for modeling large-scale cortical processing

Micah Richert[1], Jayram Moorkanikara Nageswaran[2], Nikil Dutt[2] and Jeffrey L. Krichmar[1,2] *

[1] *Department of Cognitive Sciences, University of California, Irvine, CA, USA*
[2] *Department of Computer Science, University of California, Irvine, CA, USA*

Edited by:
Andrew P. Davison, CNRS, France

Reviewed by:
Eilif Muller, École Polytechnique Fédérale de Lausanne, Switzerland
Romain Brette, École Normale Supérieure de Paris, France
Andreas Kirkeby Fidjeland, Imperial College London, UK

****Correspondence:***
Jeffrey L. Krichmar, Department of Cognitive Sciences, University of California, 2328 Social and Behavioral Sciences Gateway, Irvine, CA 92697-5100, USA.
e-mail: jkrichma@uci.edu

We have developed a spiking neural network simulator, which is both easy to use and computationally efficient, for the generation of large-scale computational neuroscience models. The simulator implements current or conductance based Izhikevich neuron networks, having spike-timing dependent plasticity and short-term plasticity. It uses a standard network construction interface. The simulator allows for execution on either GPUs or CPUs. The simulator, which is written in C/C++, allows for both fine grain and coarse grain specificity of a host of parameters. We demonstrate the ease of use and computational efficiency of this model by implementing a large-scale model of cortical areas V1, V4, and area MT. The complete model, which has 138,240 neurons and approximately 30 million synapses, runs in real-time on an off-the-shelf GPU. The simulator source code, as well as the source code for the cortical model examples is publicly available.

Keywords: visual cortex, spiking neurons, STDP, short-term plasticity, simulation, computational neuroscience, software, GPU

Code available at: http://www.socsci.uci.edu/~jkrichma/CARLsim

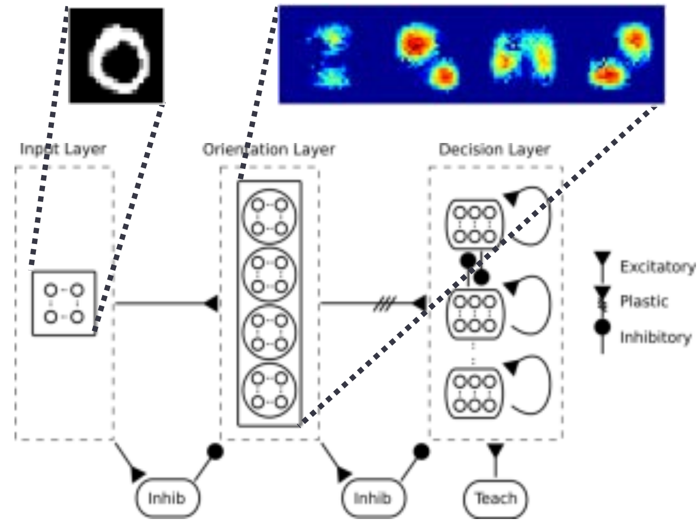
CARLsim Applications

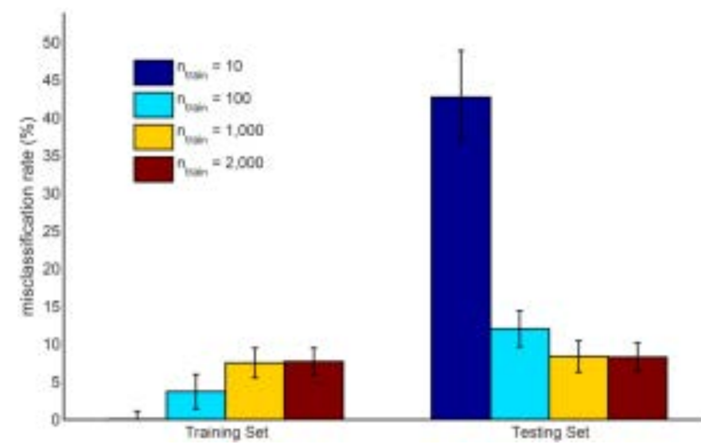
- Used to create large-scale simulations of cognitive processes
  - 10k – 100K neurons with millions of synapses

- Sample Applications
  - Visual Processing
    - Large-scale model of cortical areas V1, V4, and area MT *[Richert et al. 2011]*
  - Neuromodulation
    - Top-down and bottom-up attention *[Avery et al. 2013a]*
    - Working memory and behavior *[Avery et al. 2013b]*
  - Object Categorization
    - Classifying handwritten digits *[Beyeler et al. 2013]*
  - Neural Plasticity
    - Biologically plausible STDP and Homeostatis *[Carlson et al. 2013]*

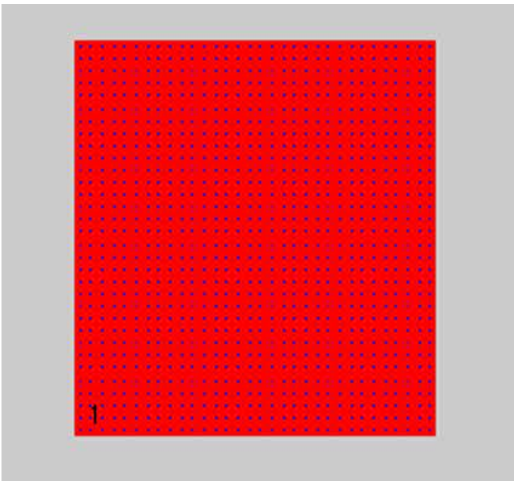
CARLsim Example: MNIST Classification

- MNIST database of handwritten digits
  - 60,000 training / 10,000 test patterns
- Neurofidelity
  - Network: spiking neurons, ion channels
  - Training: Calcium-based learning rule (STDP-like)
  - Classification: decision-making (drift-diffusion / race model)
- Accuracy
  - 92 % correct classifications
  - Biologically plausible reaction times
- Efficiency
  - 71,026 neurons, ~133 million synapses
  - Real-time processing on a single NVIDIA Tesla M2090
- Scalability
  - Modular approach
  - Extend to more general neuromorphic implementation

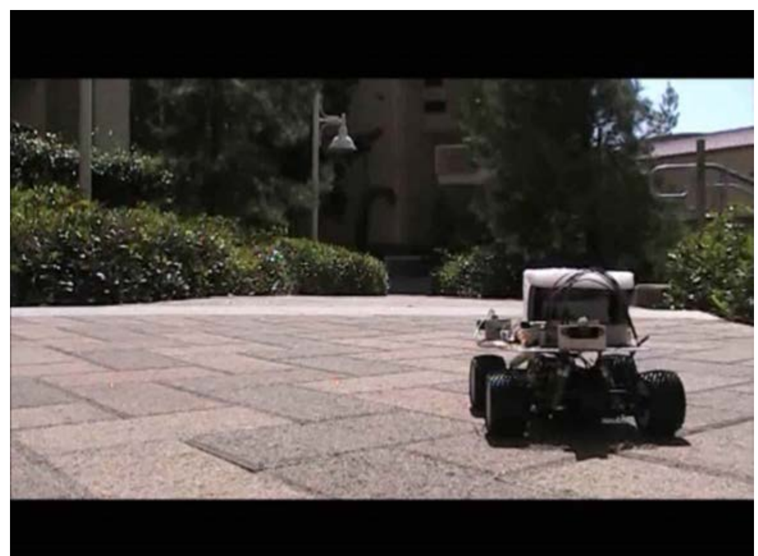




CARLsim Example: Vision Processing System



- Implemented motion selectivity in spiking model of Area MT
- 32x32 Resolution, 138,240 neurons; ~30 million synapses (shown on the left)
  - Running in real-time on single GPU card.
- Higher resolution, ~550K neurons; ~120M synapses (shown below)



Outline

- What are SNNs?
- Neuromorphic Hardware and Simulation Tools
- CARLSim SNN Simulator and Applications
- **Parameter Tuning of Large-Scale SNNs**

Parameter Tuning of Neurobiologically Inspired Networks: Time Consuming!

- **Example 1*:** Tuning a small, lobster pyloric circuit:
  - Tuned 3-neuron, 5 synapse circuit
  - Database of 20M configurations was generated, with only 20% functional
- **Example 2+:** Tuning small SNN:
  - 5 parameters with 10 values, each needs 10^5 simulations
  - Simulation Time: ~15 days for 1K network with 5 parameters
- **Lesson:**
  - Even for small networks, search space is extremely large with many solutions!
- ***Biology provides some constraints, BUT***
  - designer must choose many parameter values ***manually*** to achieve appropriate neuronal dynamics.
- ***Our Solution: Automated Parameter Tuning using GPUs and EAs***

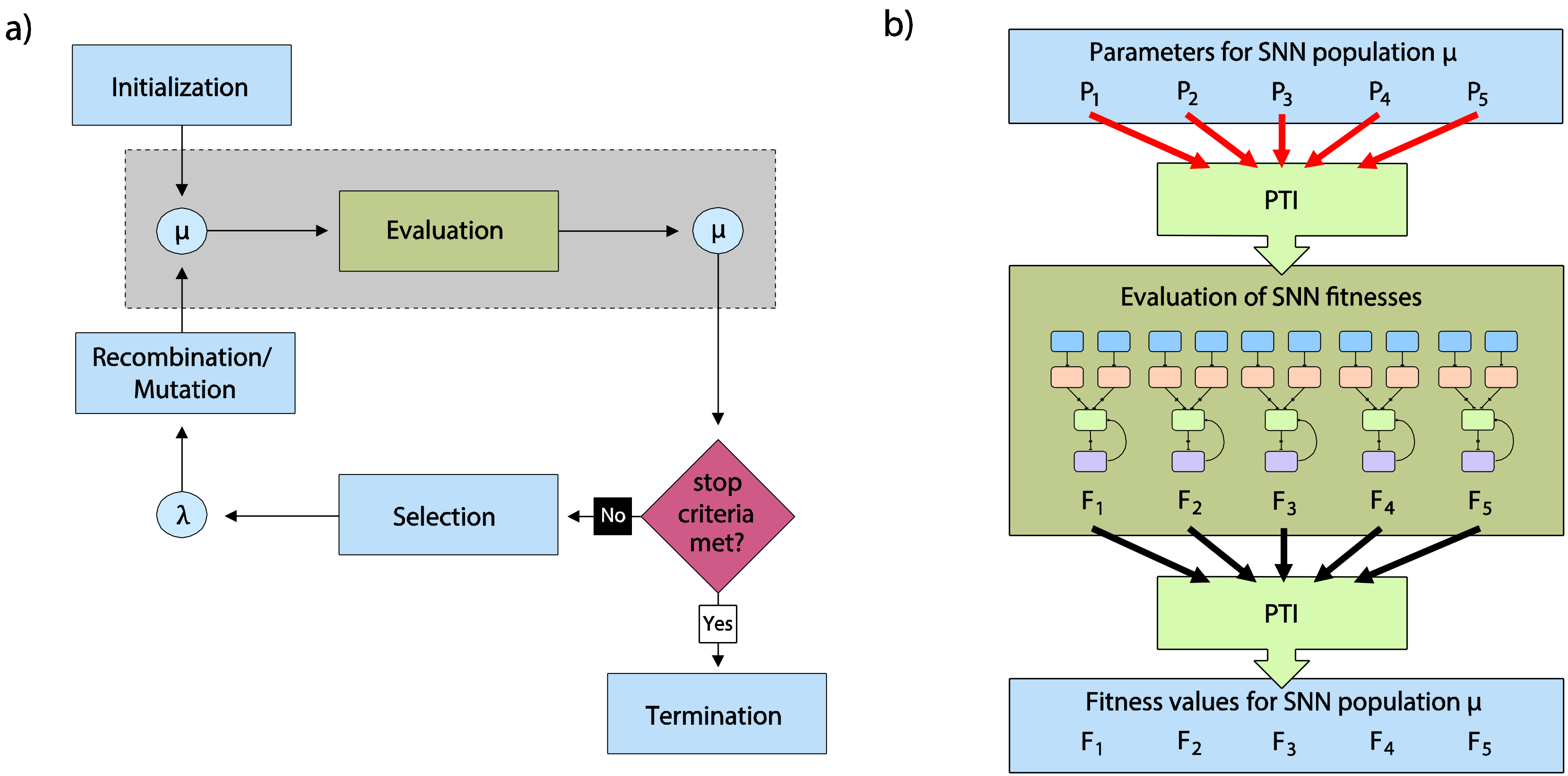
*From Prinz, A.A., Bucher, D., and Marder, E. (2004). Similar network activity from disparate circuit parameters. Nat Neurosci 7, 1345-1352

+From Jayram Moorkanikara's PhD Dissertation at UCI, 2010

Automated Parameter Tuning Framework

- Automated parameter-tuning framework can quickly and efficiently tune large-scale spiking neural networks
- Leverage
  - Recent progress in evolutionary algorithms.
  - Optimization with off-the-shelf graphics processing units (GPUs)
- The parameter search guided by principles of neuroscience
  - Biological networks adapt their responses to
    - increase the amount of transmitted information, reduce redundancies, and span the stimulus space

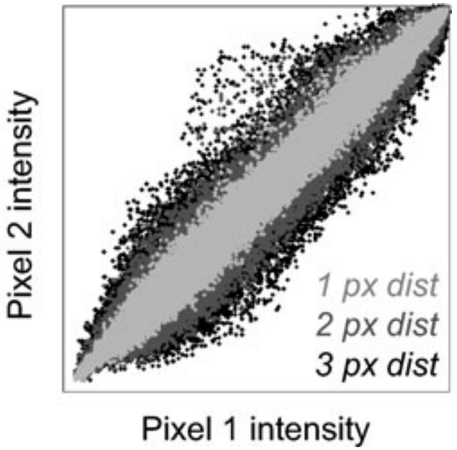
Automated Parameter Framework for Tuning Spiking Neural Networks (SNNs)

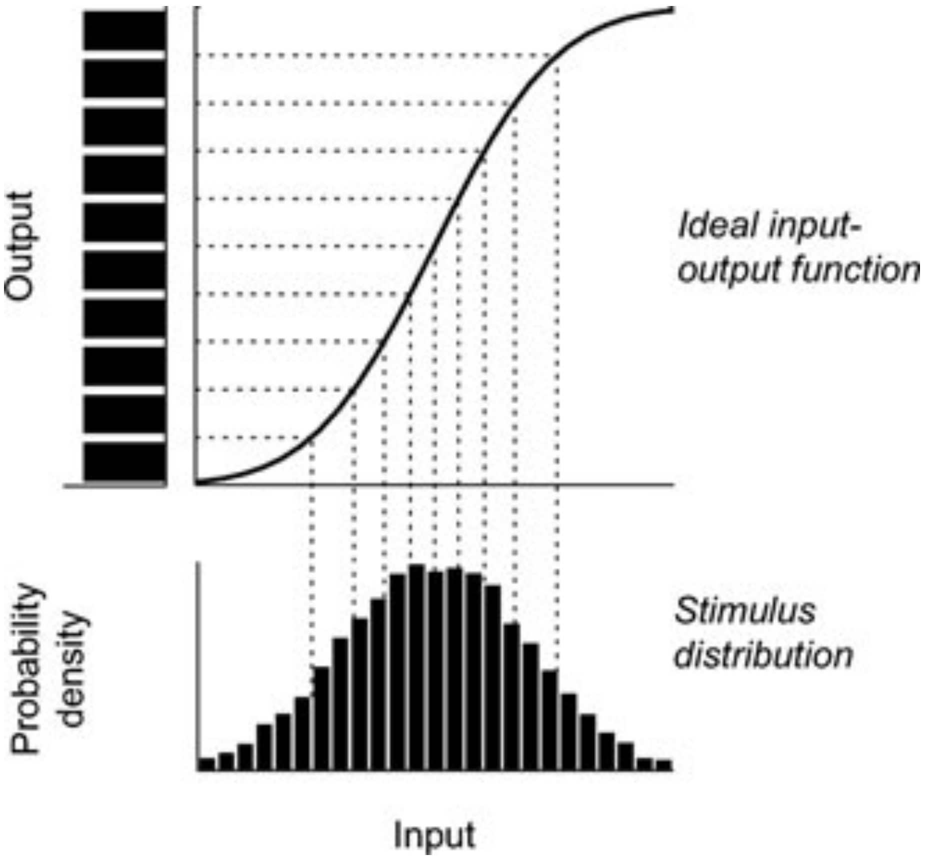


Efficient Coding Hypothesis

- Fundamental idea:
  - Sensory systems adapt their responses to the regularities of their input
  - Increase the amount of transmitted information at any given time

1. Maximize efficiency (*reduce redundancy*)
2. Responses should be independent of one another (*decorrelation*)
3. A stimulus should involve only a small fraction of the available neurons (*sparse*)





Louie, K., and Glimcher, P.W. (2012). Efficient coding and the neural representation of value. Ann N Y Acad Sci *1251*, 13-32.

Fitness Function Based on the Efficient Coding Hypothesis

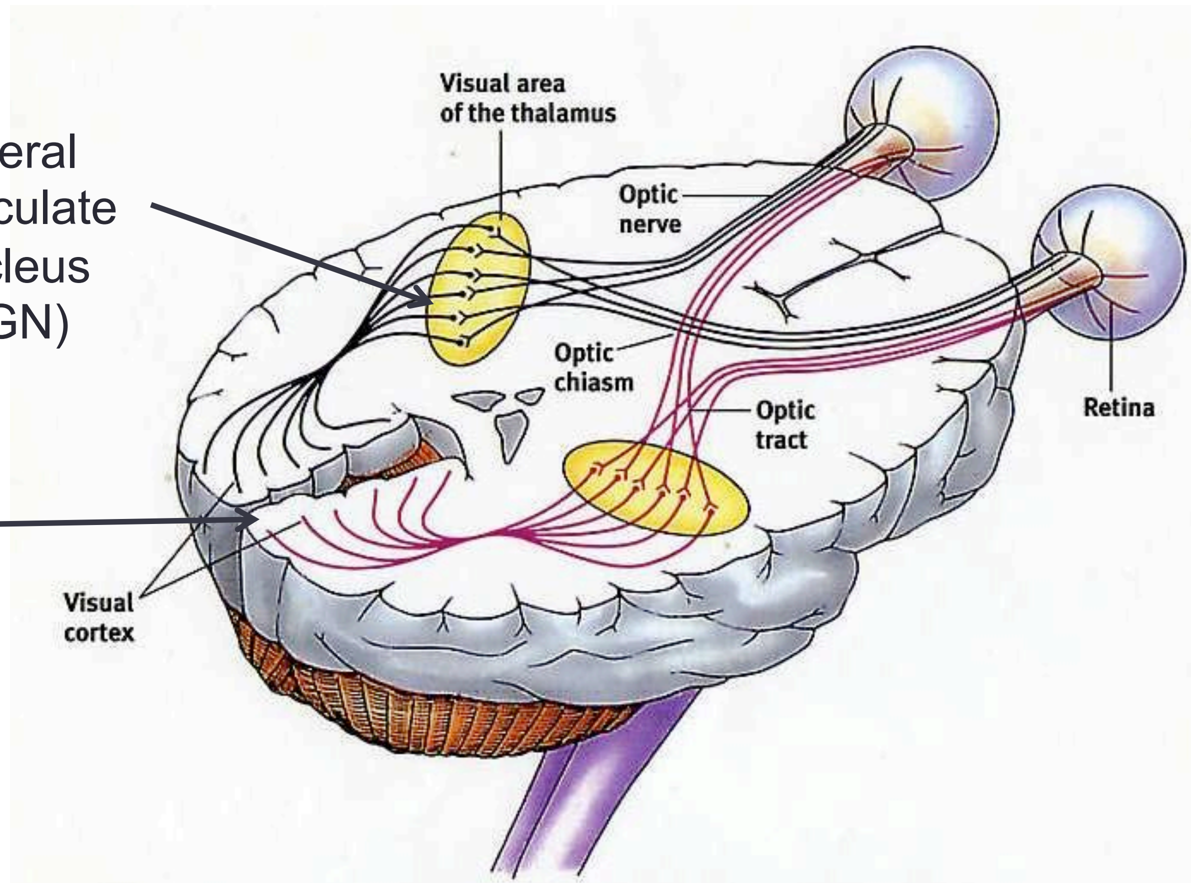
$$Fitness_{\text{total}} = \frac{1}{Fitness_{\text{decorr}} + Fitness_{\text{Gauss}} + K_{\text{scaling factor}} \cdot Fitness_{\text{maxRate}}}$$

- *$Fitness_{\text{decorr}}$* ensured decorrelation by forcing each neuron to respond maximally to different stimuli
- *$Fitness_{\text{Gauss}}$* required Gaussian tuning curves. Also lead neurons to employ their full response range to describe the stimulus
- *$Fitness_{\text{maxRate}}$* limited the maximum firing rate of each neuron which contributes to sparsity

Visual Pathway in the Brain

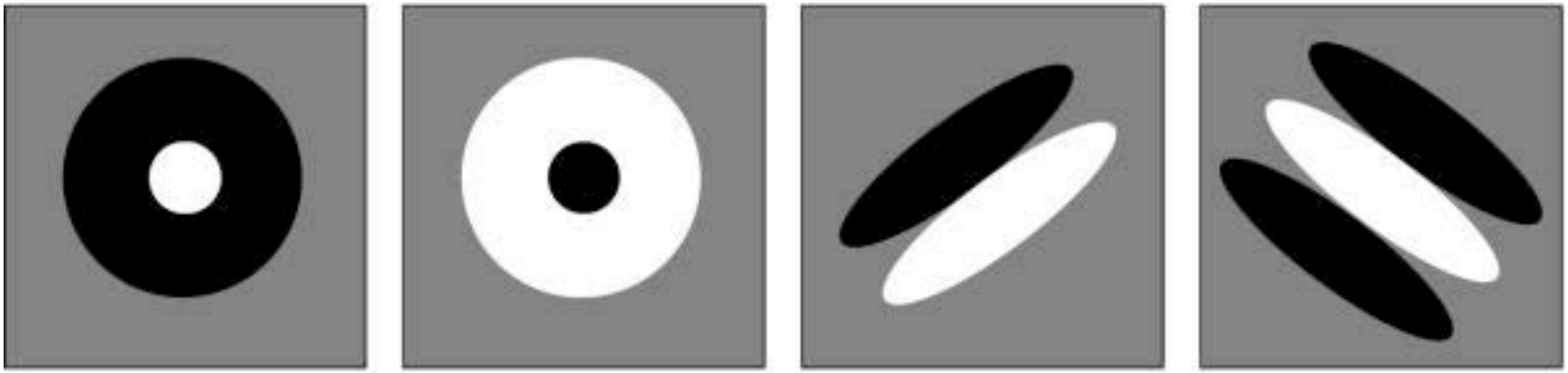
Lateral Geniculate Nucleus (LGN)

Primary Visual Cortex (V1)



- Sample Parameter Tuning Example for Homeostasis and STDP:
  - Unsupervised learning of V1 simple cell receptive fields in response to patterned inputs.

Receptive Fields in LGN and V1



(*a*) ON cell in retina or LGN

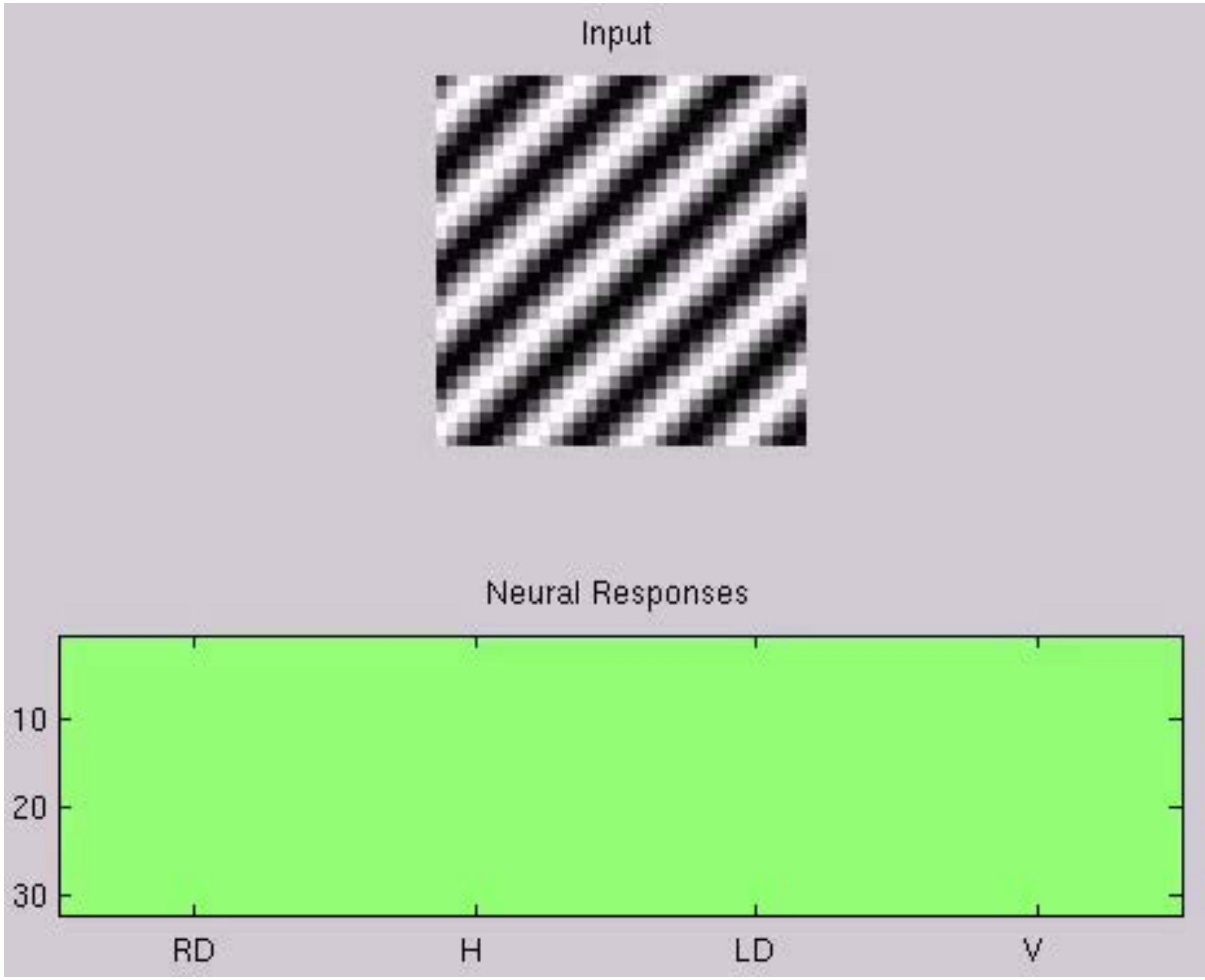
(*b*) OFF cell in retina or LGN

(*c*) 2-lobe V1 simple cell

(*d*) 3-lobe V1 simple cell

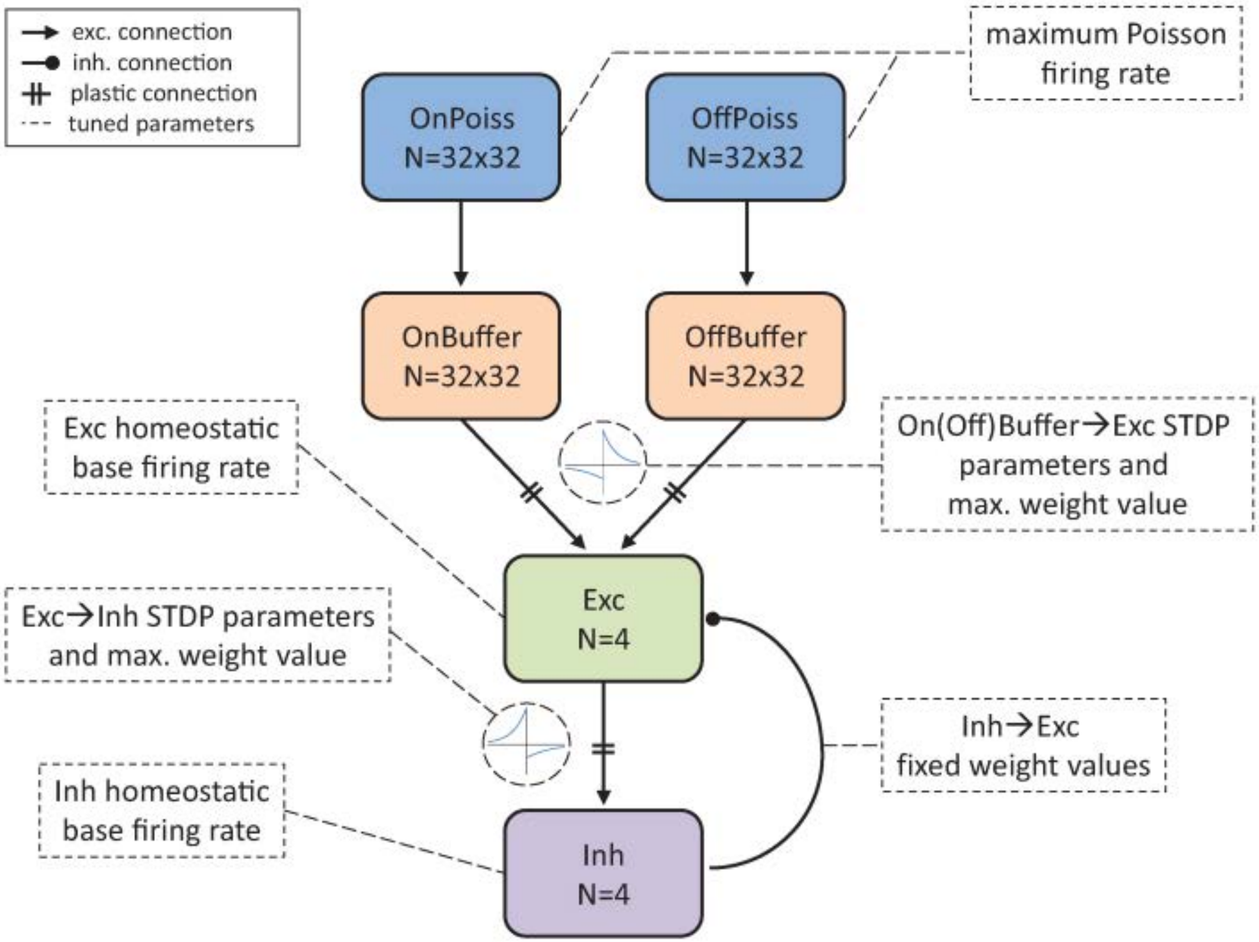
K.D. Carlson, M. Richert, N. Dutt and J.L. Krichmar, "Biologically Plausible Models of Homeostasis and STDP: Stability and Learning in Spiking Neural Networks," in Proc. IJCNN 2013

Simulated Visual Cortex Responses to Sinusoidal Gratings



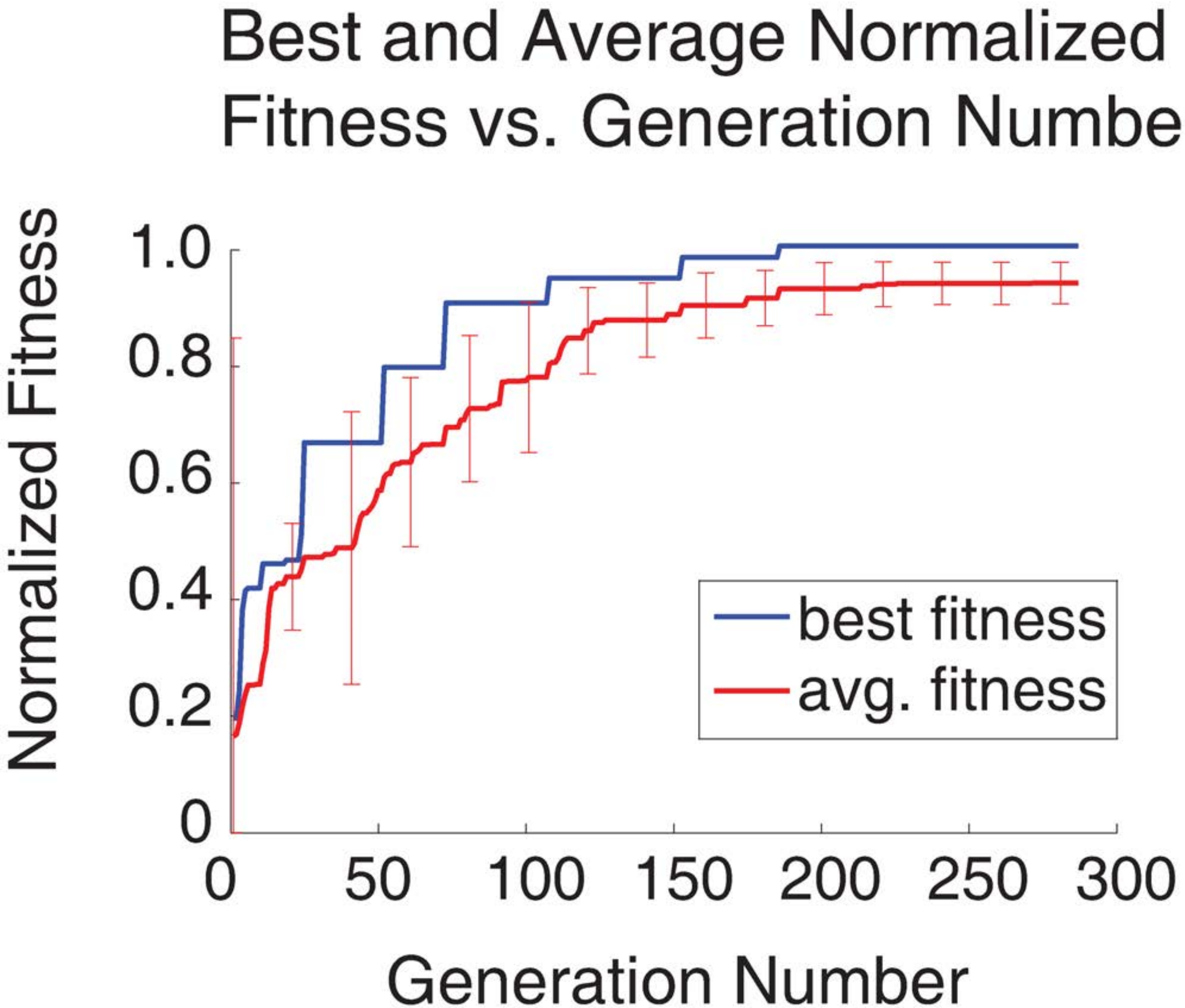
K.D. Carlson, M. Richert, N. Dutt and J.L. Krichmar, “Biologically Plausible Models of Homeostasis and STDP: Stability and Learning in Spiking Neural Networks,” in Proc. IJCNN 2013

Tuning SNNs that Generate Self Organizing Receptive Fields (SORF)



- Network size
  - 4104 neurons
- Indirect encoding
  - 14 parameters to search
- Training phase
  - 40 sinusoidal orientations presented
  - 2400 presentations
- Testing phase
  - 8 sinusoidal orientations presented to the network
  - Responses of the Exc neurons were evaluated using the ECH fitness function

K.D. Carlson, M. Richert, N. Dutt and J.L. Krichmar, “Biologically Plausible Models of Homeostasis and STDP: Stability and Learning in Spiking Neural Networks,” in Proc. IJCNN 2013



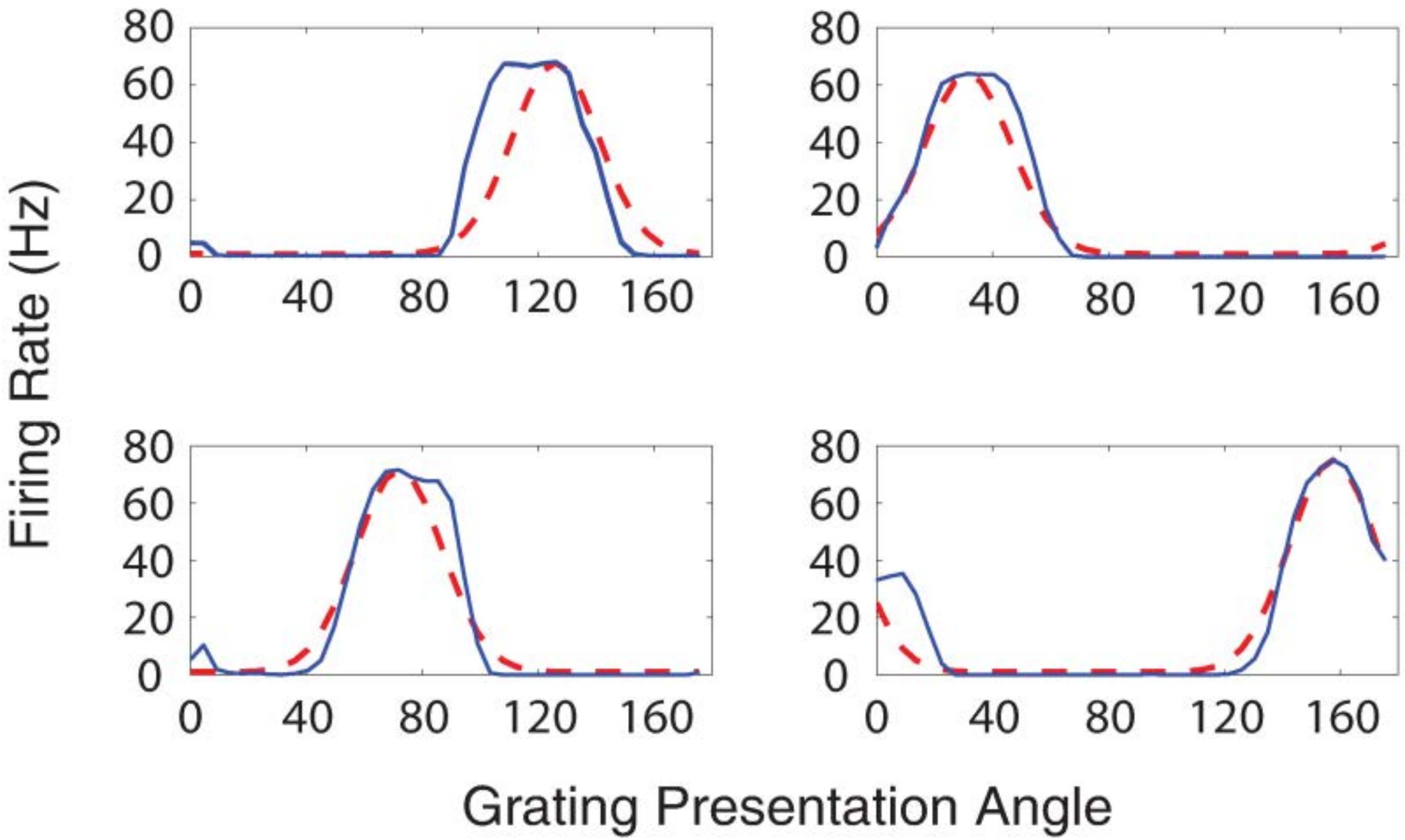
- 10 SNNs in parallel
  - Up to 40.
- 287 EA generations in 127 hours.
- Run on a single NVIDIA Tesla M2090

K.D. Carlson, M. Richert, N. Dutt and J.L. Krichmar, “Biologically Plausible Models of Homeostasis and STDP: Stability and Learning in Spiking Neural Networks,” in Proc. IJCNN 2013

Synaptic Weight Progression During Training for a High Fitness Individual

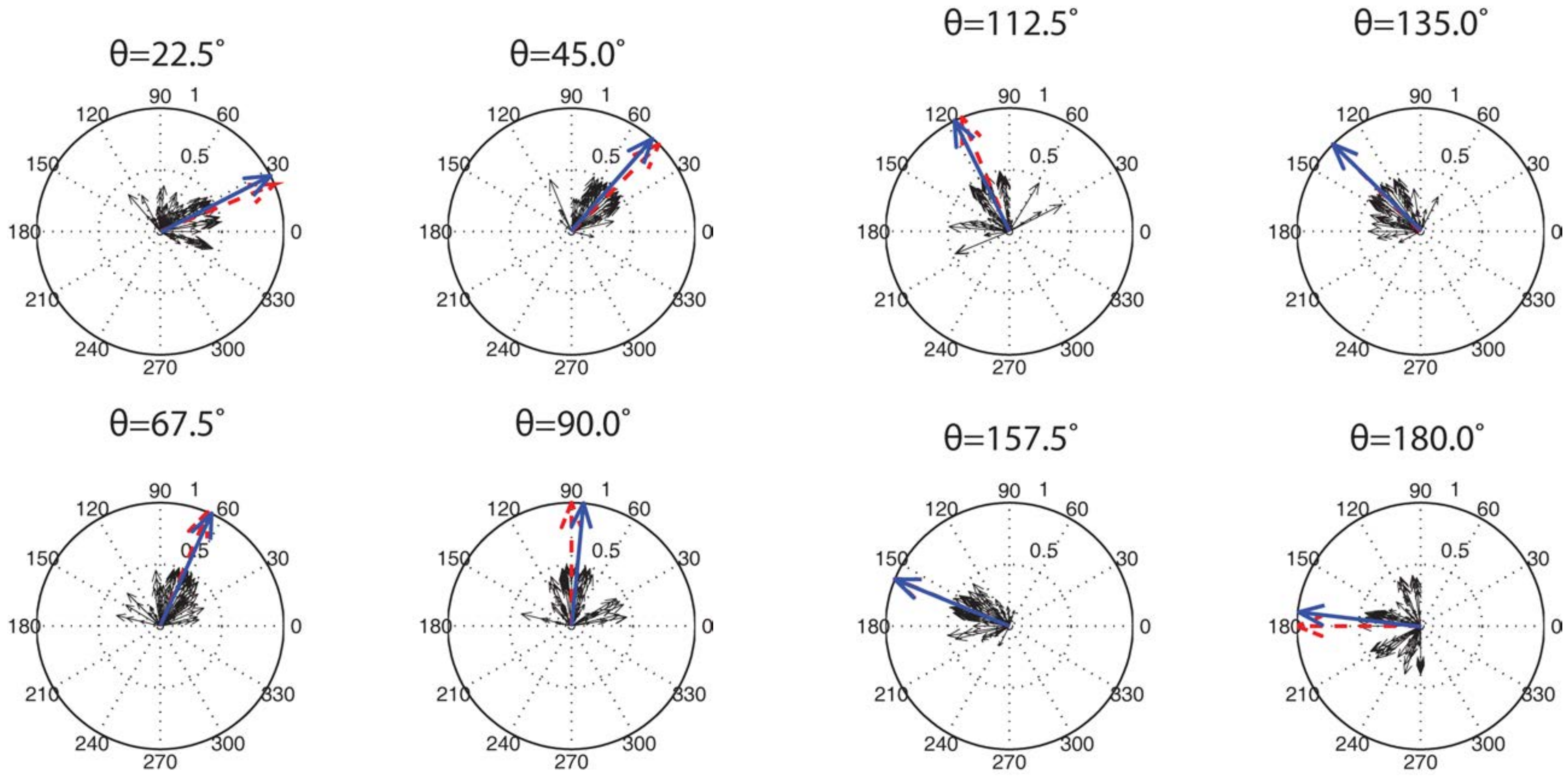
- Synaptic weights for the On(Off)Buffer→Exc connections
  - Light regions denote strong weights, dark regions denote weak weights

Response Of Neurons To Sinusoidal Gratings



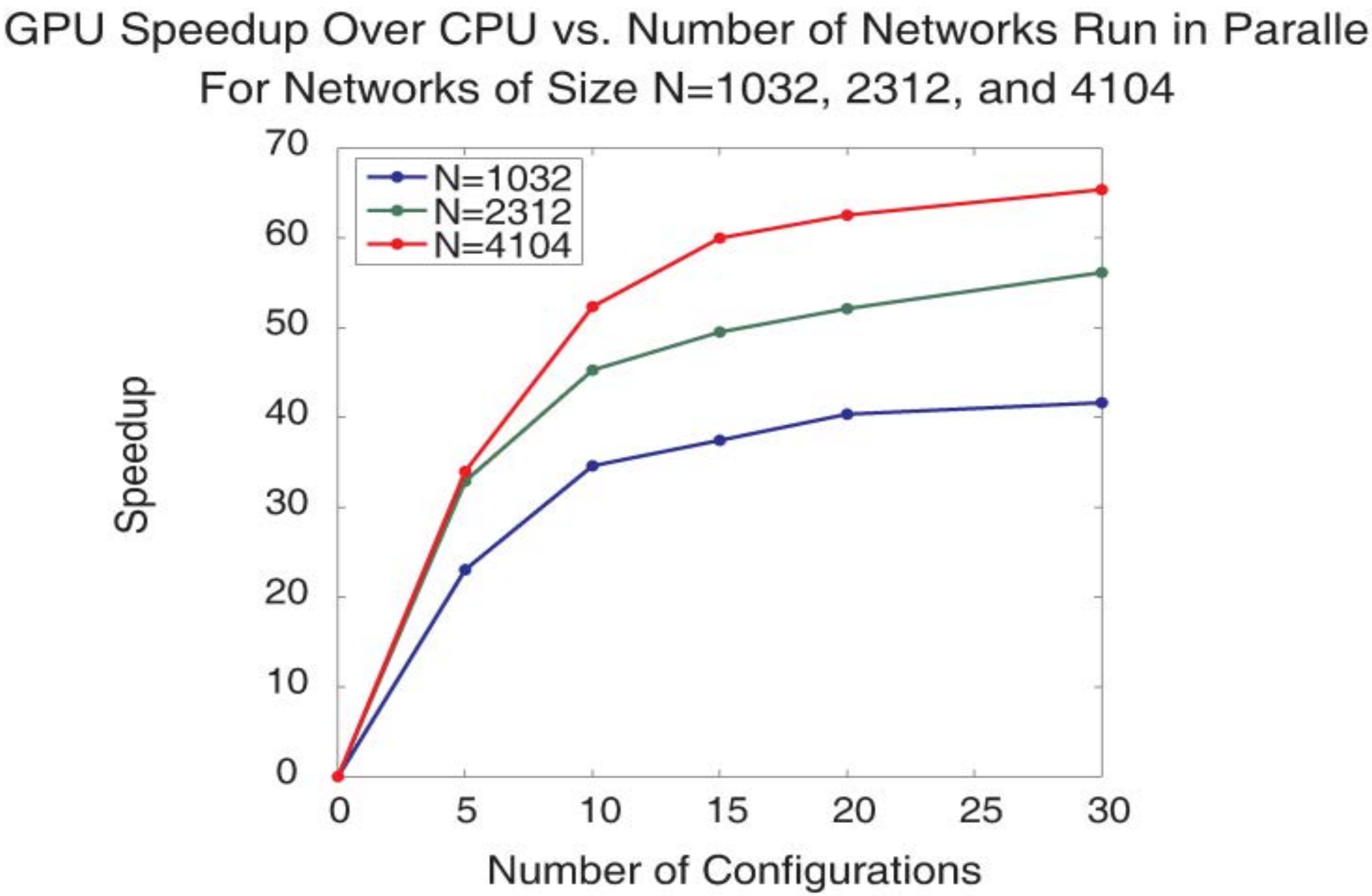
- **Blue line:** firing rate of simulated individual Exc group neuron
- **Red line:** ideal Gaussian tuning curve firing rate response for V1

Population Response After Evolution



- **Population decoding for diff presentation angles (70 sets of 4 exc neurons):**
  - Small black arrows: component vectors for individual neurons
  - **Blue arrow: normalized decoded population vector**
  - **Red arrow: normalized ideal grating presentation vector**

Automated Parameter Tuning Framework Performance



K.D. Carlson, M. Richert, N. Dutt and J.L. Krichmar, "Biologically Plausible Models of Homeostasis and STDP: Stability and Learning in Spiking Neural Networks," in Proc. IJCNN 2013

Conclusions

- Large-scale, complex, realistic brain simulations are necessary:
  - For the field of neuromorphic engineering to produce results and applications of practical value
  - To help computational neuroscientists develop new theories of neural function
- To address this challenge:
  - our approach leverages
    - the optimization capabilities of evolutionary computation
    - exploits graphical processing unit (GPU) parallelism
- The efficient coding hypothesis
  - May provide a metric for tuning networks of simulated spiking neurons
  - May also given insights into how real brain networks process information and achieve homeostasis

Thanks to…

Team CARL Now (2014) – UC Irvine

Front row – Alexis Craig, Emily Rounds, Kris Carlson, Liam Bucci, Ting-Shuo Chou

Back row – Feng Rong, Andrew Zaldivar, Nicolas Oros, Jeff Krichmar, Derrik Asher, Michael Beyeler, Nik Dutt

Team CARL Then (2009) – UC Irvine

Front row – Jay Nageswaran, Mike Avery, Chelsea Guthrie, Micah Richert

Back row – Andrew Zaldivar, Brian Cox, Michael Wei, Jeff Krichmar

Supported by the National Science Foundation, the Defense Advanced Research Projects Agency, and the Intelligence Advanced Research Projects Activity.

Thanks!

Questions or Comments???