Next Generation Neuroscience Simulation Platforms

- Challenges and Chances from an Engineering Perspective -

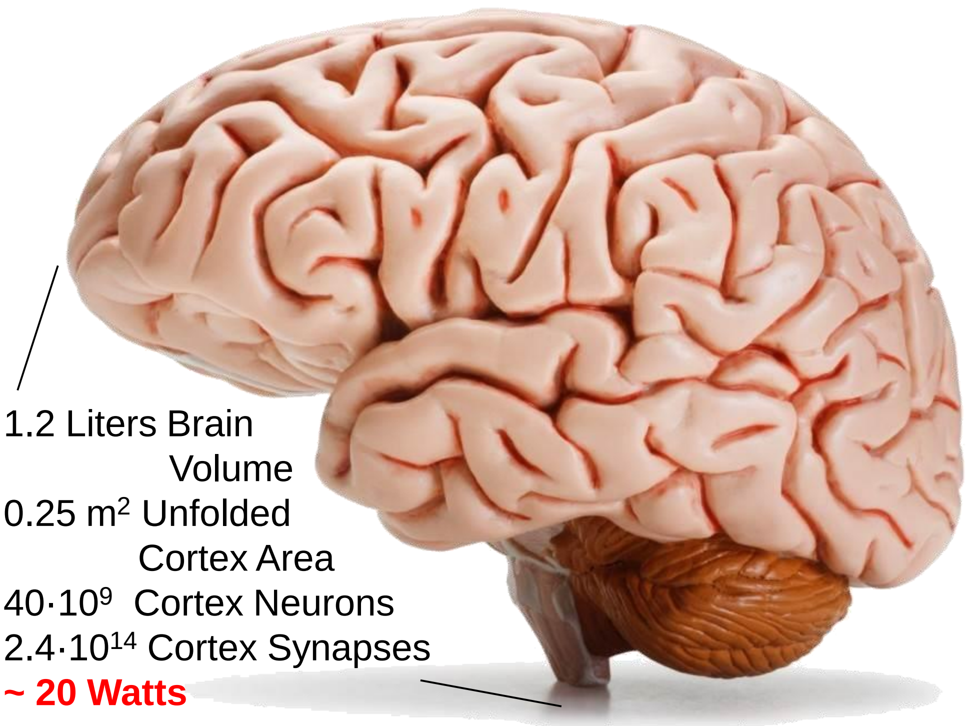
Tobias G. Noll

Special thanks to:
Georgia Psychou, Arne Heittmann, Markus Diesmann, and Tom Tetzlaff !

Outline

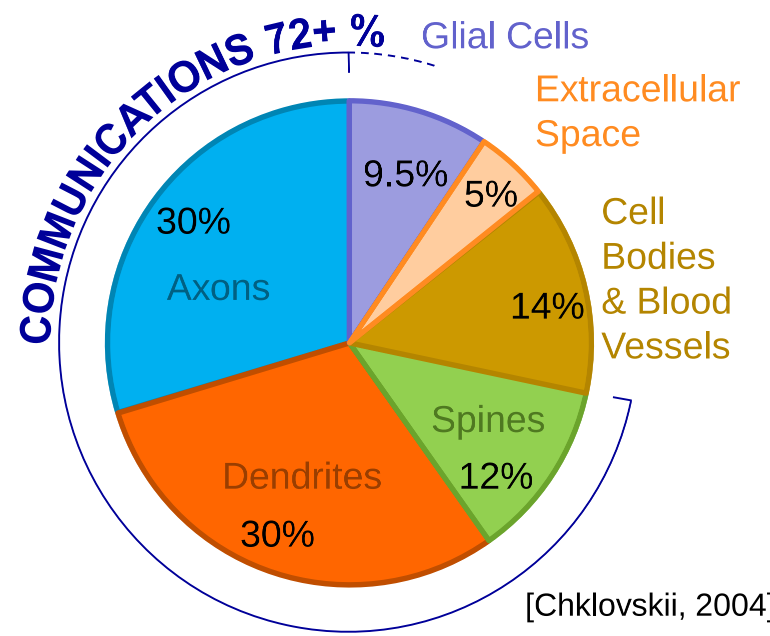
- ❒ Introduction
- ❒ Neuronal Network Components and Models
- ❒ Exemplary Large-Scale Networks
- ❒ Future Requirements
- ❒ Simulation Principles
  - Computation: Numerical ODE Solvers
  - Communication: AER
- ❒ State-of-the Art
- ❒ Brick Walls
  - Computation: …
  - Communication: …
- ❒ Conclusion

The Human Brain – Facts and Figures



1.2 Liters Brain Volume
0.25 m^2 Unfolded Cortex Area
$40 \cdot 10^9$ Cortex Neurons
$2.4 \cdot 10^{14}$ Cortex Synapses
~ 20 Watts

Volume Breakdown:

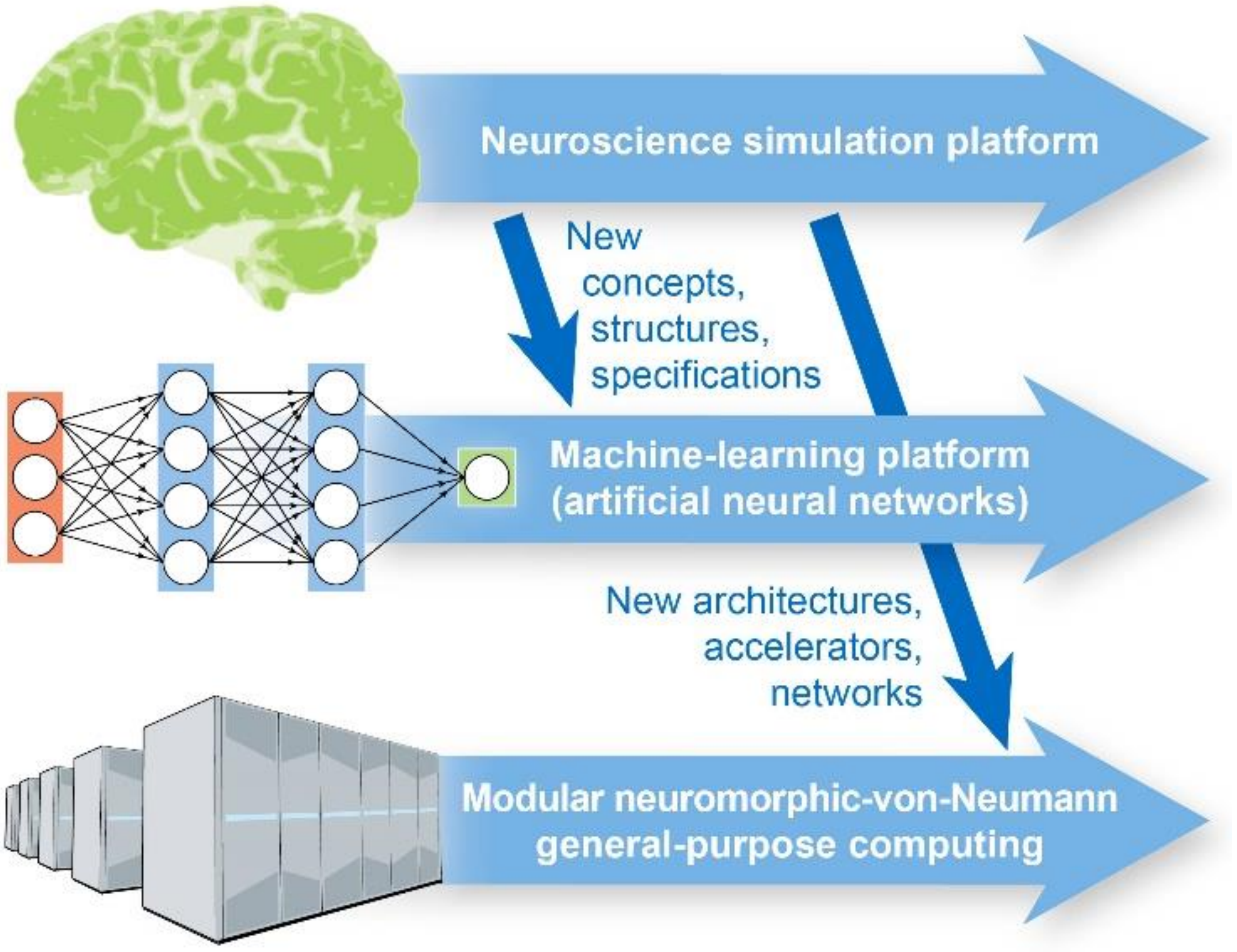


Fan In:
$10^9/10^5 = 10^4$ Synapses / Neuron

Fan Out:
10^4 Projection Targets / Neuron

Local Connectivity:
$10^4/10^5 = 0.1$; i.e. each neuron projects to / receives from 10% of all others

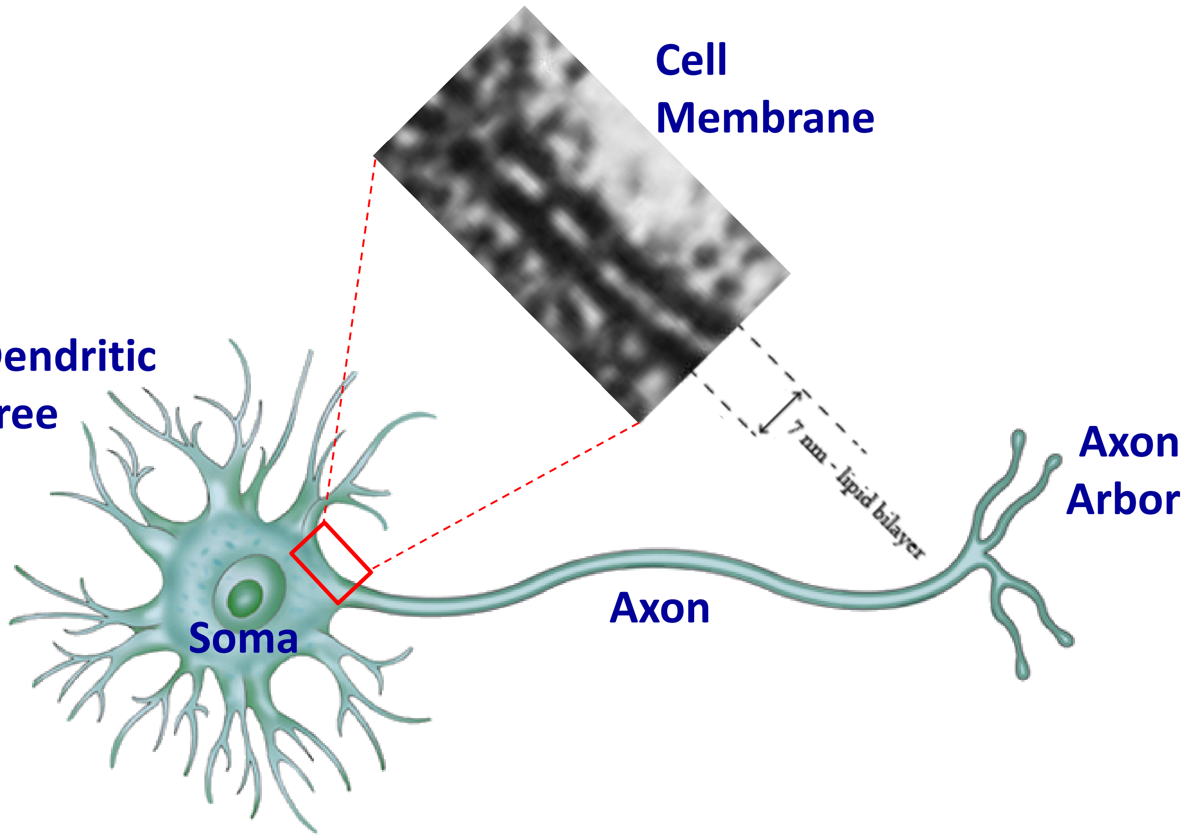
Long-Term Goals



Outline

- Introduction
- **Neuronal Network Components and Models**
- Exemplary Large-Scale Networks
- Future Requirements
- Simulation Principles
  - Computation: Numerical ODE Solvers
  - Communication: AER
- State-of-the Art
- Brick Walls
  - Computation: …
  - Communication: …
- Conclusion

Neuron

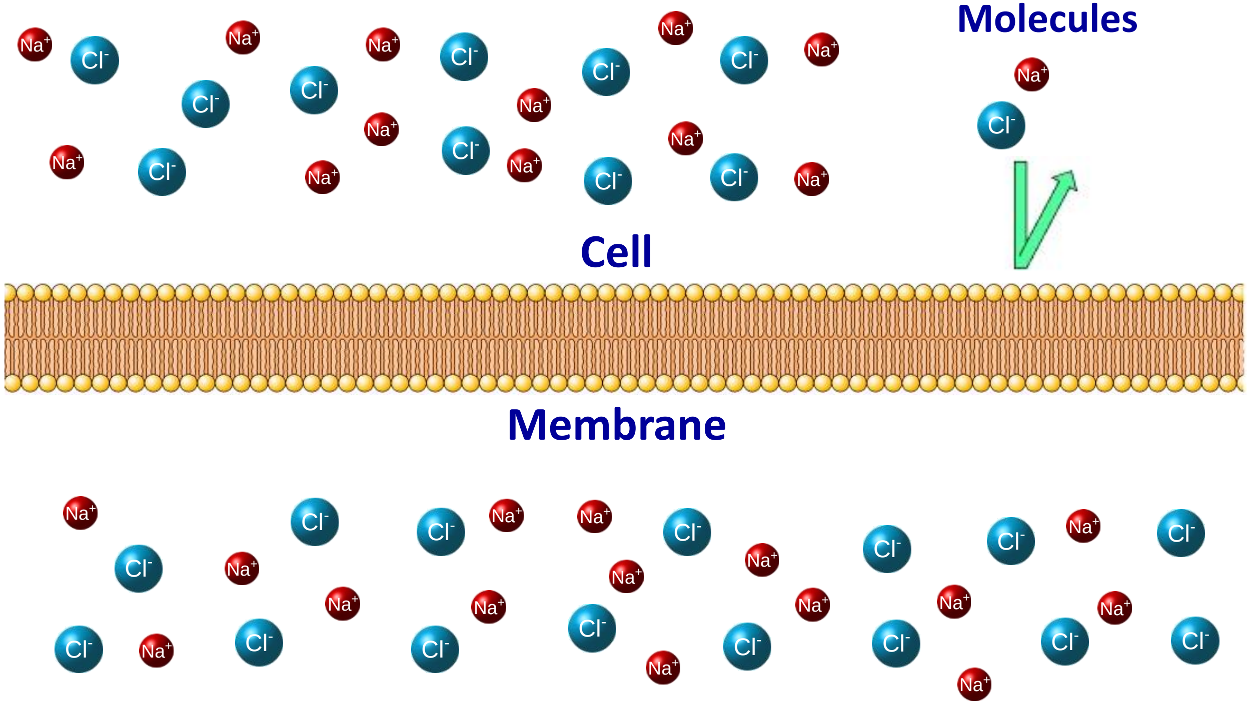

Cell
Membrane
7 nm - lipid bilayer
Dendritic
Tree
Soma
Axon
Axon
Arbor

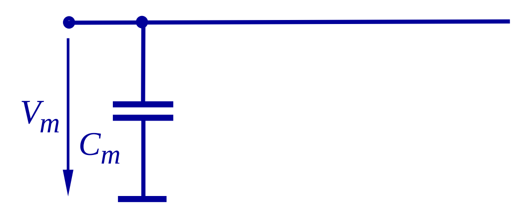
Cortical Spiking Neuron and Action Potentials

Very much simplified

Extracellular Fluid = Reference Potential

Any Charged Molecules

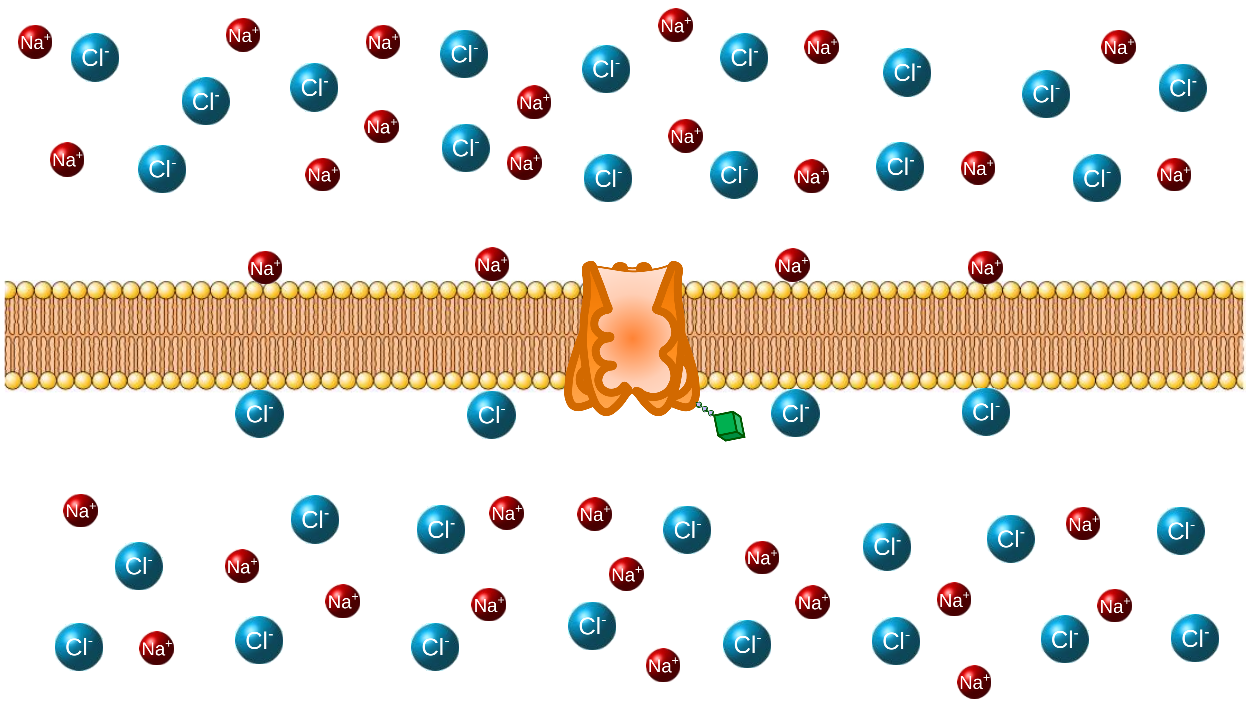


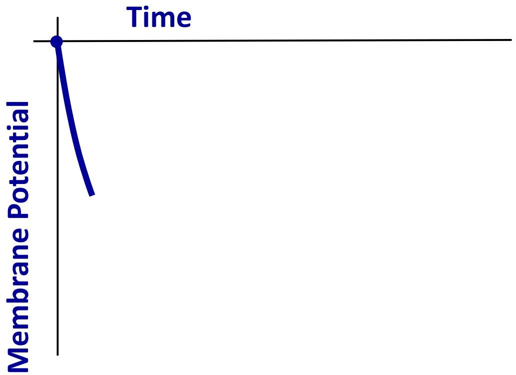


Intracellular Cytosol

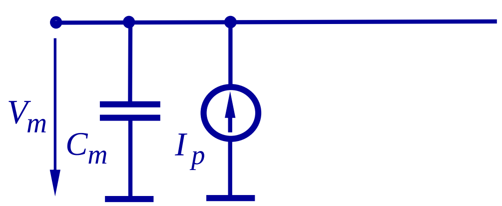
Cortical Spiking Neuron and Action Potentials

Extracellular Fluid = Reference Potential

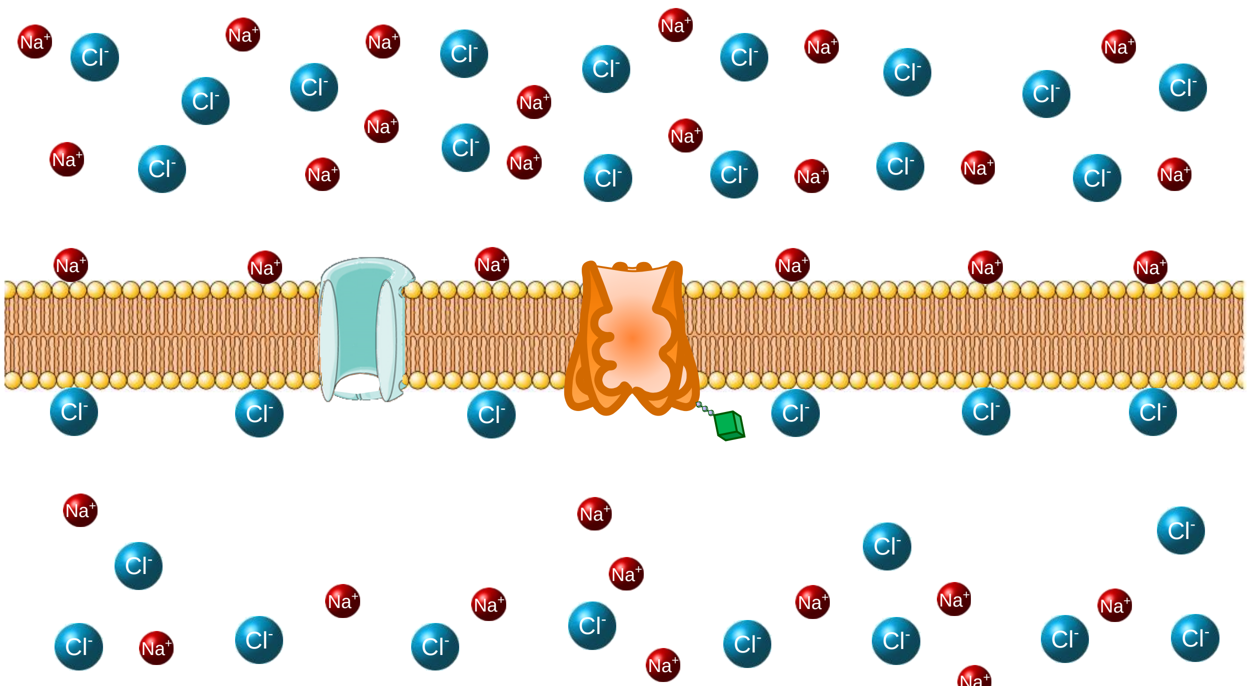


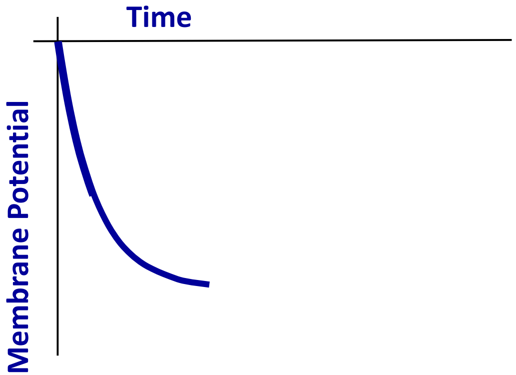


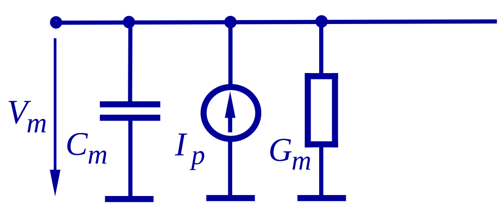
Intracellular Cytosol



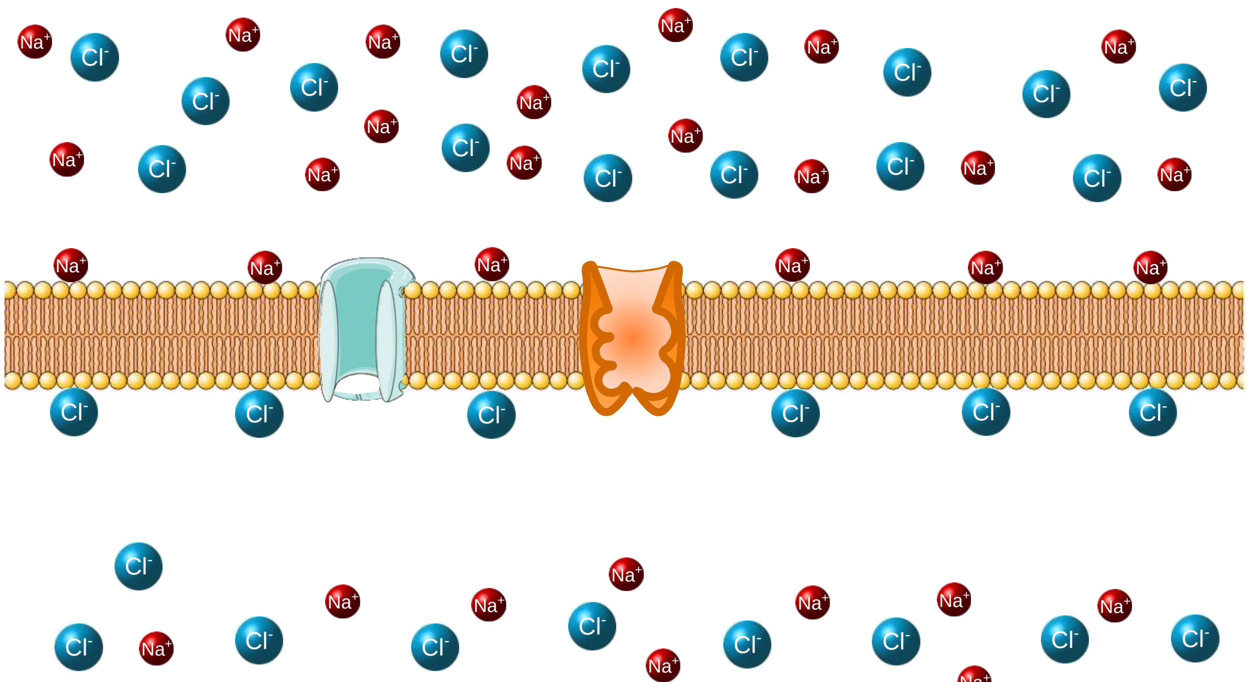
Cortical Spiking Neuron and Action Potentials

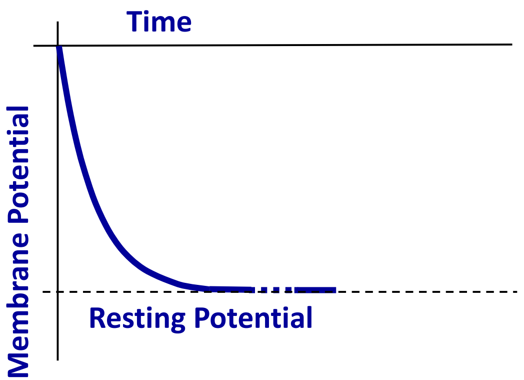


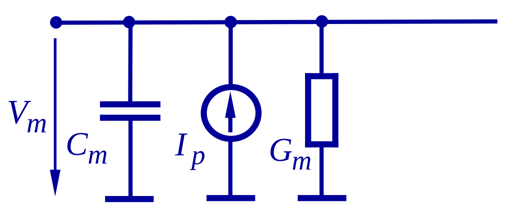




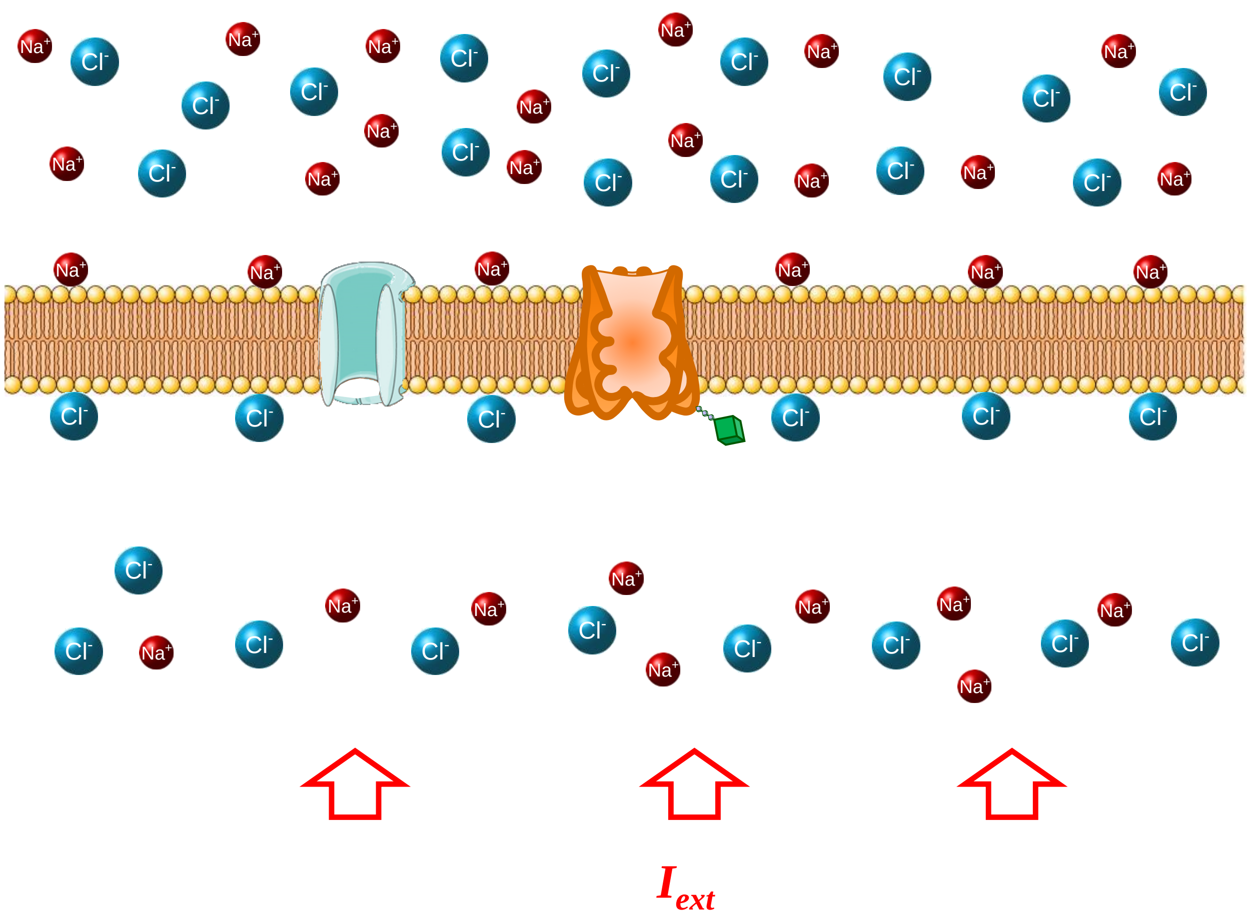
Cortical Spiking Neuron and Action Potentials

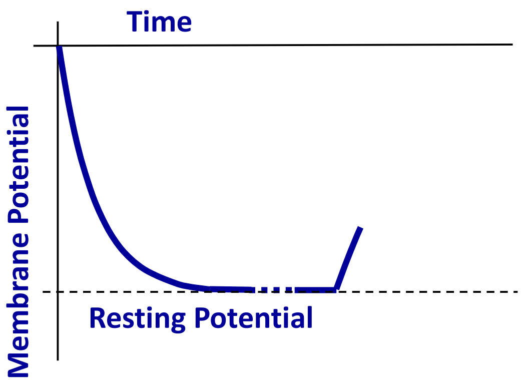


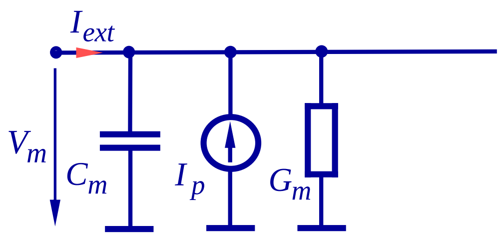




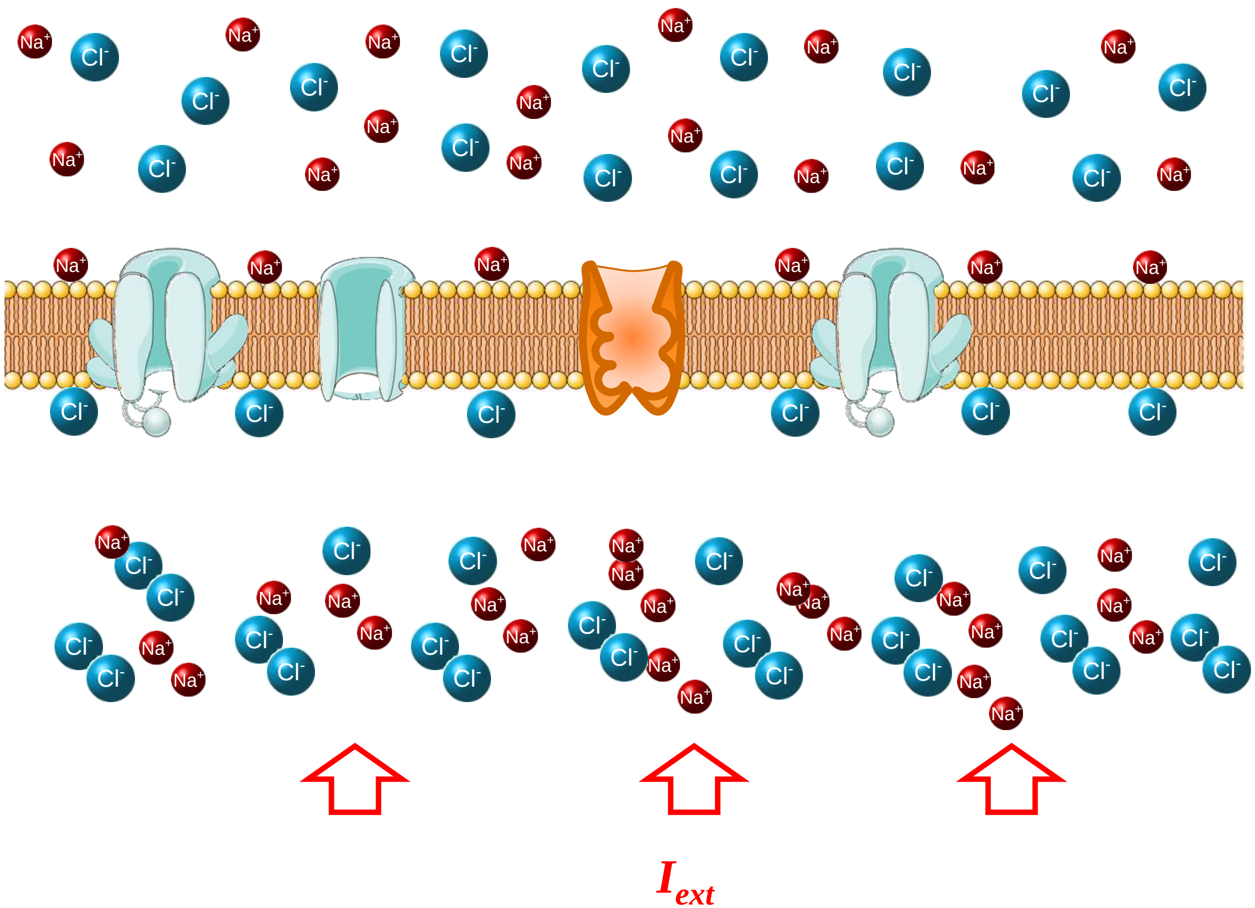
Cortical Spiking Neuron and Action Potentials

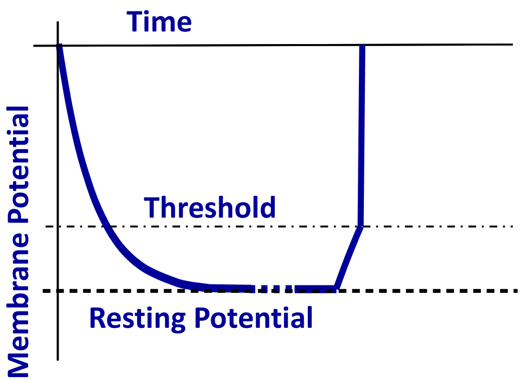


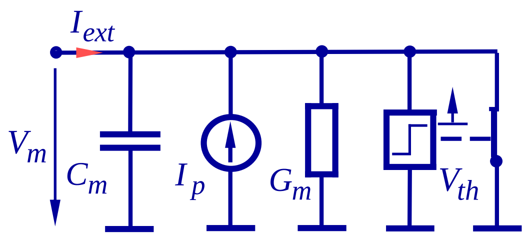




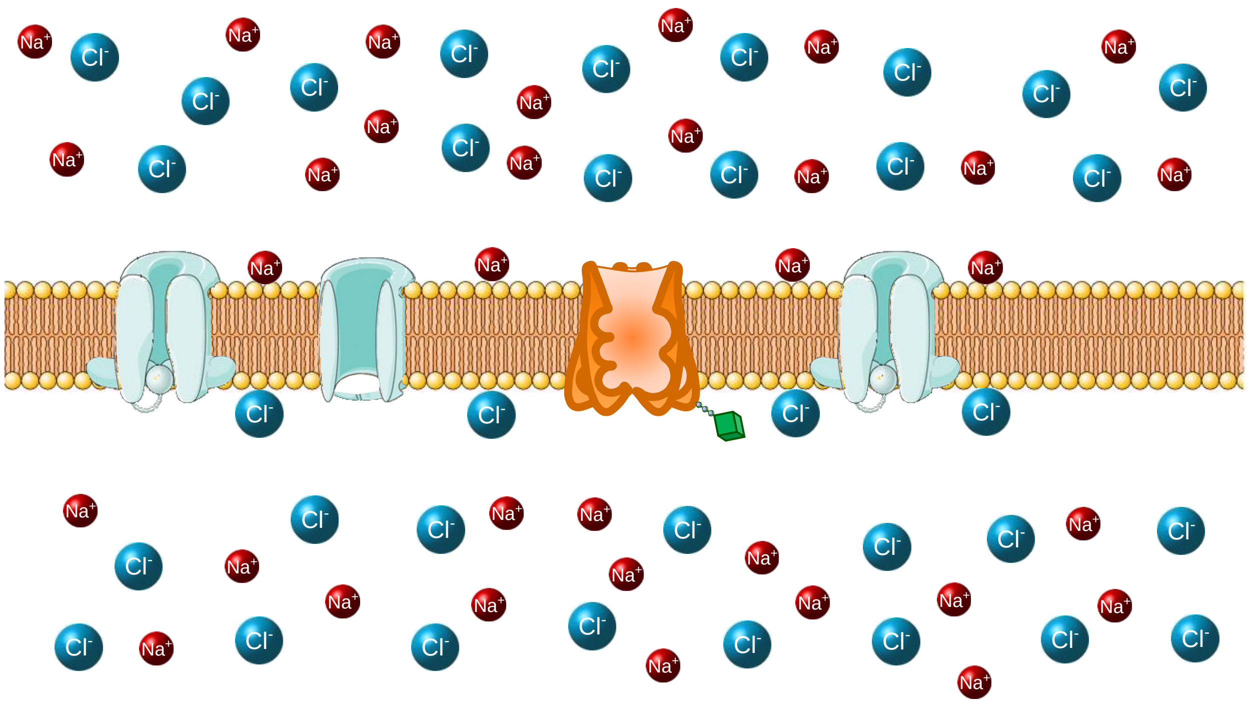
Cortical Spiking Neuron and Action Potentials

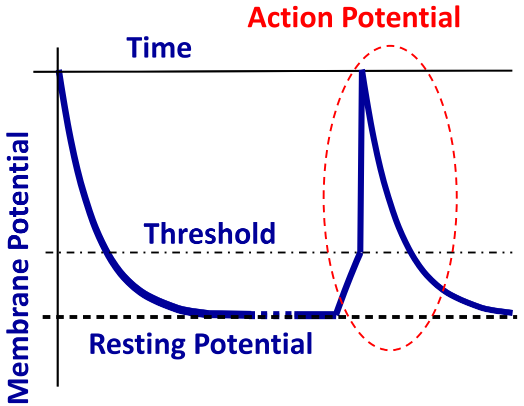


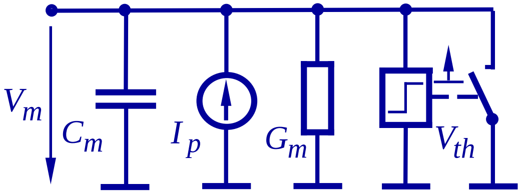




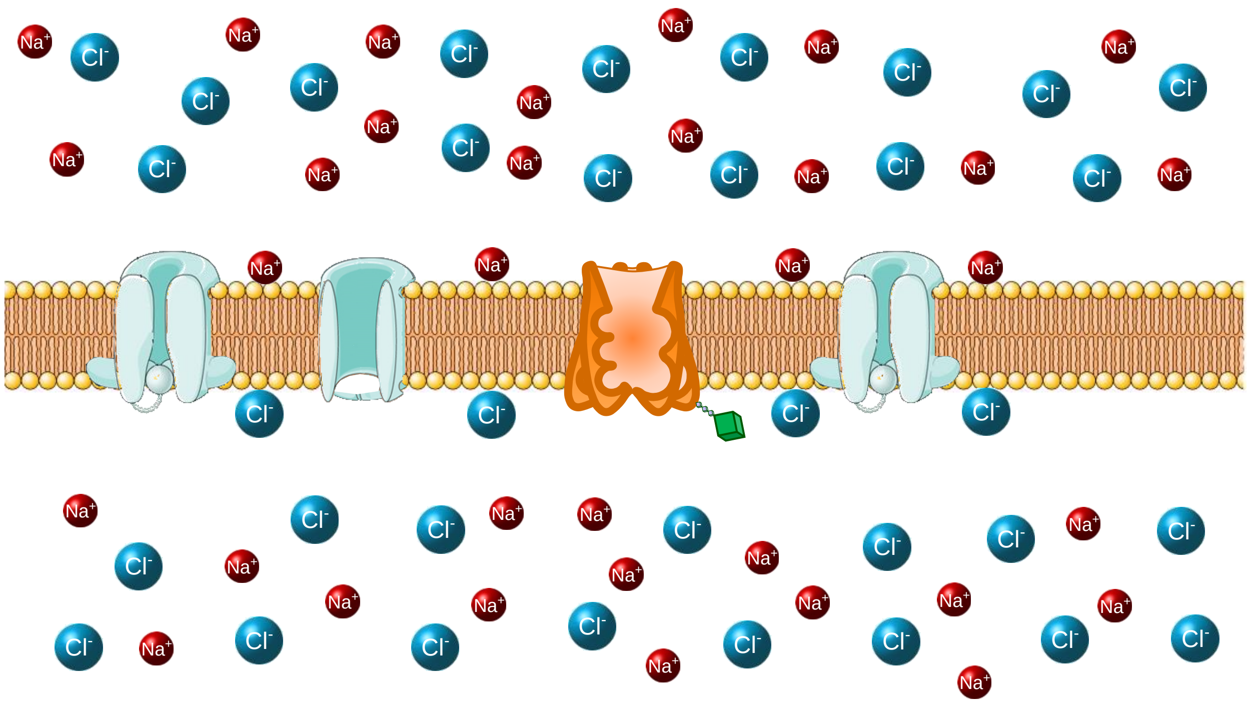
Cortical Spiking Neuron and Action Potentials

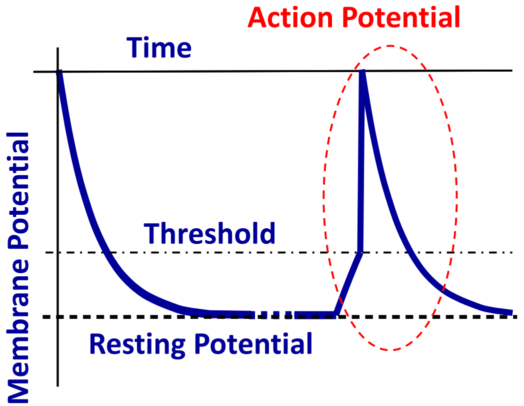


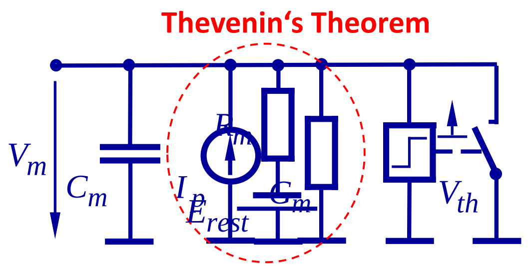




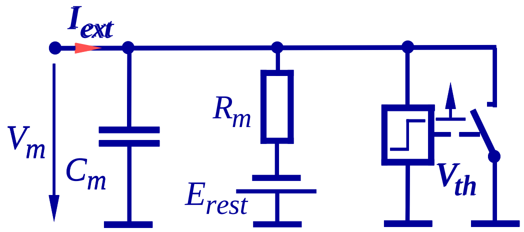
Cortical Spiking Neuron and Action Potentials







Lapique's Leaky Integrate-and-Fire Model (1907)



- **Kirchhoff Nodal Analysis**

$$C_m \cdot \frac{\mathrm{d}V_m(t)}{\mathrm{d}t} = -\frac{V_m(t) - E_{rest}}{R_m} + I_{ext}(t)$$

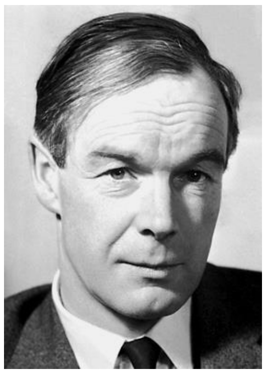
$$\frac{\mathrm{d}V_m(t)}{\mathrm{d}t} = \underbrace{\frac{1}{C_m \cdot R_m}}_{\tau_m} \cdot \left[E_{rest} - V_m(t)\right] + \frac{1}{R_m} \cdot I_{ext}(t)$$

$$\textit{iff} \quad V_m(t) \geq V_{th}: \quad \text{Spike and } V_m(t) = E_{rest}$$

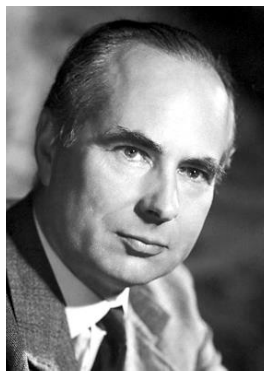
Purely phenomenological model

Point Neuron Models: The Hodgkin-Huxley Model (1952)

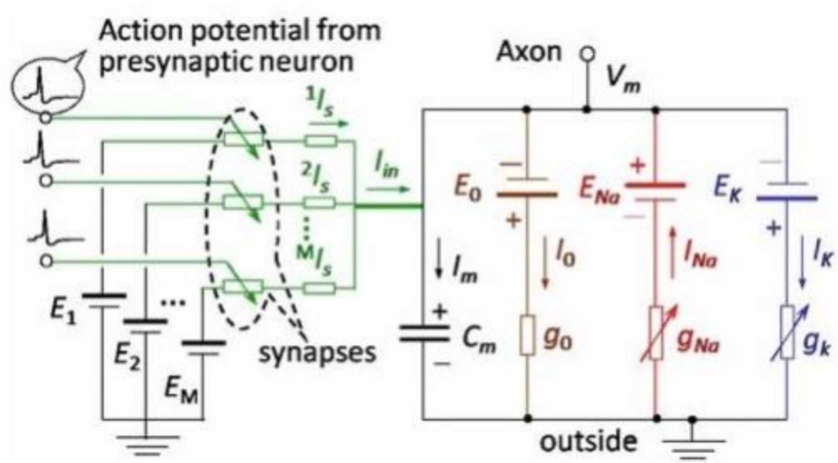
The Nobel Prize in Physiology or Medicine 1963



Allan Lloyd Hodgkin



Andrew Fielding Huxley



$$C \cdot \frac{\mathrm{d}}{\mathrm{d}\,t} V(t) = -\sum_k I_k(t) + I(t)$$

$$\sum_k I_k(t) = g_{Na} \cdot m^3 \cdot h \cdot \left[V(t) - E_{Na}\right] + g_K \cdot n^4 \cdot \left[V(t) - E_K\right] + g_L \cdot \left[V(t) - E_L\right]$$

$$\frac{\mathrm{d}}{\mathrm{d}\,t} m(t) = \alpha_m(V) \cdot \left[1 - m(t)\right] - \beta_m(V) \cdot m(t)$$

$$\frac{\mathrm{d}}{\mathrm{d}\,t} n(t) = \alpha_n(V) \cdot \left[1 - n(t)\right] - \beta_n(V) \cdot n(t)$$

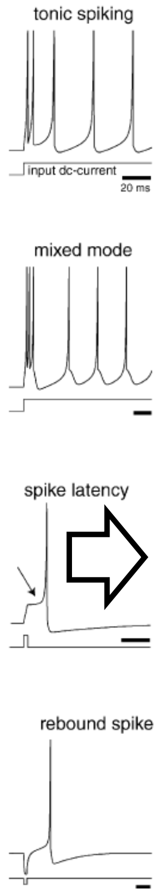
$$\frac{\mathrm{d}}{\mathrm{d}\,t} h(t) = \alpha_h(V) \cdot \left[1 - h(t)\right] - \beta_h(V) \cdot h(t)$$

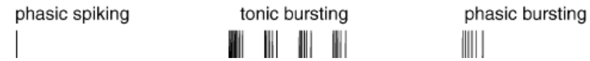
$$\alpha_m(V) = \frac{2.5 - 0.1 \cdot V(t)/\mathrm{mV}}{\left[\exp\left(2.5 - 0.1 \cdot V(t)/\mathrm{mV}\right) - 1\right]}$$

$$\beta_m(V) = 4 \cdot \exp\left(-V(t)/18\,\mathrm{mV}\right)$$

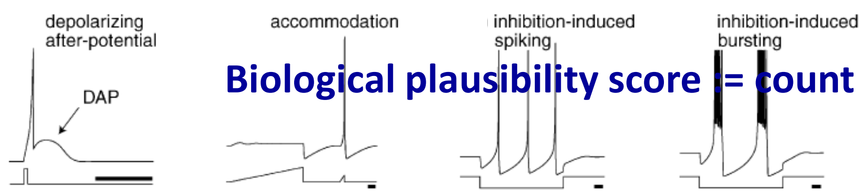
Derived from experiments with the giant squid axon

Benchmarking Point Neuron Models [Izhikevich, 2004]



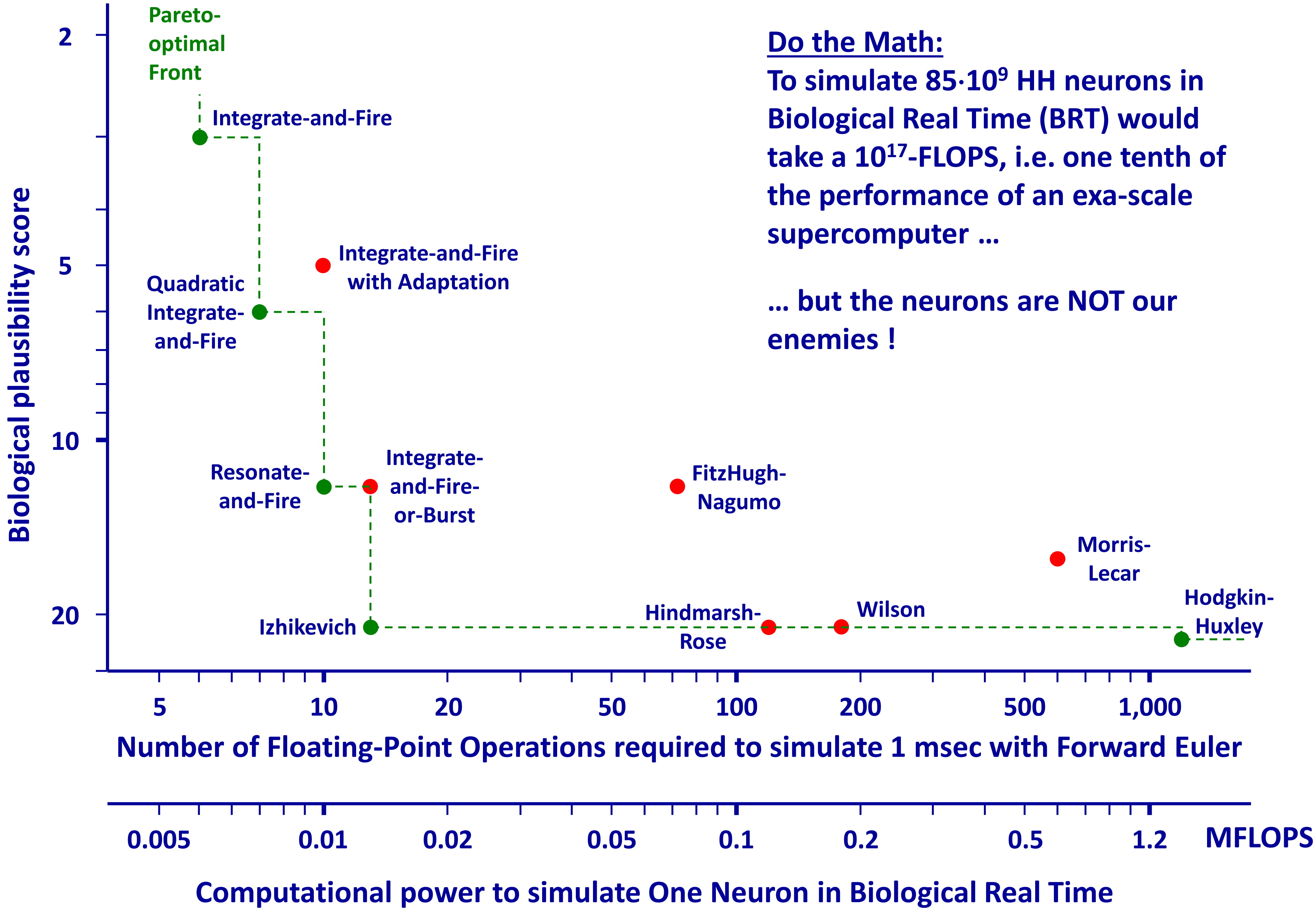


Models	biophysically meaningful	tonic spiking	phasic spiking	tonic bursting	phasic bursting	mixed mode	spike frequency adaptation	class 1 excitable	class 2 excitable	spike latency	subthreshold oscillations	resonator	integrator	rebound spike	rebound burst	threshold variability	bistability	DAP	accommodation	inhibition-induced spiking	inhibition-induced bursting	chaos	# of FLOPS
integrate-and-fire	-	+	-	-	-	-	-	+	-	-	-	-	+	-	-	-	-	-	-	-	-	-	5
integrate-and-fire with adapt.	-	+	-	-	-	-	+	+	-	-	-	-	+	-	-	-	-	+	-	-	-	-	10
integrate-and-fire-or-burst	-	+	+		+	-	+	+	-	-	-	-	+	+	+	-	+	+	-	-	-		13
resonate-and-fire	-	+	+	-	-	-	-	+	+	-	+	+	+	+	-	-	+	+	+	-	-	+	10
quadratic integrate-and-fire	-	+	-	-	-	-	-	+	-	+	-	-	+	-	-	+	+	-	-	-	-	-	7
Izhikevich (2003)	-	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	13
FitzHugh-Nagumo	-	+	+	-		-	-	+	-	+	+	+	-	+	-	+	+	-	+	+	-	-	72
Hindmarsh-Rose	-	+	+	+			+	+	+	+	+	+	+	+	+	+	+	+	+	+		+	120
Morris-Lecar	+	+	+	-		-	-	+	+	+	+	+	+	+		+	+	-	+	+	-	-	600
Wilson	-	+	+	+			+	+	+	+	+	+	+	+	+	+		+	+				180
Hodgkin-Huxley	+	+	+	+			+	+	+	+	+	+	+	+	+	+	+	+	+	+		+	1200



Biological plausibility score := count of non-negative entries

Benchmarking Point Neuron Models [Izhikevich, 2004]



Do the Math:

To simulate $85 \cdot 10^9$ HH neurons in Biological Real Time (BRT) would take a 10^{17}-FLOPS, i.e. one tenth of the performance of an exa-scale supercomputer …

… but the neurons are NOT our enemies !

Today‘s Zoo of Point Neuron Models

Hodkin-and-Huxley [Hodkin, Huxley, 1952]

$$C\cdot\frac{\mathrm{d}}{\mathrm{d}t}V(t) = -\sum_k I_k(t) + I(t)$$

$$\sum_k I_k(t) = g_{Na}\cdot m^3\cdot h\cdot[V(t)-E_{Na}] + g_K\cdot n^4\cdot[V(t)-E_K] + g_L\cdot[V(t)-E_L]$$

$$\frac{\mathrm{d}}{\mathrm{d}t}m(t) = \alpha_m(V)\cdot[1-m(t)] - \beta_m(V)\cdot m(t)$$

$$\frac{\mathrm{d}}{\mathrm{d}t}n(t) = \alpha_n(V)\cdot[1-n(t)] - \beta_n(V)\cdot n(t)$$

$$\frac{\mathrm{d}}{\mathrm{d}t}h(t) = \alpha_h(V)\cdot[1-h(t)] - \beta_h(V)\cdot h(t)$$

$$\alpha_m(V) = \frac{2.5 - 0.1\cdot V(t)/\mathrm{mV}}{[\exp(2.5-0.1\cdot V(t)/\mathrm{mV})-1]}$$

$$\beta_m(V) = 4\cdot\exp(-V(t)/18\,\mathrm{mV})$$

IAF-or-Burst (IFB) [Smith et al., 2000]

$$C\cdot\frac{d}{dt}V(t) = g_L\cdot(V_L - V(t)) + g_T\cdot h\cdot\chi(V(t)-V_h) + I(t)$$

$$\frac{d}{dt}h(t) = \begin{cases} -\dfrac{h(t)}{\tau^-} & V \ge V_h \\ \dfrac{1-h(t)}{\tau^+} & V < V_h \end{cases}$$

$$V(t^+) = V_r \text{ if } V(t^-) = V_{th}$$

Izhikevich [Izhikevich, 2003]

$$\frac{d}{dt}V(t) = 0.04\cdot V^2(t) + 5\cdot V(t) + 140 - w(t) + I(t)$$

$$\frac{d}{dt}w(t) = a\cdot(b\cdot V(t) - w(t))$$

$$\text{if } V(t) \ge +30mV,\ \text{then} \begin{cases} V(t) \leftarrow V_{reset} \\ w \leftarrow w + d \end{cases}$$

Quadratic IAF [Latham et al., 2000]

$+I(t)$

$+I(t)$

Adaptive Exponential IAF (AdEx) [Brette, Gerstner, 2005]

$$C\cdot\frac{d}{dt}V(t) = g_L\cdot(V(t)-V_R) + g_L\cdot\Delta_T\cdot e^{\frac{V(t)-\vartheta_{rh}}{\Delta_T}} - w(t) + I(t)$$

$$\tau\cdot\frac{d}{dt}w(t) = a\cdot(V(t)-E_L) - w(t)$$

$$V(t^+) = V_r \text{ if } V(t^-) = V_{th}$$

FitzHugh-Nagumo [FitzHugh, 1961]

$$\frac{d}{dt}V(t) = V(t) - \frac{1}{3}\cdot V^3(t) - W(t) + I(t)$$

$$\tau\cdot\frac{d}{dt}W(t) = V(t) + a - b\cdot W(t)$$

Integrate-and-Fire with Adaptation [Hill, 1936]

$$C\cdot\frac{\mathrm{d}}{\mathrm{d}t}V(t) = I(t) + \#$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\#(t) = \#$$

$$V(t^+) = V_r \text{ if } V(t^-) = V_{th}$$

Morris-Lecar [Morris, Lecar, 1981]

$$C\cdot\frac{\mathrm{d}}{\mathrm{d}t}V(t) = -\sum_k I_k(t) + I(t) \qquad \tau(V(t)) = \cosh\left(\frac{V(t)-E_3}{E_{3n}}\right)^{-1}$$

$$\tau(V)\cdot\frac{d}{dt}w(t) = w_\infty(V) - w(t) \qquad m(V) = \frac{1}{2}\left(1+\tanh\left(\frac{V-E_1}{E_{1,n}}\right)\right)$$

$$\sum_k I_k(t) = g_1\cdot m(V)\cdot(V-V_0) + g_2\cdot w\cdot(V-V_2) + g_L\cdot(V-V_L) \qquad w_\infty(V) = \frac{1}{2}\left(1+\tanh\left(\frac{V-E_2}{E_{2,n}}\right)\right)$$

Multi-Time-Scale Adaptive Threshold (MAT) [Kobayashi et al., 2009]

$$\tau\cdot\frac{d}{dt}V(t) = -V(t) + R\cdot I(t)$$

$$\theta(t) = \omega + \sum_k H(t-t_k)$$

$$H(t) = \sum_{j=1}^{L}\alpha_j\cdot e^{-\frac{t}{\tau_j}}$$

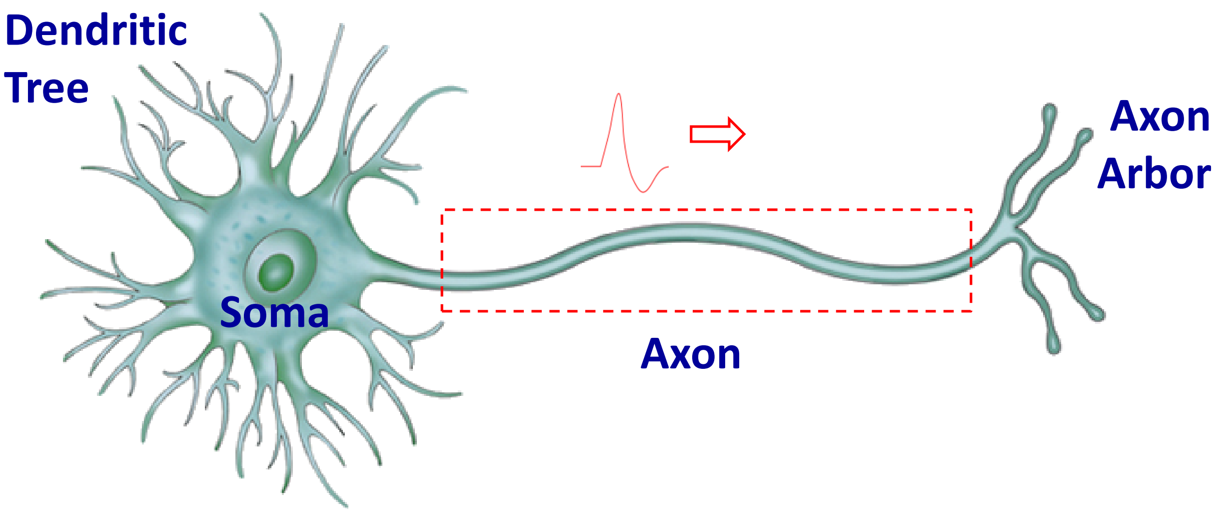
Integrate-and-Fire [Lapicque, 1907]

$$C\cdot\frac{\mathrm{d}}{\mathrm{d}t}V(t) = I(t)$$

$$V(t^+) = V_r \text{ if } V(t^-) = V_{th}$$

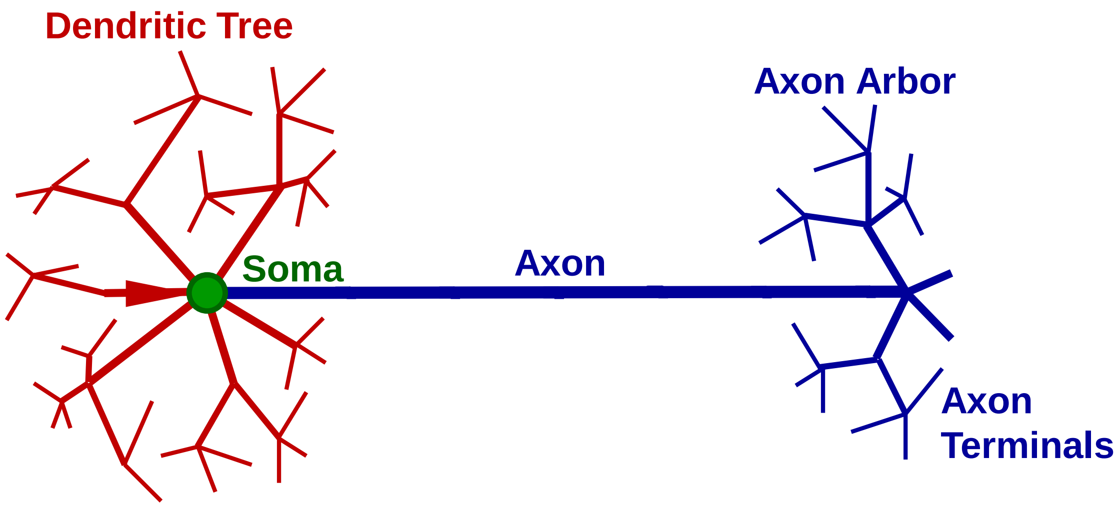
1900 1920 1940 1960 1980 2000 2020

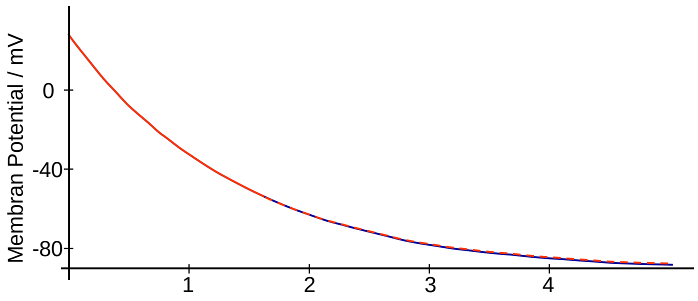
Neuron



Axons

Gray Matter: Non-myelinated Axons for short-range interconnect

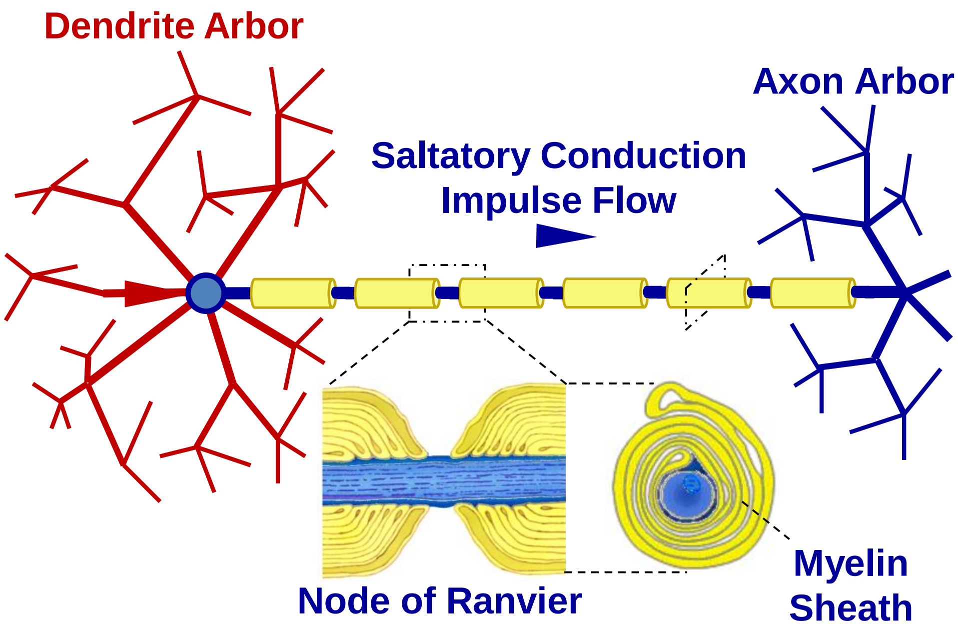


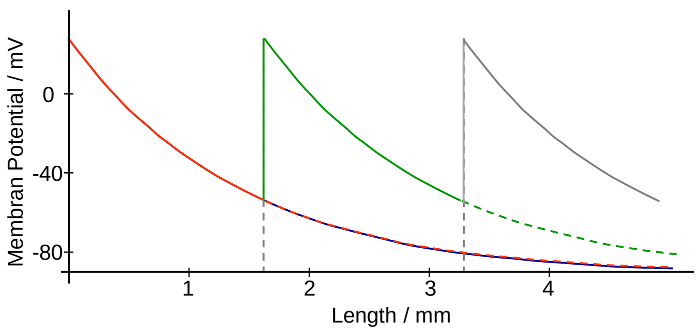


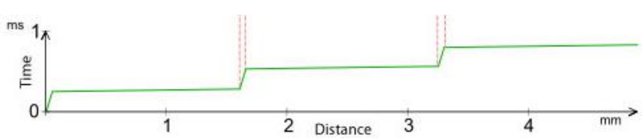
Axon model in large-scale simulation:
Axonal propagation delay t_{axon}

Axons

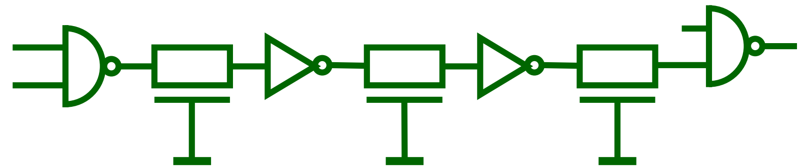
White Matter: Myelin-Coated Axons for long-range interconnect

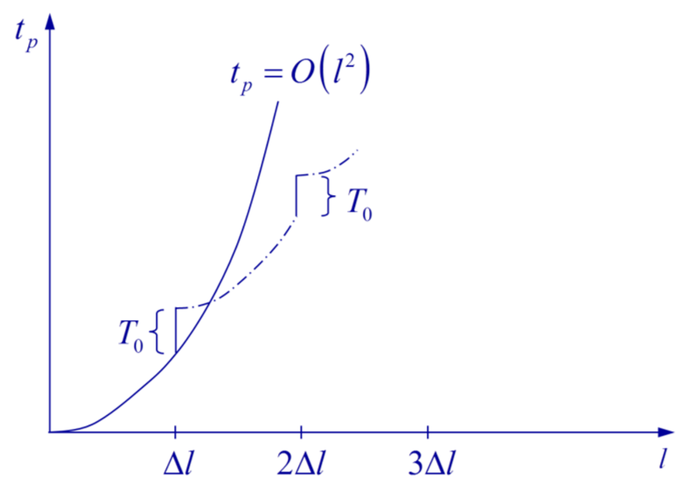




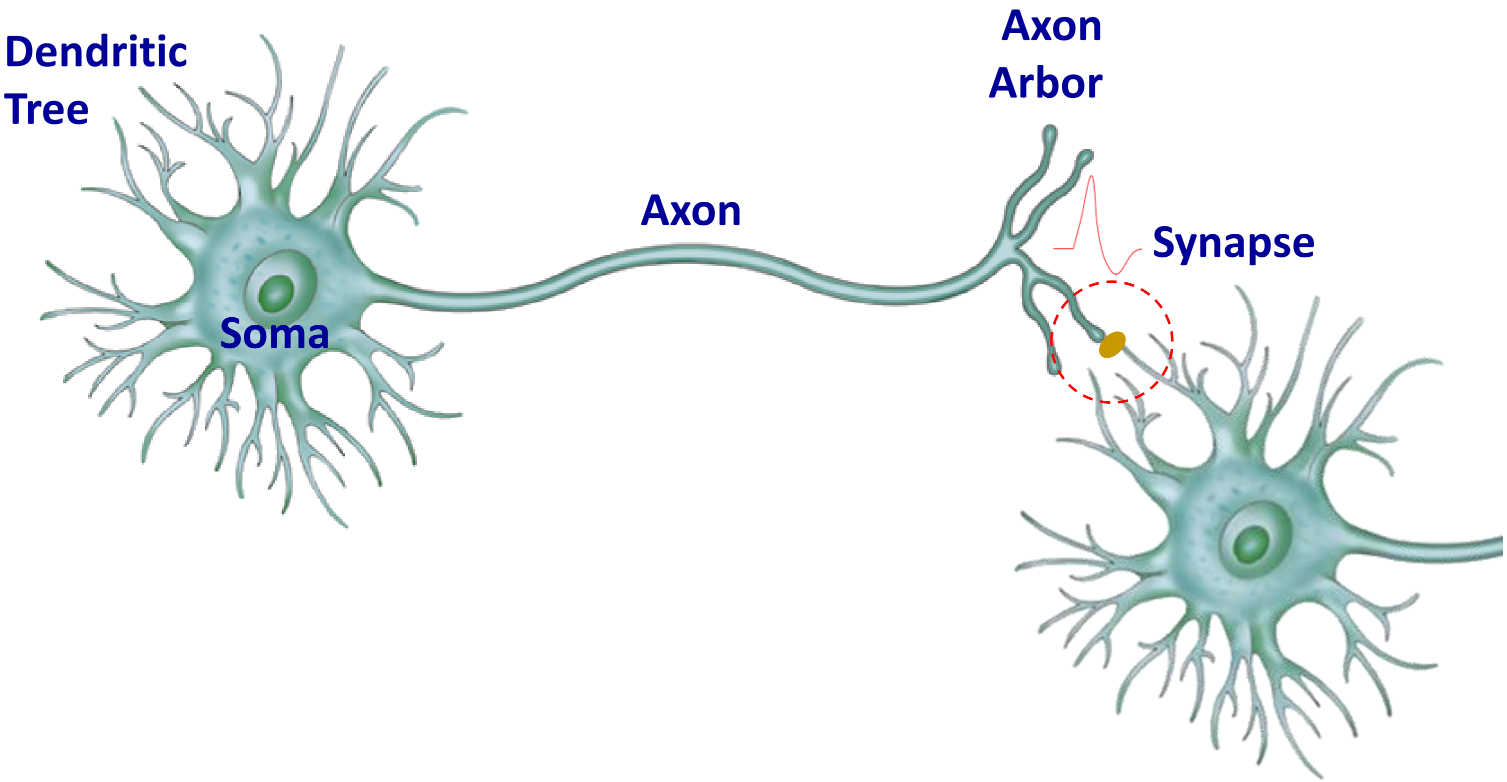


CMOS: Buffered RC-Interconnect





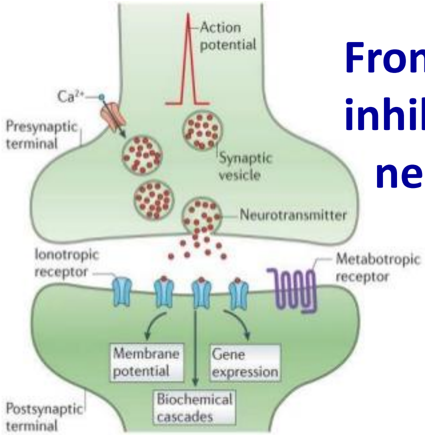
Neuron



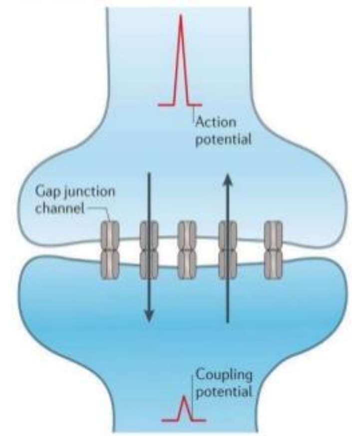
Chemical Synapses

Electrical Synapses

From excitatory neuron

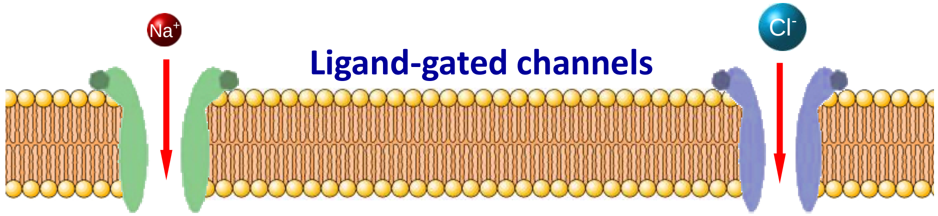


From inhibitory neuron



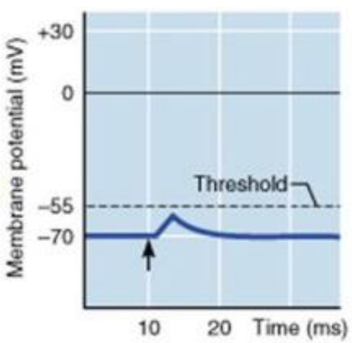
Excitatory

Inhibitory

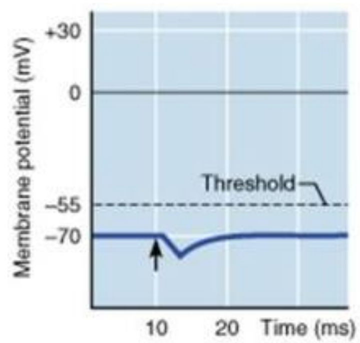


$I_s > 0$

$I_s < 0$



Excitatory postsynaptic potential (EPSP)



Inhibitory postsynaptic potential (IPSP)

Chemical Synapse Models

❒ Conductance-Based („COBA“) Models

Delta Function (Dirac pulse)

Impulse Response: $$g_{syn}(t) = \hat{g}_{syn} \cdot \delta(t - t_0)$$

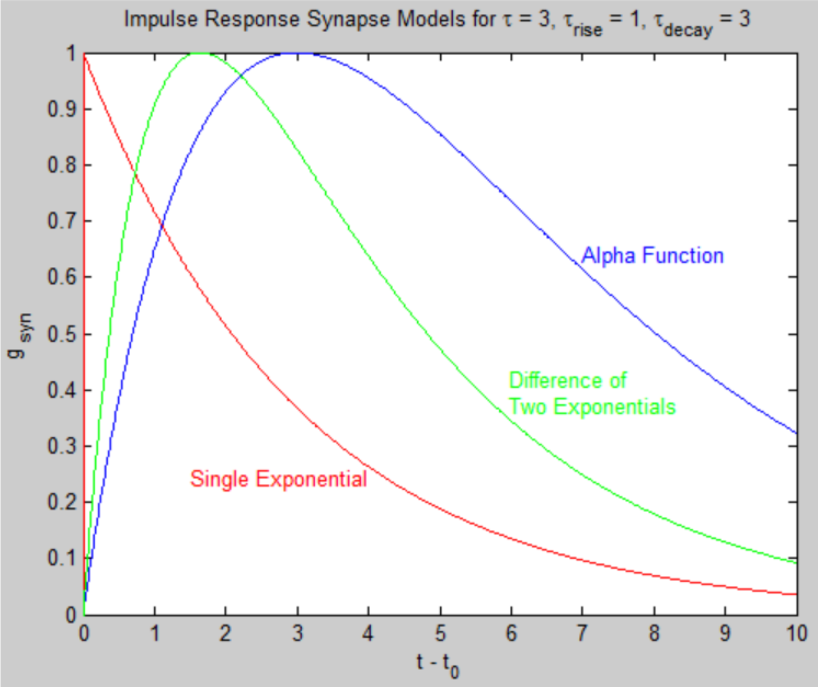
ODE: -

Single Exponential (Instantaneous rise)

Impulse Response: $$g_{syn}(t) = \hat{g}_{syn} \cdot \exp\left[-(t - t_0)/\tau_{syn}\right] \cdot \sigma(t - t_0)$$

ODE:

$$\tau_{syn} \cdot \frac{\mathrm{d}\, g_{syn}(t)}{\mathrm{d}t} = -g_{syn}(t) + \hat{g}_{syn} \cdot \delta(t - t_0)$$



Alpha Function

Impulse Response: $$g_{syn}(t) = \hat{g}_{syn} \cdot (t - t_0)/\tau_{syn} \cdot \exp\left[1 - (t - t_0)/\tau_{syn}\right] \cdot \sigma(t - t_0)$$

ODE:

$$\frac{\mathrm{d}\, g(t)}{\mathrm{d}t} = -\frac{g(t)}{\tau_{syn}} + h(t)$$

$$\frac{\mathrm{d}\, h(t)}{\mathrm{d}t} = -\frac{h(t)}{\tau_{syn}} + h_o(t) \cdot \delta(t - t_0)$$

Beta Function (Difference of two exponentials)

Impulse Response: $$g_{syn}(t) = \hat{g}_{syn} \cdot f \cdot \left\{\exp\left[-(t - t_0)/\tau_{decay}\right] - \exp\left[-(t - t_0)/\tau_{rise}\right]\right\} \cdot \sigma(t - t_0)$$

ODEs:

$$\frac{\mathrm{d}\, g(t)}{\mathrm{d}t} = -\frac{g(t)}{\tau_{decay}} + h(t)$$

$$\frac{\mathrm{d}\, h(t)}{\mathrm{d}t} = -\frac{h(t)}{\tau_{rise}} + h_o(t) \cdot \delta(t - t_0)$$

[A. Roth, M.C.W. van Rossum]

Chemical Synapse Models

❐ Conductance-Based („COBA“) Single Exponential Synapse to LIF Neuron

$$C_m \cdot \frac{\mathrm{d}V_m(t)}{\mathrm{d}t} = -V_m(t)/R_m + g_{syn}(t) \cdot \left[E_{syn} - V_m(t)\right]$$

$$\tau_{syn} \cdot \frac{\mathrm{d}\,g_{syn}(t)}{\mathrm{d}t} = -g_{syn}(t) + \hat{g}_{syn} \cdot \delta(t - t_0)$$

- ❍ **Bad news: Non-linear ODE system**
- ❍ **Good news: One-way coupled ODE system**

❐ Current-Based („CUBA“) Single Exponential Synapse to LIF Neuron

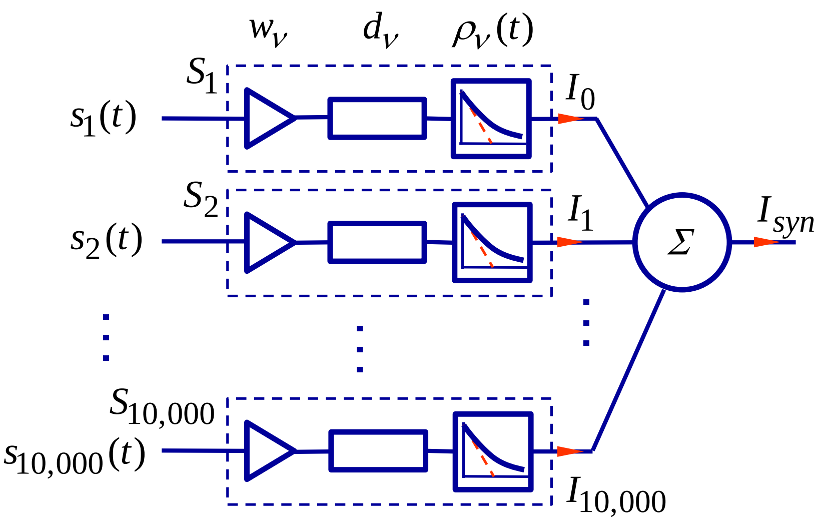
Approx.: $V_m(t)$ = const. in $\left[E_{syn} - V_m(t)\right] =: \tilde{E}_{syn}$

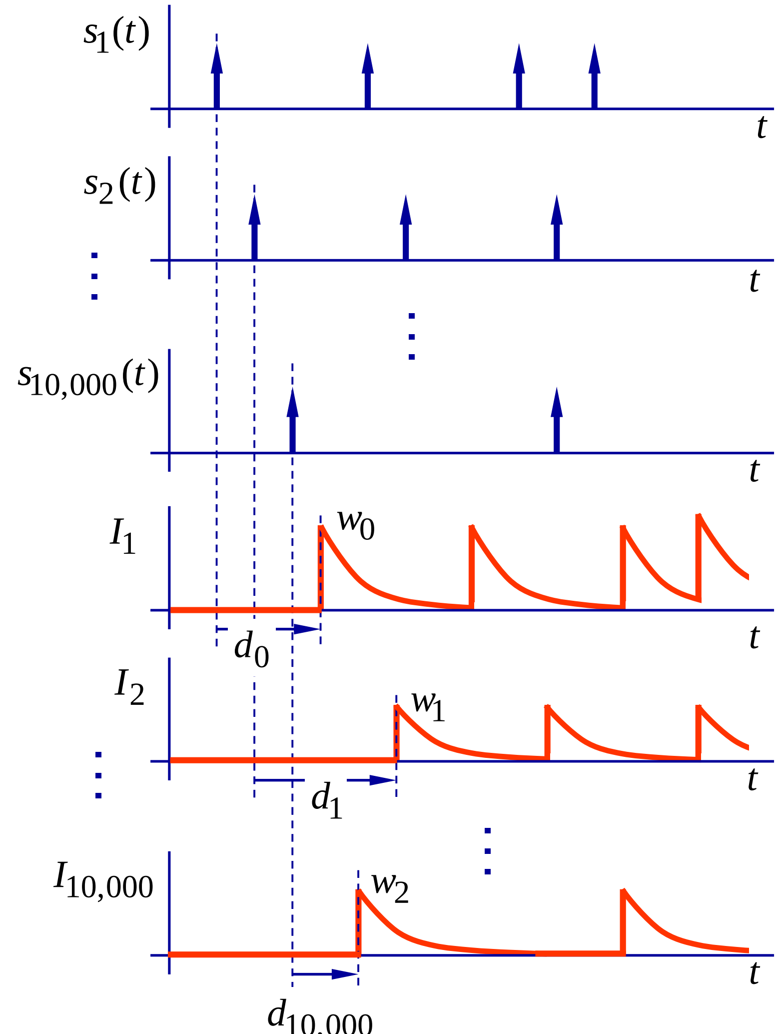
$$C_m \cdot \frac{\mathrm{d}V_m(t)}{\mathrm{d}t} = -V_m(t)/R_m + g_{syn}(t) \cdot \tilde{E}_{syn}$$

$$\tau_{syn} \cdot \frac{\mathrm{d}\,g_{syn}(t)}{\mathrm{d}t} = -g_{syn}(t) + \hat{g}_{syn} \cdot \delta(t - t_0)$$

- ❍ **Good news: Linear, one-way coupled ODE system**

Synapse Processing

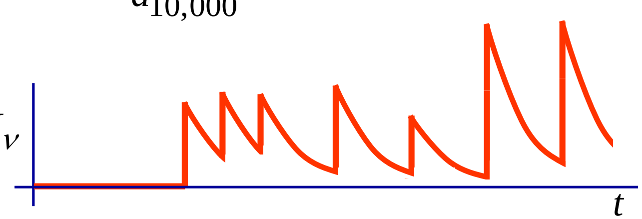




- **Giant effort to process 10,000 S/N**

Total synaptic current

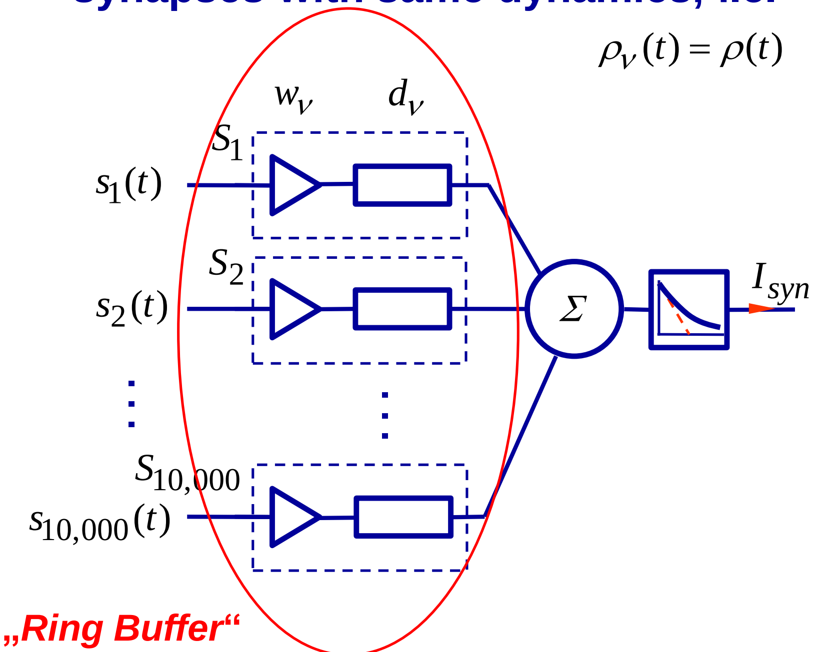
$$I_{syn} = \sum_{\nu=1}^{10,000} I_\nu$$

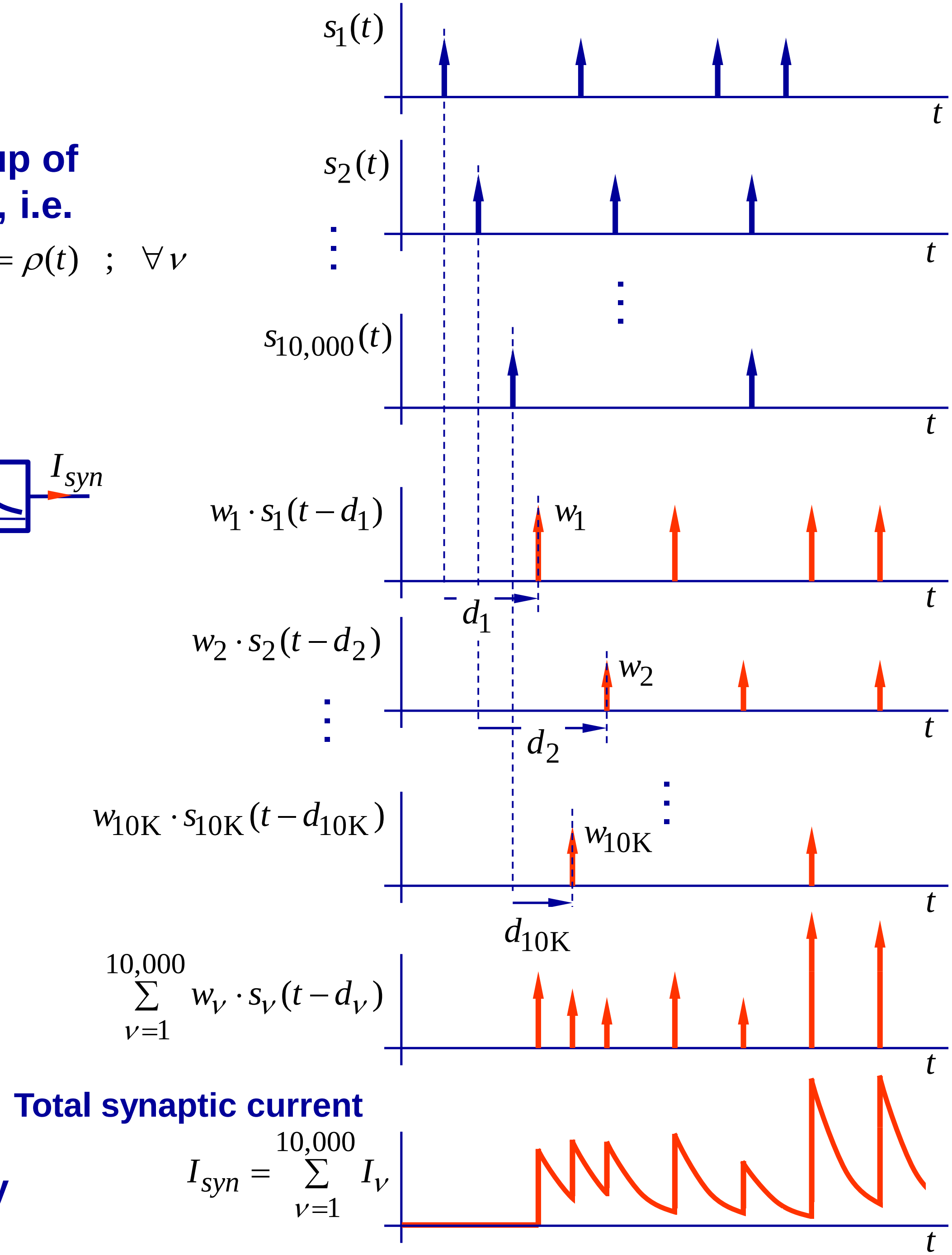


Synapse Processing

- One „lumped synapse“ for group of synapses with same dynamics, i.e.

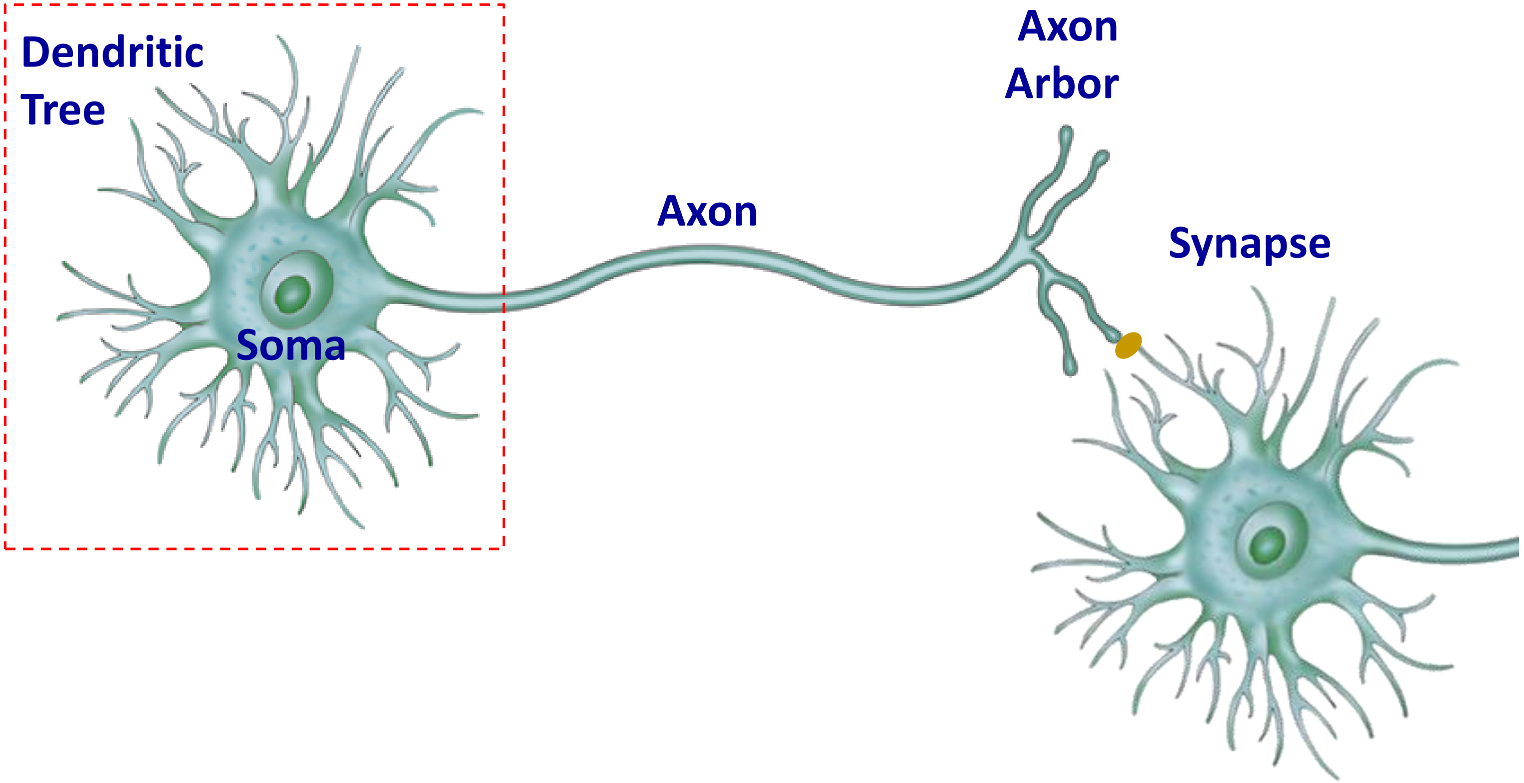
$$\rho_\nu(t) = \rho(t) \quad ; \quad \forall \nu$$





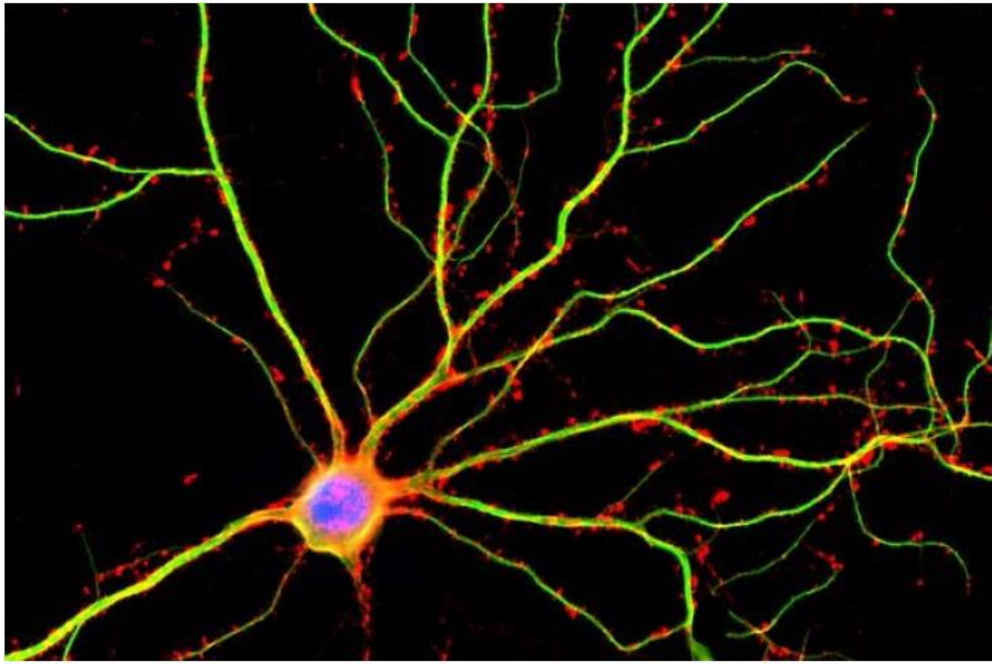
- Works for current- and conductance-based synapses
- At least two lumped synapses, one excitatory and one inhibitory

Neuron



Compartmental Dendrite Models

Brain is 10 times more active than previously measured, UCLA researchers find



Shelley Halpain/UC San Diego

Science RESEARCH ARTICLES

Cite as: J. J. Moore *et al.*, *Science* 10.1126/science.aaj1497 (2017).

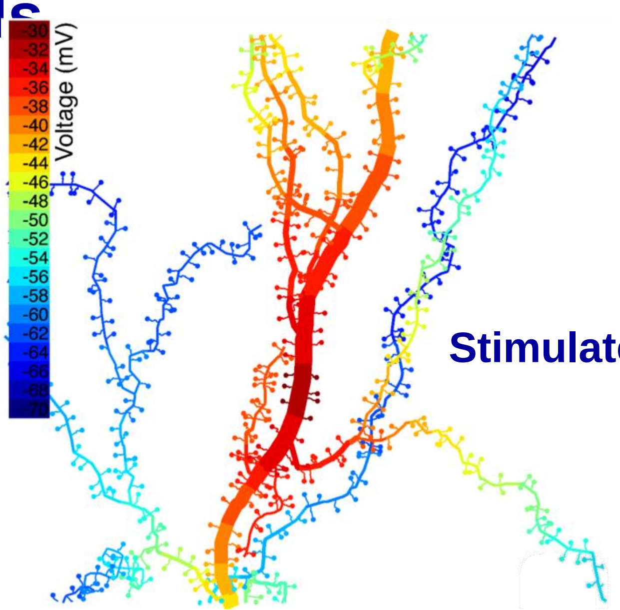
Dynamics of cortical dendritic membrane potential and spikes in freely behaving rats

Jason J. Moore,[1,2]* Pascal M. Ravassard,[1,3] David Ho,[1,2] Lavanya Acharya,[1,4] Ashley L. Kees,[1,2] Cliff Vuong,[1,3] Mayank R. Mehta[1,2,3,5]*

[1]W. M. Keck Center for Neurophysics, Integrative Center for Learning and Memory, and Brain Research Institute, University of California at Los Angeles, Los Angeles, CA, USA. [2]Neuroscience Interdepartmental Program, University of California at Los Angeles, Los Angeles, CA, USA. [3]Department of Physics and Astronomy, University of California at Los Angeles, Los Angeles, CA, USA. [4]Biomedical Engineering Interdepartmental Program, University of California at Los Angeles, Los Angeles, CA, USA. [5]Departments of Neurology and Neurobiology, University of California at Los Angeles, Los Angeles, CA, USA.

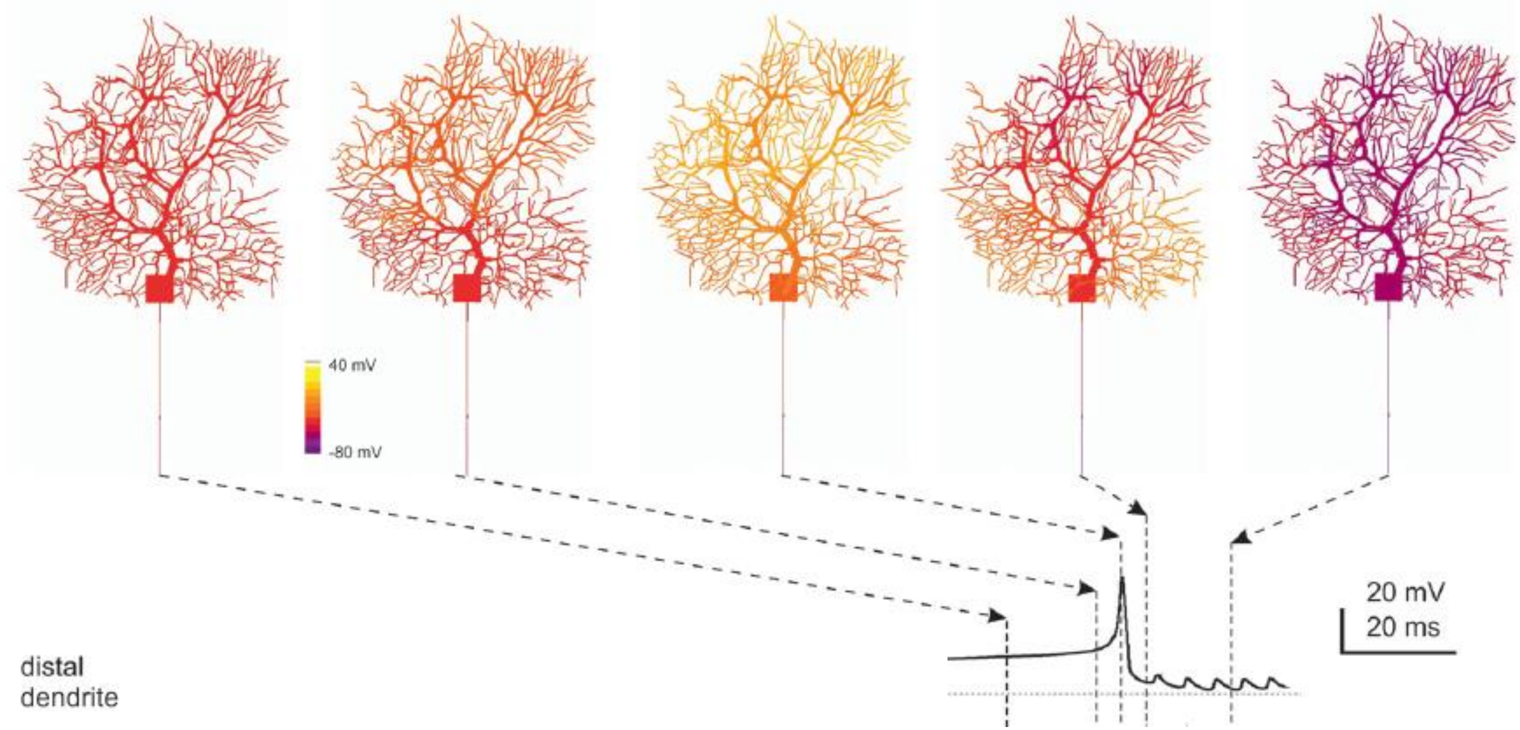
Compartmental Dendrite Models

Examples Potential Distribution:



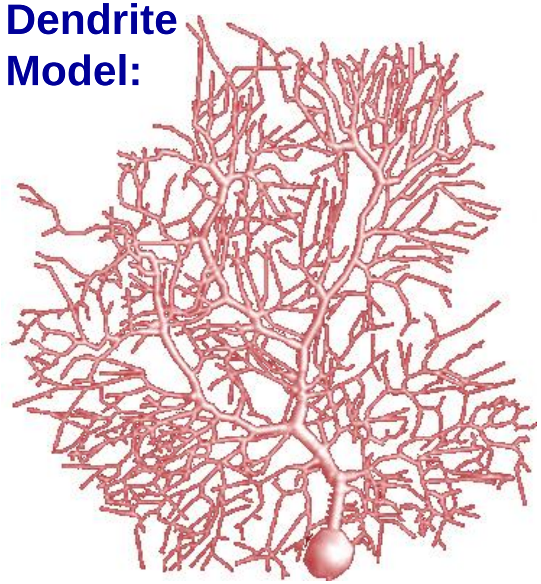
Stimulated synapse

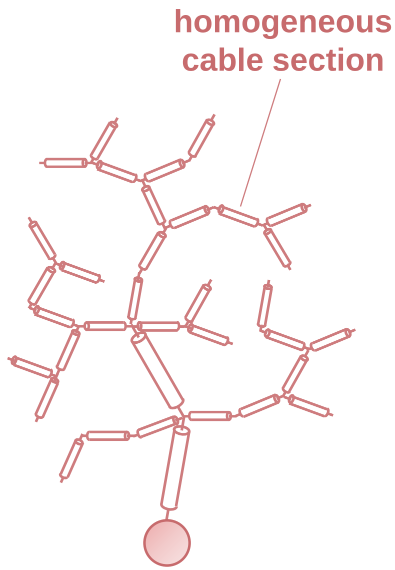
Firing neuron

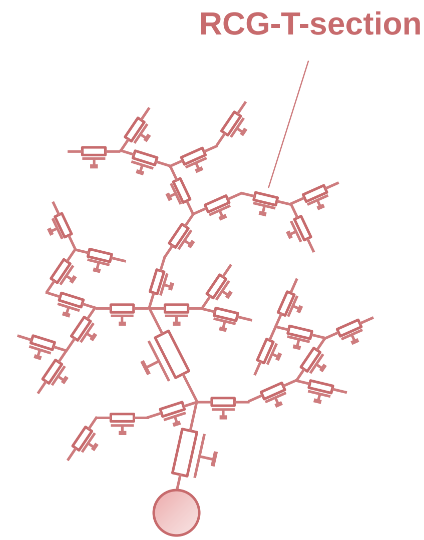


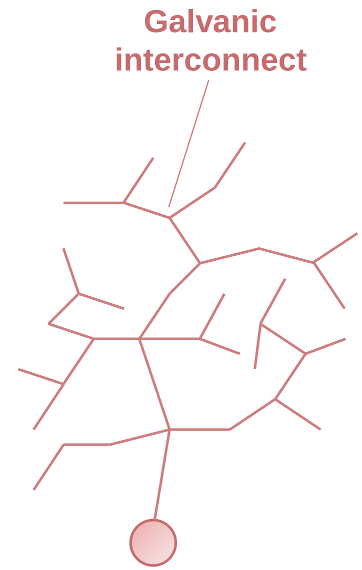
Compartmental Dendrite Models

Dendrite Model:









Arbitrary conductance and capacitance distribution in space

Set of cylindrical membrane cables

Set of discrete isopotential RCG compartments

Equipotential node

Mathematical Model:

System of PDEs

$$\frac{1}{2\pi a}\frac{\partial}{\partial x}\left(\frac{\pi a^2}{R_a}\frac{\partial V}{\partial x}\right) = C_m \frac{\partial V}{\partial t} + I_{\mathrm{HH}}$$

System of ODEs + cable equations (solutions of Telegraph‘s equation)

System of ODEs

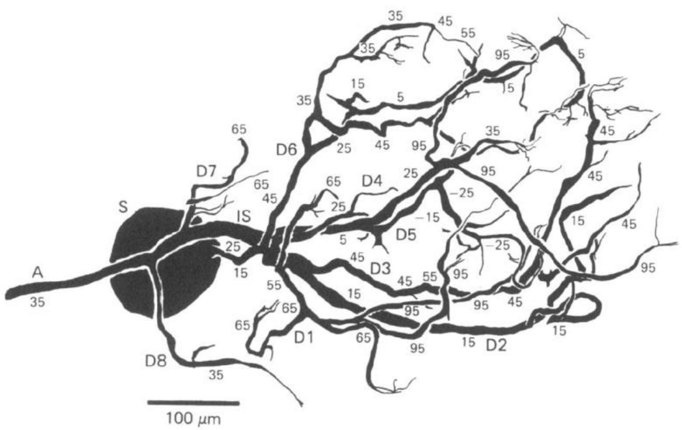
Point neuron ODE(s), only

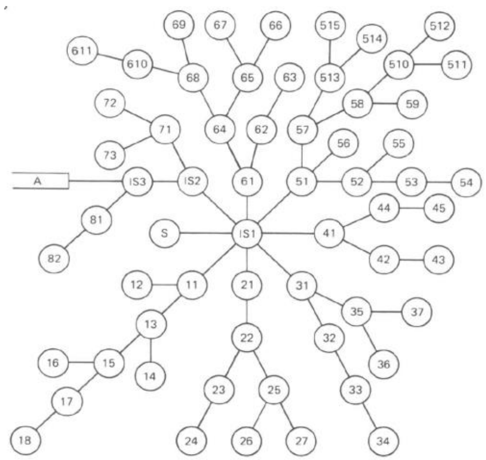
Compartmental Dendrite Models: Examples

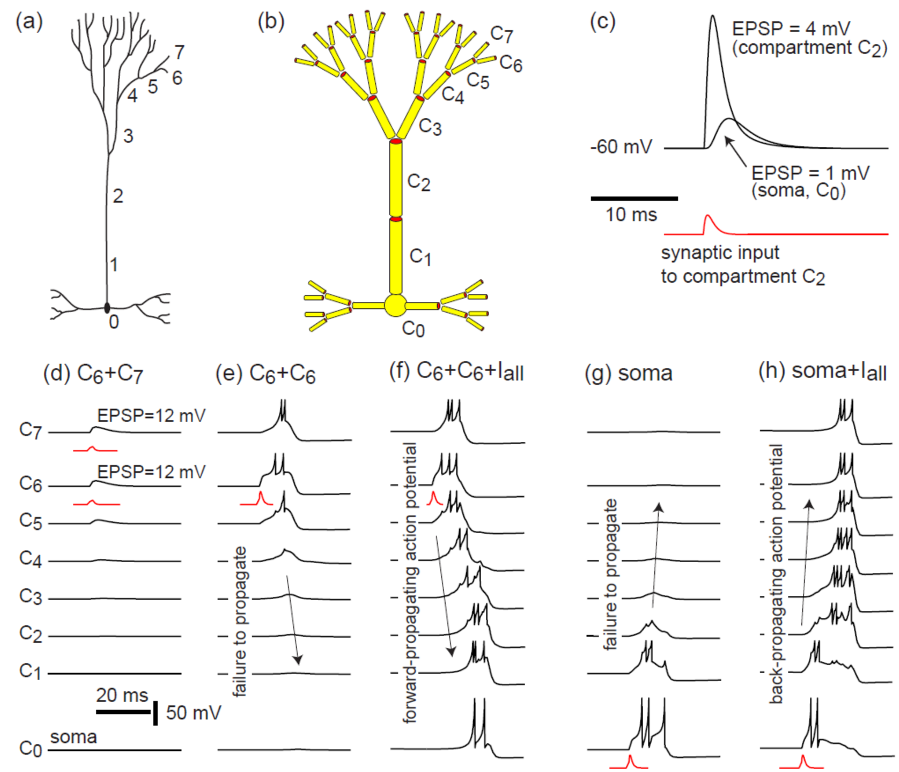
J. Physiol. (1984), **348**, *pp.* 89–113 89
With 9 text-figures
Printed in Great Britain

COMPARTMENTAL MODELS OF ELECTROTONIC STRUCTURE AND SYNAPTIC INTEGRATION IN AN IDENTIFIED NEURONE

BY DONALD H. EDWARDS, JR* AND BRIAN MULLONEY
From the Department of Zoology, University of California, Davis, CA 95616, U.S.A.

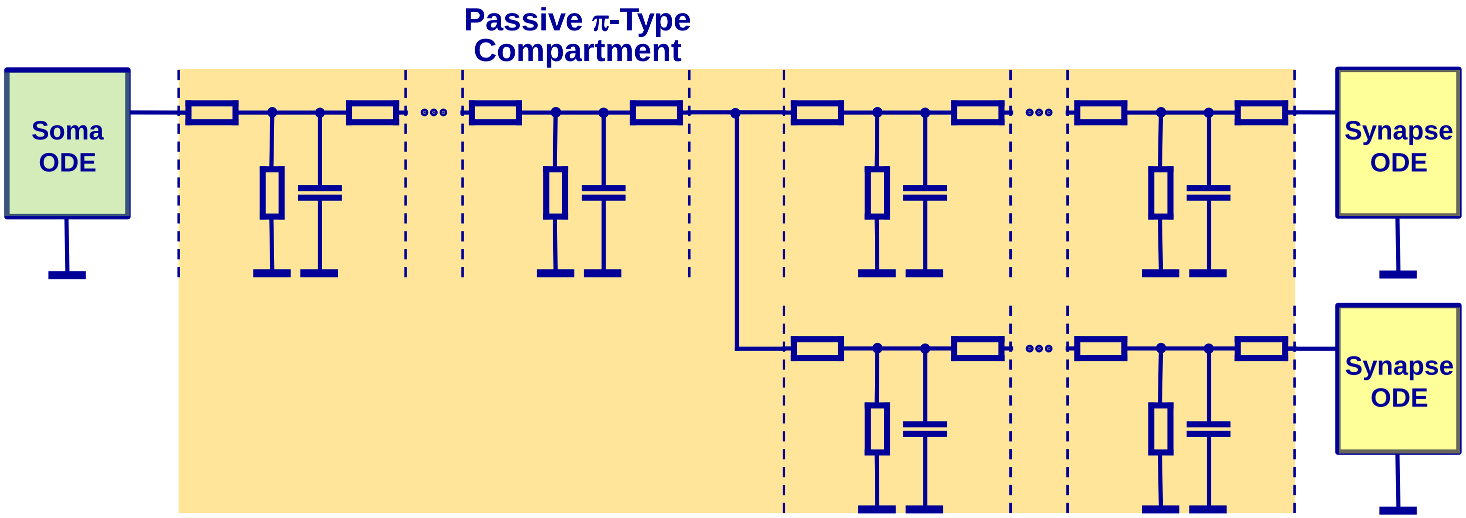






[Izhikevich, 2007]

Compartmental Dendrite Models



KNA: Coupled ODE System

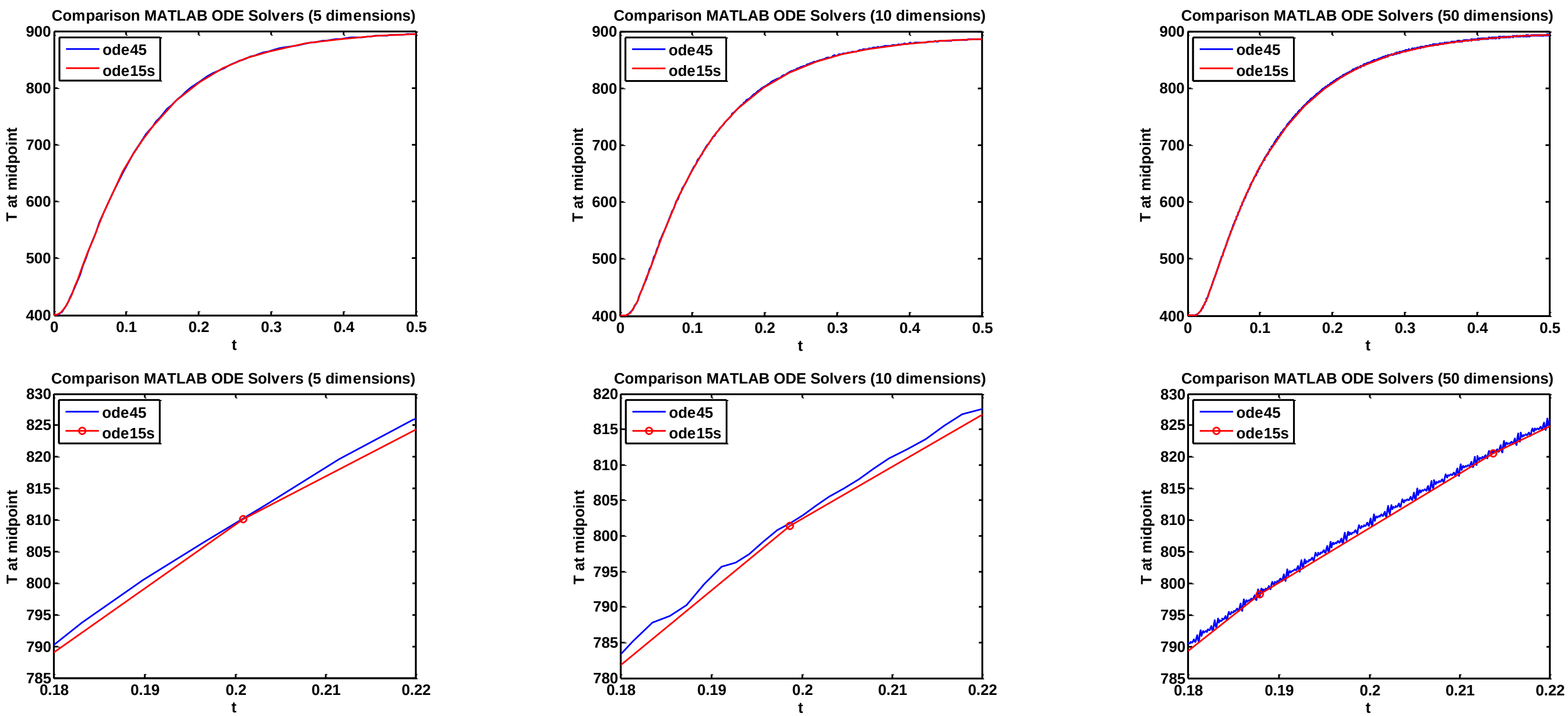
Conductance Matrix (<u>essentially</u> tri-diagonal):

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} C_m \cdot V_m \\ \vdots \\ \vdots \\ C_1 \cdot V_1 \\ C_2 \cdot V_2 \\ \\ \\ \\ \vdots \\ \\ \\ \\ \\ C_N \cdot V_N \end{bmatrix} = \begin{bmatrix} \\ \\ \\ X & X & X & & & & & & & & & \\ & X & X & X & & & & & & & 0 & \\ & & & \ddots & & & & & & & & \\ & & & X & X & X & & & & & & \\ & & & & X & X & X & & X & & & \\ & & & & & X & X & X & X & & & \\ & & & & & & & \ddots & & & & \\ & & & & & & & X & X & & & \\ & & & & & X & X & & X & X & & \\ & & & & & & & & & \ddots & & \\ 0 & & & & & & & & & X & X & X \\ & & & & & & & & & & X & X \end{bmatrix} \cdot \begin{bmatrix} V_m \\ \\ \\ V_1 \\ V_2 \\ \\ \\ \\ \\ \\ \\ \\ \\ V_N \end{bmatrix} + I_{ext}$$

Fill-in due to branching

- **Good news: Sparse matrix**
- **Bad news: Large, fully coupled system**
- **Even more bad: „Stiff“ ODE system**

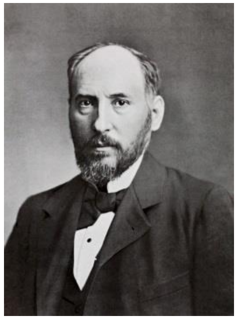
Issues with Stiff ODE Systems



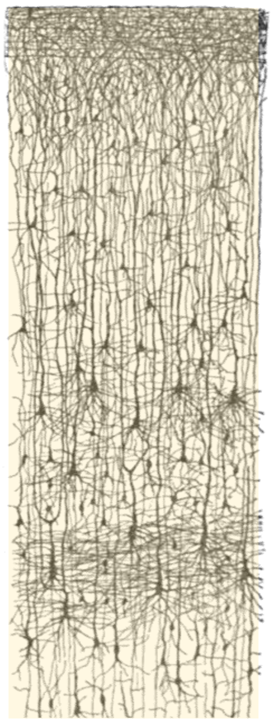
Outline

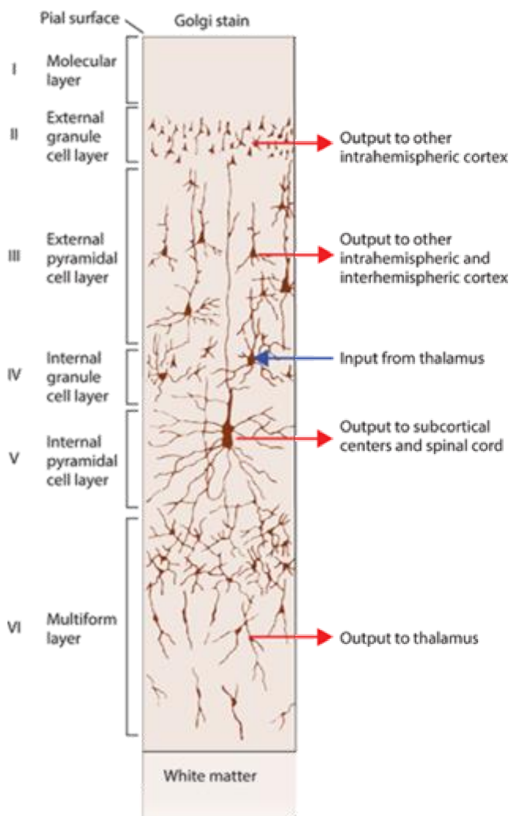
- Introduction
- Neuronal Network Components and Models
- **Exemplary Large-Scale Networks**
- Future Requirements
- Simulation Principles
  - Computation: Numerical ODE Solvers
  - Communication: AER
- State-of-the Art
- Brick Walls
  - Computation: …
  - Communication: …
- Conclusion

Neuron Populations and Balanced Networks



Santiago Ramón y Cajal, 1899



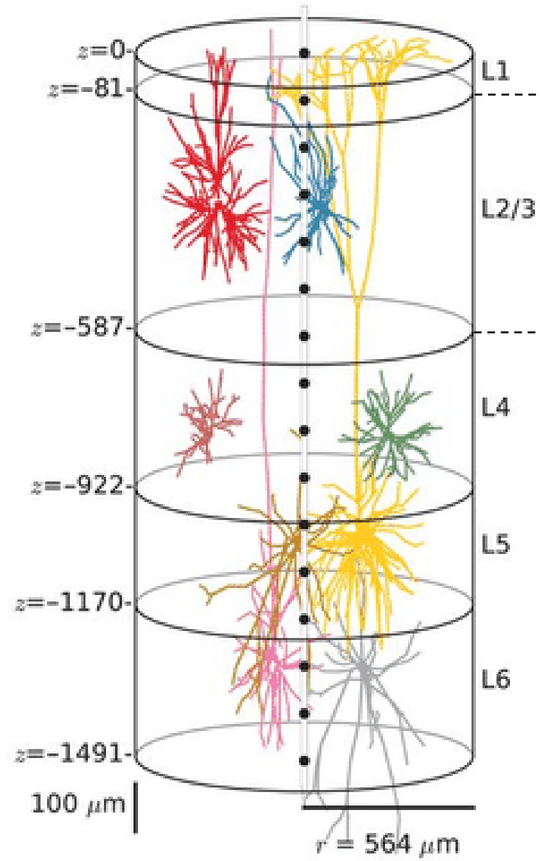


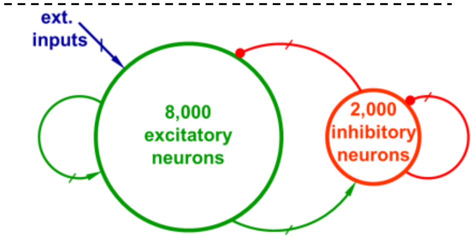
Source: Tony Mosconi, Victoria Graham: *Neuroscience for Rehabilitation*

10786 • The Journal of Neuroscience, November 16, 2005 • 25(46):10786 –10795

Signal Propagation and Logic Gating in Networks of Integrate-and-Fire Neurons

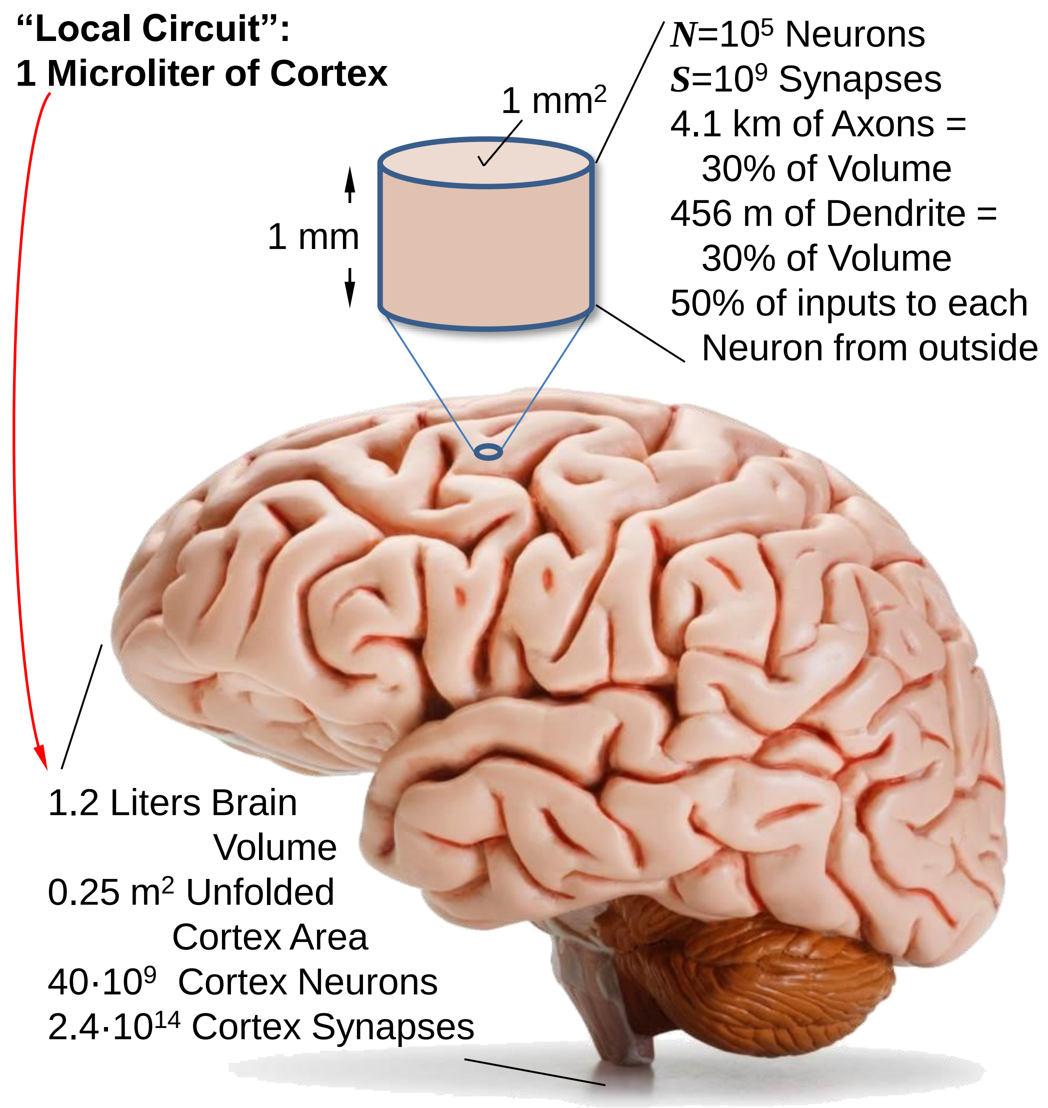
Tim P. Vogels and L. F. Abbott
Volen Center for Complex Systems and Department of Biology, Brandeis University, Waltham, Massachusetts 02454-9110





N=**10,000** LIF point-neurons
0.02·N=**200** exp. dec. synapses/neuron
CU/COBA synapses (i.e. non-linear ODE)
No axonal propagation delays
1 external synapse / neuron
Simulation: 0.1ms grid
Non-linear, non-stiff ODE solver (t.b.d.)
Fat-tree network

The Microcircuit Model

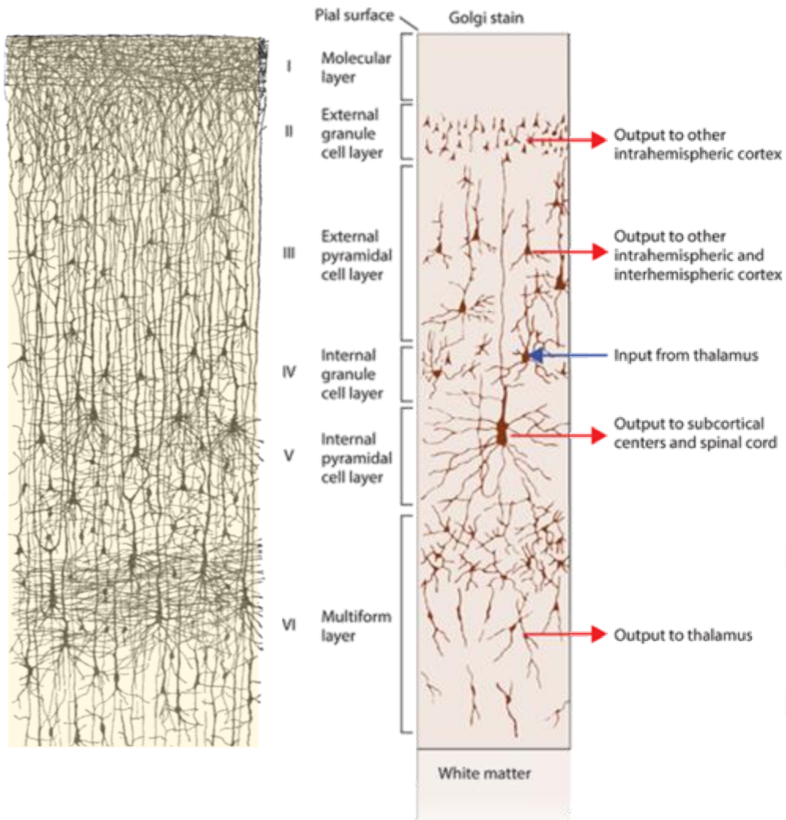


Neuron Populations and Balanced Networks

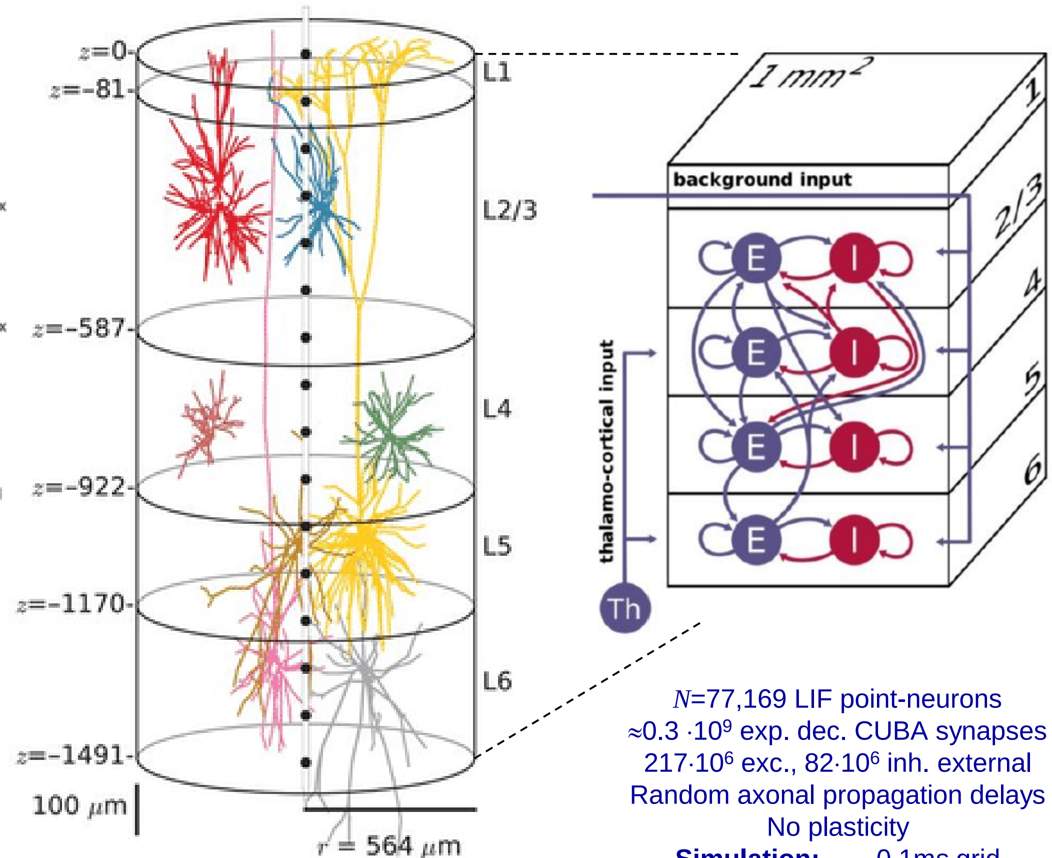
Cerebral Cortex March 2014;24:785–806
doi:10.1093/cercor/bhs358
Advance Access publication December 2, 2012

The Cell-Type Specific Cortical Microcircuit: Relating Structure and Activity in a Full-Scale Spiking Network Model

Tobias C. Potjans[1,2,3] and Markus Diesmann[1,2,4,5]



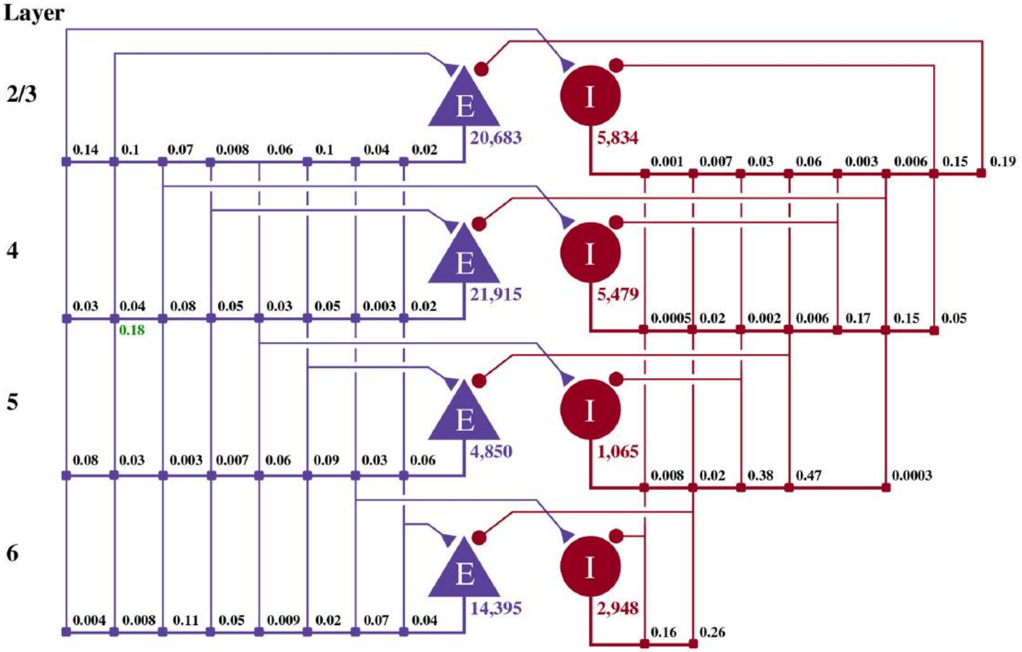
Source: Tony Mosconi, Victoria Graham: *Neuroscience for Rehabilitation*



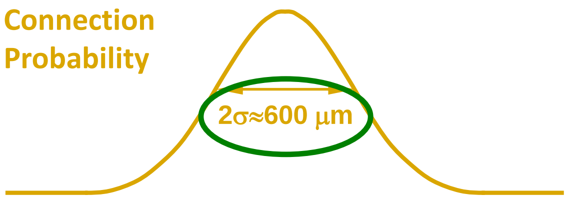
N=77,169 LIF point-neurons
≈0.3 ·10^9 exp. dec. CUBA synapses
217·10^6 exc., 82·10^6 inh. external
Random axonal propagation delays
No plasticity
Simulation: 0.1ms grid
Exact exponential integration
Uniform rand. lateral connection propab.
Network t.b.d.

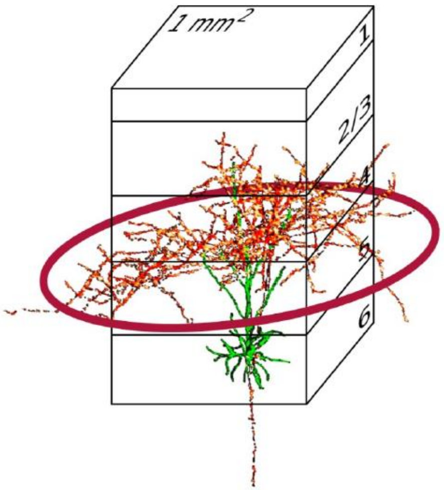
Local Connectivity

Microcircuit (≈77,000 neurons, uniform connectivity)



Local Lateral Connectivity

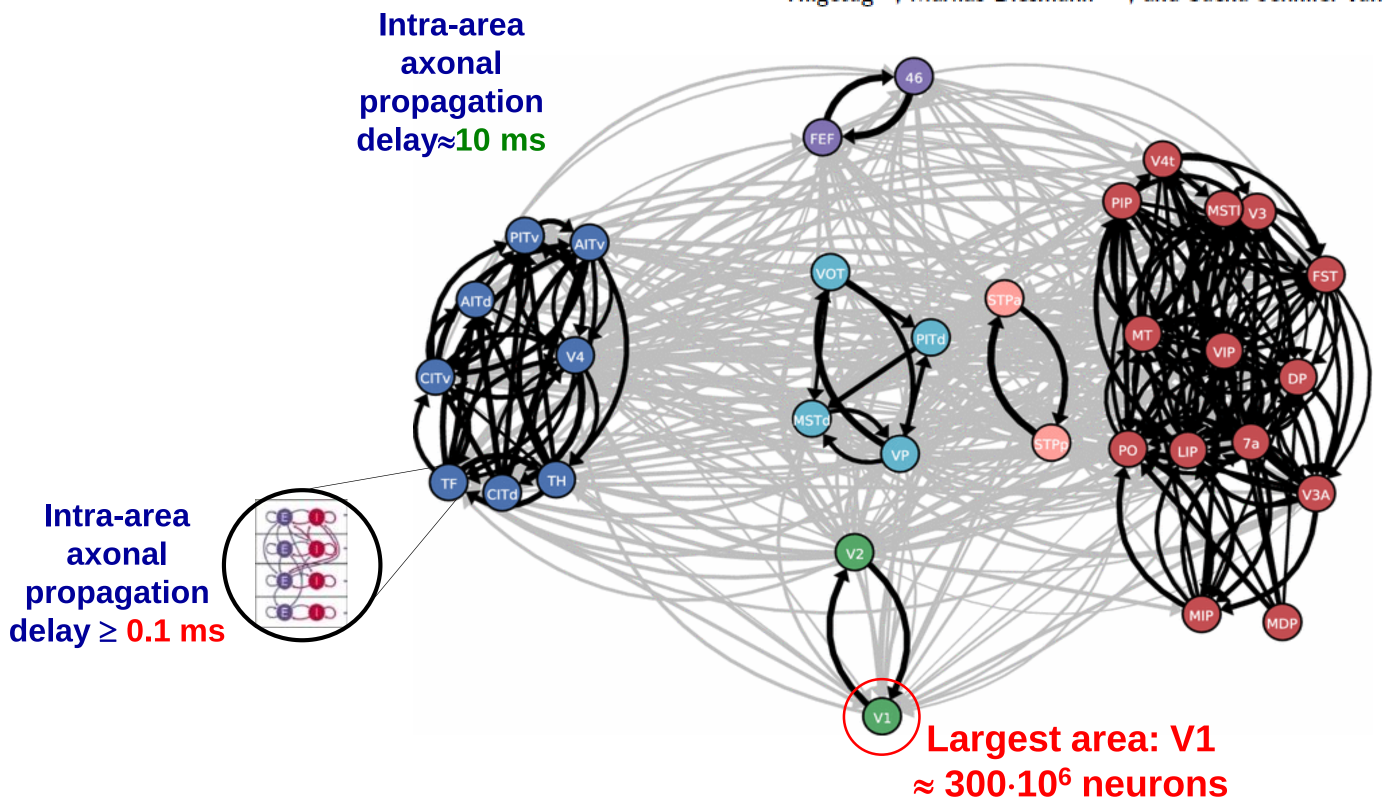




Multi-Area Models

Full-density multi-scale account of structure and dynamics of macaque visual cortex

Maximilian Schmidt[1], Rembrandt Bakker[1,2], Kelly Shen[3], Gleb Bezgin[4], Claus-Christian Hilgetag[5,6], Markus Diesmann[1,7,8], and Sacha Jennifer van Albada[1]



Downscaled Network

N=4.1 ·10^6 LIF **neurons**
3.9·10^{10} exp.-decay CUBA **synapses**
Random/deterministic axonal propagation delays
Simulation: 0.1ms grid
Exact exponential integration
Hybrid hierarchical network t.b.d.

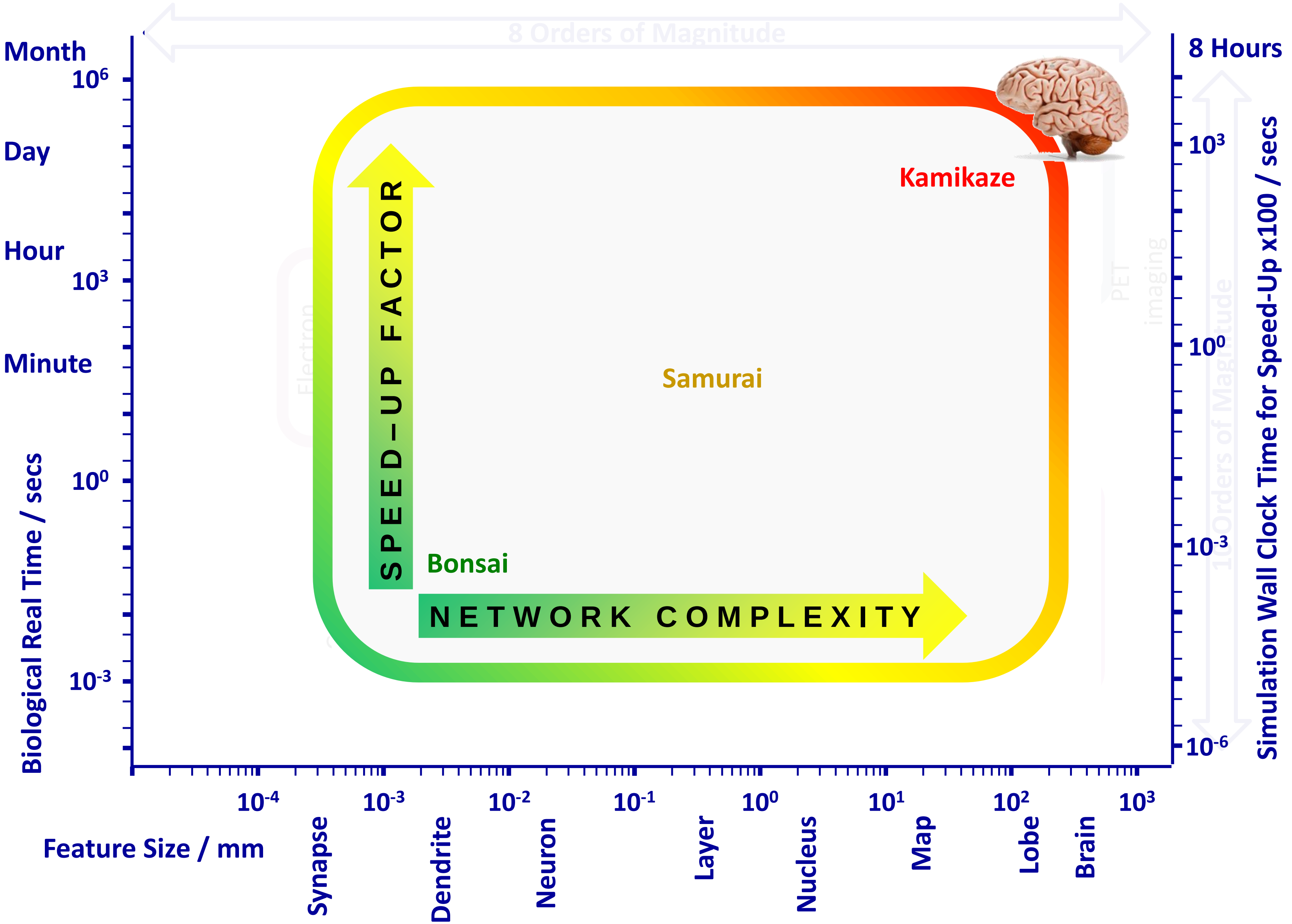
Original Network

N=8 ·10^8 LIF **neurons**
8·10^{12} exp.-decay CUBA **synapses**
Random/deterministic axonal propagation delays
Simulation: 0.1ms grid
Exact exponential integration
Hybrid hierarchical network t.b.d.

Outline

- Introduction
- Neuronal Network Components and Models
- Exemplary Large-Scale Networks
- **Future Requirements**
- Simulation Principles
    - Computation: Numerical ODE Solvers
    - Communication: AER
- State-of-the Art
- Brick Walls
    - Computation: …
    - Communication: …
- Conclusion

The Spatio-Temporal Multiscale Brain



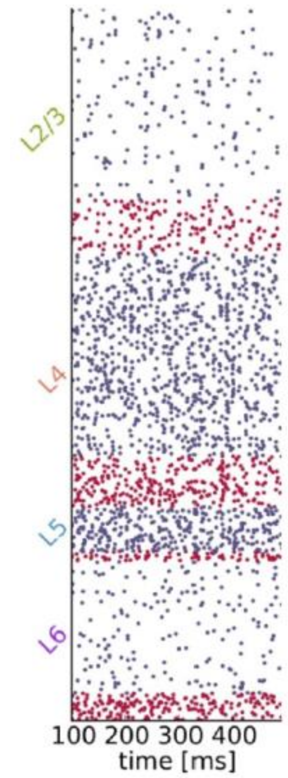
Future Requirements in Neuroscience Simulation

- Brain-Area Network Complexity (~100 Mio. Neurons)
- Significant Speed-Up w.r.t. BRT (Plasticity, Learning, Development)
- Reproducibility and Accuracy
- Taylored Flexibility (Models: „Make the common case fast“)
- Reliability
- Legacy
- Silicon Area / Volume
- Power Dissipation / Energy Consumption

Outline

- ❒ Introduction
- ❒ Neuronal Network Components and Models
- ❒ Exemplary Large-Scale Networks
- ❒ Future Requirements
- ❒ **Simulation Principles**
  - **- Computation: Numerical ODE Solvers**
  - - Communication: AER
- ❒ State-of-the Art
- ❒ Brick Walls
  - - Computation: …
  - - Communication: …
- ❒ Conclusion

Basic Principles of Digital Spiking Network Simulation



- Input to networks: Random Poisson spike trains
- Output from simulation: Recorded spike patterns
- Output: 1st-order statistics, like avg. Spike rate, ISI distribution, CV, …
  2nd-order statistics, like correlation and decorrelation
- Learning: STDP seldom; no attempt to perform functional inference, yet

Coarse Simulation Flow (w/o. Learning) Time-Driven

For each time step

Process incoming spikes

Generate Poisson stimuli

Solve ODEs

Send out spikes

Numerical Ordinary-Differential-Equation Solvers

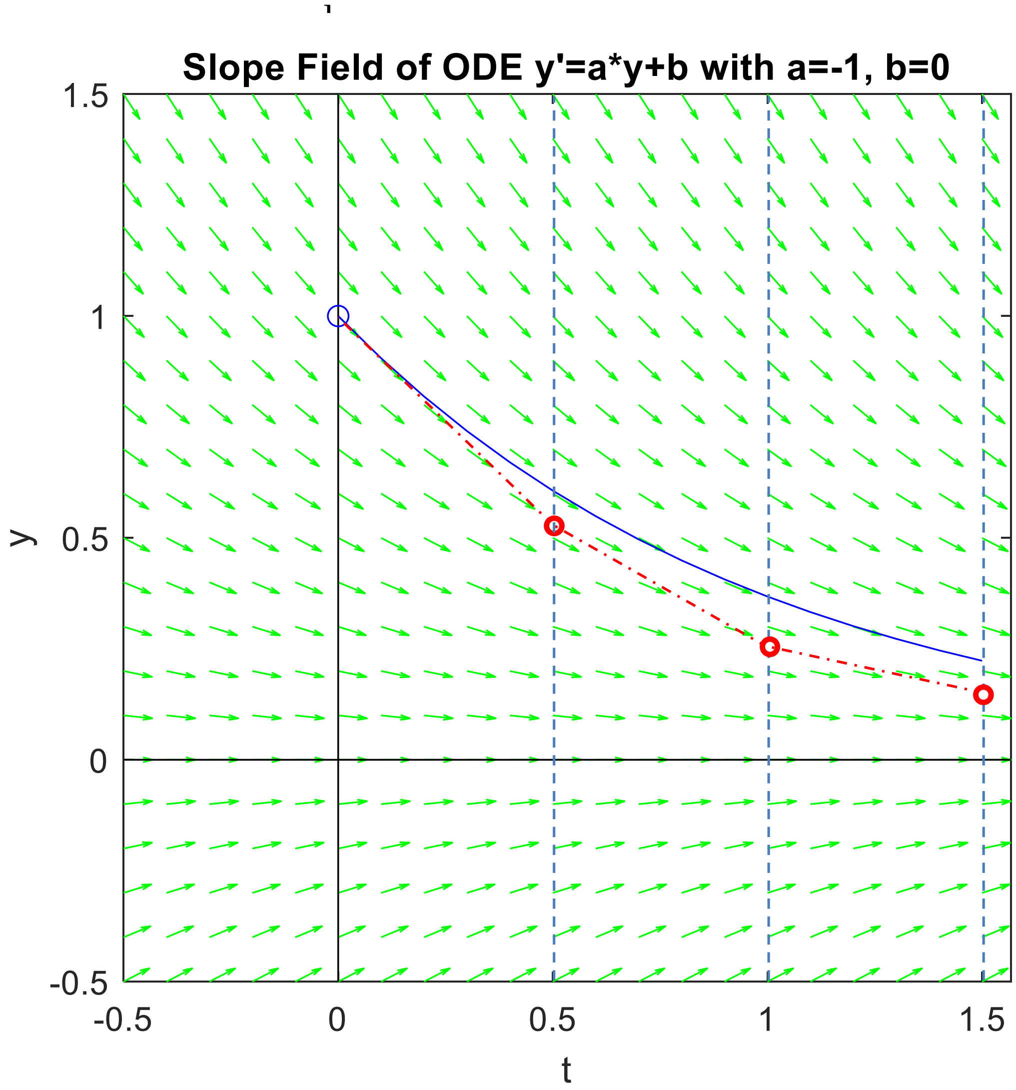
ODE: $y'(t) = \frac{\mathrm{d}}{\mathrm{d}t} y(t) = f\left[y(t)\right] = a \cdot y(t) + x(t)$ **IVP:** $y(t_0 = 0) = y_0$

N.B.: $\frac{\mathrm{d}}{\mathrm{d}t} y(t) = \lim\limits_{\Delta t \to 0} \frac{y(t + \Delta t) - y(t)}{\Delta t} = \underbrace{a \cdot y(t) + x(t)}_{\text{„slope“}}$

Discretisation in time: $\Delta t = h = \text{const.}$ $x_k := x(k \cdot h)$, $y_k :\approx y(k \cdot h)$; $k = 0, 1, 2, \ldots$

Recurrence equation: $y_{k+1} = y_k + \Delta y_k = y_k + h \cdot \text{"}slope\text{"}$

Numerical Ordinary-Differential-Equation Solvers



Numerical Ordinary-Differential-Equation Solvers

ODE: $y'(t) = \frac{\mathrm{d}}{\mathrm{d}t} y(t) = f\left[y(t)\right] = a \cdot y(t) + x(t)$ **IVP:** $y(t_0 = 0) = y_0$

N.B.: $\frac{\mathrm{d}}{\mathrm{d}t} y(t) = \lim\limits_{\Delta t \to 0} \frac{y(t + \Delta t) - y(t)}{\Delta t} = \underbrace{a \cdot y(t) + x(t)}_{\text{„slope"}}$

Discretisation in time: $\Delta t = h = \mathrm{const.}$ $x_k := x(k \cdot h)$, $y_k :\approx y(k \cdot h)$; $k = 0, 1, 2, \ldots$

Recurrence equation: $y_{k+1} = y_k + \Delta y_k = y_k + h \cdot \text{"}slope\text{"}$

Forward Euler Method: Use slope at t_k

$Use \text{ "}slope\text{" } at\ t_k$

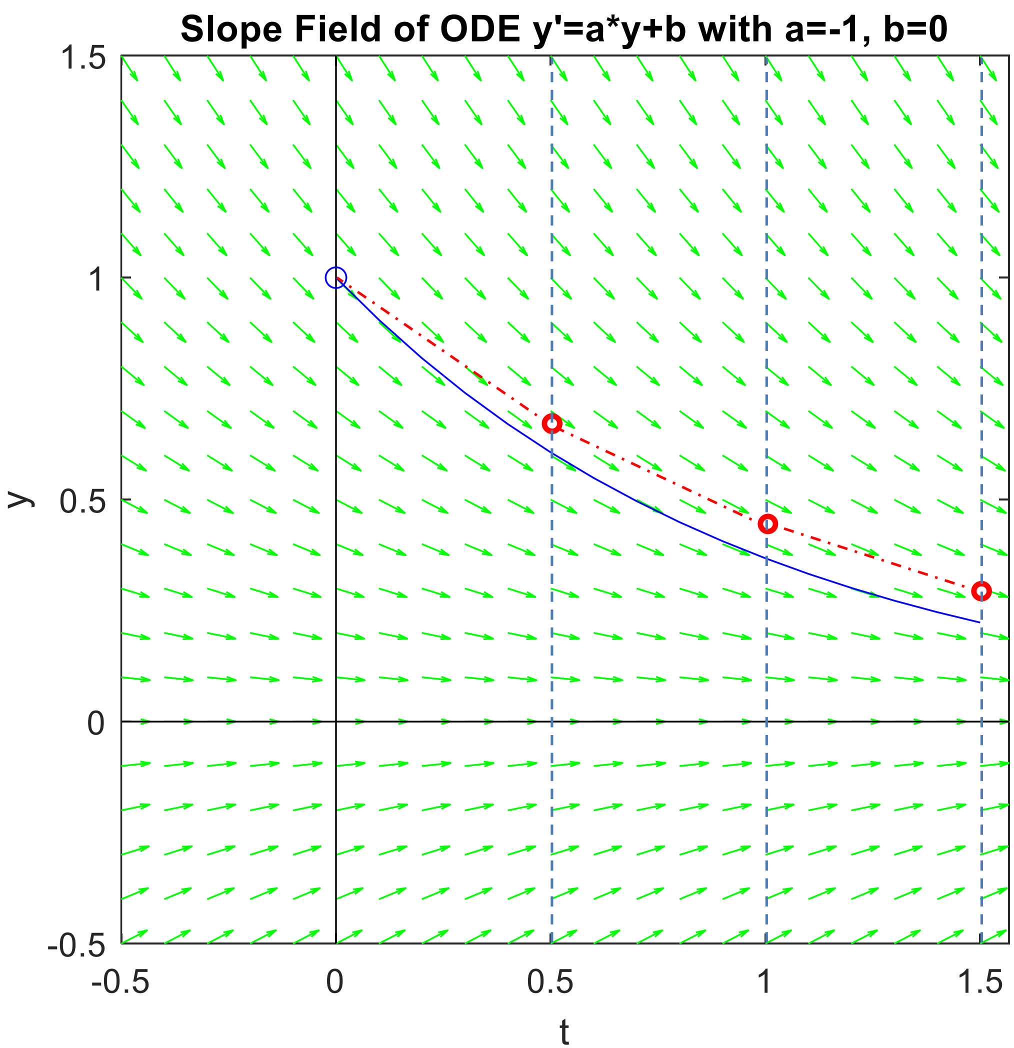
$$y_1 = y_0 + \Delta y_0 = y_0 + h \cdot \left(a \cdot y_0 + x_0\right)$$

$$y_{k+1} = y_k + h \cdot \left(a \cdot y_k + x_k\right)$$

„Explicit Method"

$$y_{k+1} = \left(1 + h \cdot a\right) \cdot y_k + h \cdot x_k$$

Numerical Ordinary-Differential-Equation Solvers



Numerical Ordinary-Differential-Equation Solvers

ODE: $y'(t) = \frac{\mathrm{d}}{\mathrm{d}t} y(t) = f\left[y(t)\right] = a \cdot y(t) + x(t)$ **IVP:** $y(t_0 = 0) = y_0$

N.B.: $\frac{\mathrm{d}}{\mathrm{d}t} y(t) = \lim\limits_{\Delta t \to 0} \frac{y(t+\Delta t) - y(t)}{\Delta t} = \underbrace{a \cdot y(t) + x(t)}_{\text{„slope“}}$

Discretisation in time: $\Delta t = h = \text{const.} \qquad x_k := x(k \cdot h) \quad , \quad y_k :\approx y(k \cdot h) \quad ; \quad k = 0, 1, 2, \ldots$

Recurrence equation: $y_{k+1} = y_k + \Delta y_k = y_k + h \cdot \text{"}slope\text{"}$

Forward Euler Method: Use slope at t_k

$Use \text{ "}slope\text{" } at\ t_k$

$$y_1 = y_0 + \Delta y_0 = y_0 + h \cdot \left(a \cdot y_0 + x_0\right)$$

$$y_{k+1} = y_k + h \cdot \left(a \cdot y_k + x_k\right)$$

„Explicit Method“

$$y_{k+1} = (1 + h \cdot a) \cdot y_k + h \cdot x_k$$

Backward Euler Method: Use slope at t_{k+1}

$Use \text{ "}slope\text{" } at\ t_{k+1}$

$$y_1 = y_0 + \Delta y_1 = y_0 + h \cdot \left(a \cdot y_1 + x_1\right)$$

$$y_{k+1} = y_k + h \cdot \left(a \cdot y_{k+1} + x_{k+1}\right)$$

„Implicit Method“

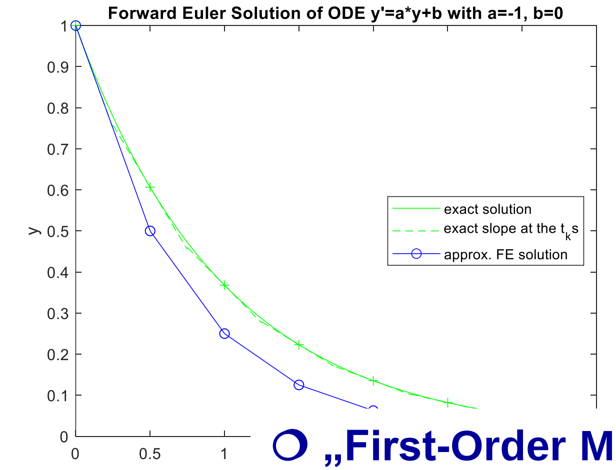
$$y_{k+1} = \frac{1}{(1 - h \cdot a)} \cdot y_k + h \cdot x_{k+1}$$

Difference equations of 1st-order IIR filters … just different coefficients

Numerical Ordinary-Differential-Equation Solvers

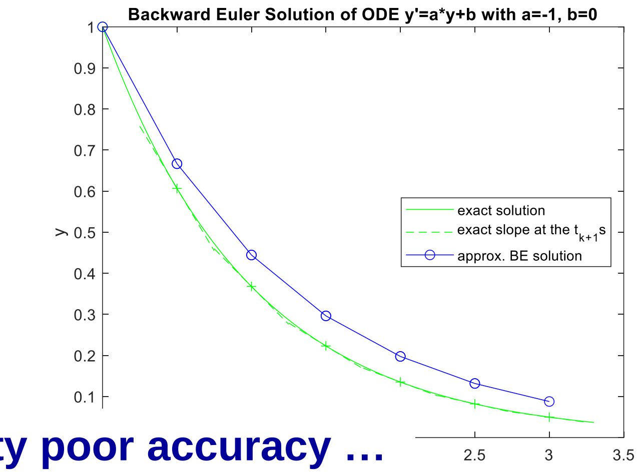
Forward Euler Method

$$y_{k+1} = y_k \cdot (1 + h \cdot a) + h \cdot x_k$$

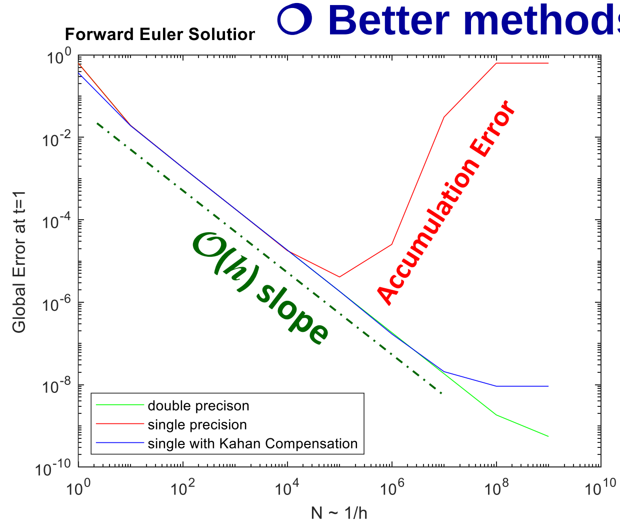


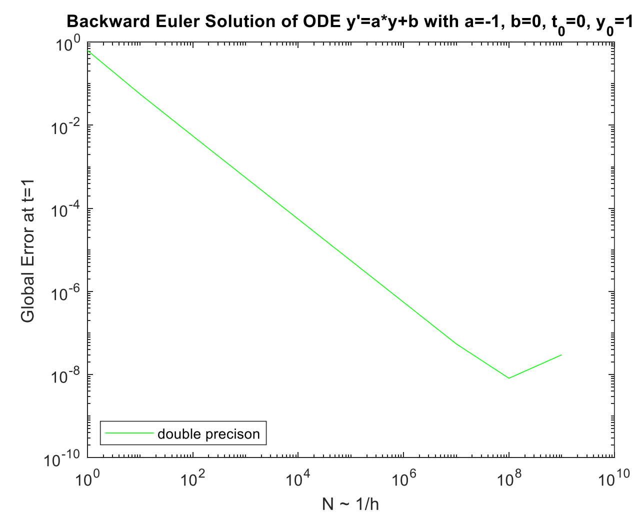
Backward Euler Method

$$y_{k+1} = y_k + \frac{h}{(1 - h \cdot a)} \cdot x_{k+1}$$



- „First-Order Methods“: Pretty poor accuracy …
- Better methods exist …



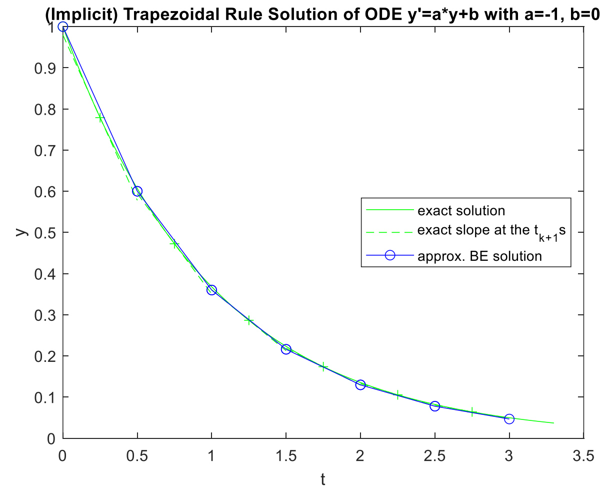


Numerical Ordinary-Differential-Equation Solvers

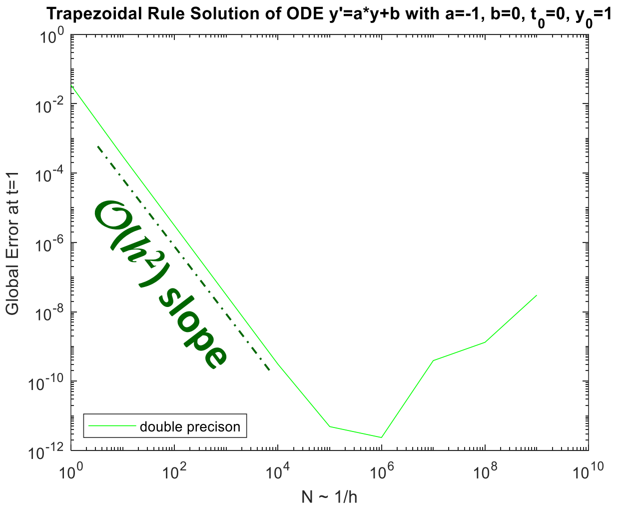
Implicit Trapezoidal Rule: Use slope avg. slope at t_k and t_{k+1}

$$y_{k+1} = \frac{(1 + h/2 \cdot a)}{(1 - h/2 \cdot a)} \cdot y_k + h/2 \cdot (x_k + x_{k+1})$$

Difference equations of 2nd-order IIR filter
In DSP filter design: „*Bilinear Transform*“

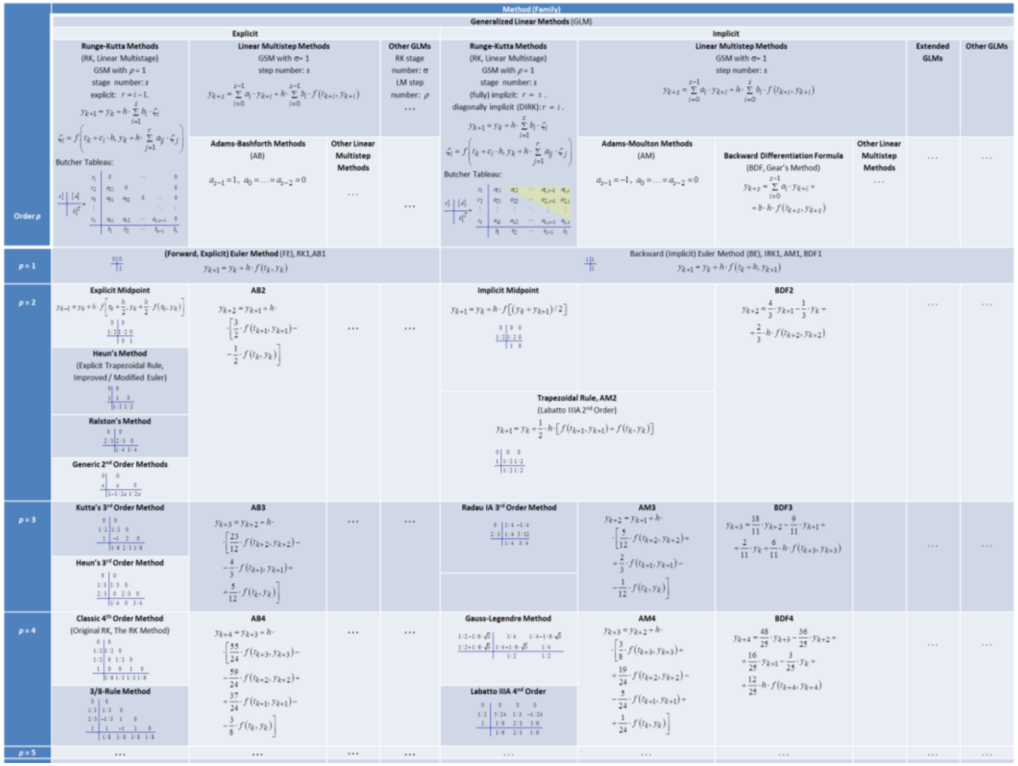


- **„Second-Order Method“: Better accuracy …**
  …at higher operation count
- **Methods with even higher orders exist …**



More Sophisticated Numerical ODE Solvers

- Linear Multistep, Runge-Kutta Multistage, …



- And many, many more …
- All need to compromise operation count per step vs. step size and are applicable to
- … linear and non-linear ODEs
- … systems of coupled ODEs: Requires solution of linear / non-linear system of algebraic equations
- But for the linear case, there‘s a „champion“ …

Numerical Ordinary-Differential-Equation Solvers

Exact Exponential Integration – for linear ODEs and coupled systems ONLY !

$$y'(t) = a \cdot y(t) + x(t) \qquad s \cdot Y(s) = a \cdot Y(s) + X(s)$$

Impulse response, i.e. $x(t) = \delta(t)$

$$h(t) = \exp(a \cdot t) \qquad H(s) = \frac{Y(s)}{X(s)} = \frac{1}{s-a}$$

Analytical solution $y(t) = y_0 \cdot \exp(a \cdot t) + x(t) * h(t) = y_0 \cdot \exp(a \cdot t) + \int_{s=0}^{t} x(t) \cdot \exp\left[a \cdot (\mathrm{t} - s)\right] \cdot \mathrm{d}s$

Recurrence equation: Consider each simulation step as an IVP

$$y_{k+1} = y_k \cdot \exp\left(a \cdot h\right) + \int_{s=0}^{h} x(k \cdot h + s) \cdot \exp\left[a \cdot \left[(h-s)\right]\right] \mathrm{d}s$$

- **1st order IIR; DSP filter design „*Impulse Invariant Mapping*“**
- **If convolution integral can be solved analytically : Exact – i.e., zero error**
- **N.B.: Ultimate input to neuron-synapse models are Dirac-pulse trains – i.e., the integral becomes trivial …**
- **Can be easily extended to linear ode systems: Apply matrix calculus and definition of „matrix exponential“:**

$$\exp(a) = 1 + \frac{1}{1!} \cdot a + \frac{1}{2!} \cdot a^2 + \frac{1}{3!} \cdot a^3 + \ldots \quad \Rightarrow \quad \exp([A]) = 1 + \frac{[A]}{1!} + \frac{[A]^2}{2!} \cdot + \frac{[A]^3}{3!} \cdot + \ldots$$

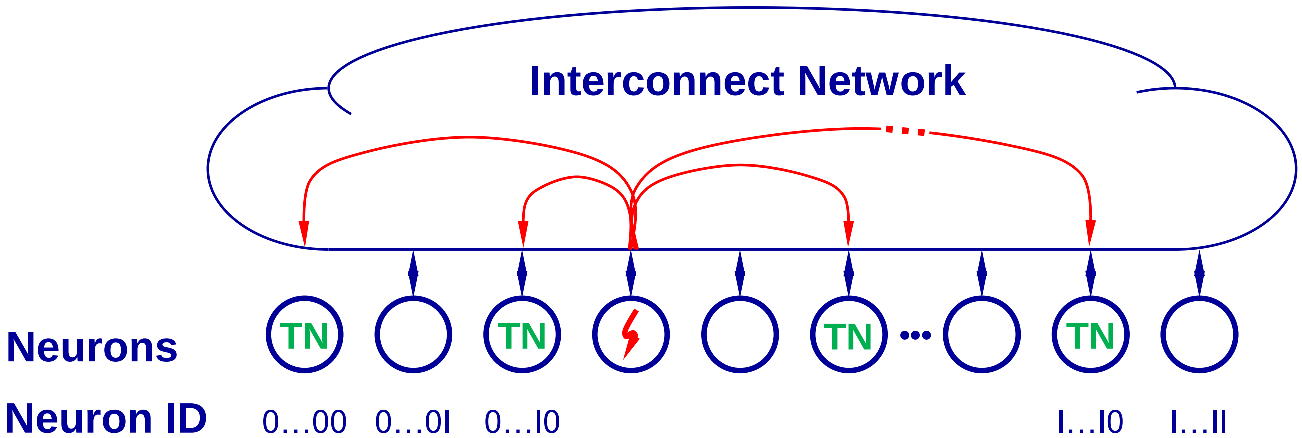
Outline

- ❒ Introduction
- ❒ Neuronal Network Components and Models
- ❒ Exemplary Large-Scale Networks
- ❒ Future Requirements
- ❒ **Simulation Principles**
    - - Computation: Numerical ODE Solvers
    - **- Communication: AER**
- ❒ State-of-the Art
- ❒ Brick Walls
    - - Computation: …
    - - Communication: …
- ❒ Conclusion

Communication Scheme

❒ **Action Potential = „Event“ ⇨ Communication Packet**

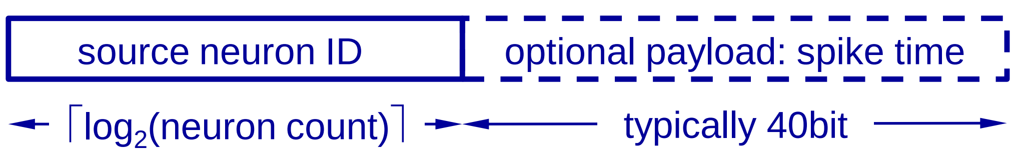
❒ **Source Address Event Representation (AER)**



❍ **Every neuron needs address translation:**

source neuron ID → target synapse ID

❍ **Packet layout:**



Outline

- Introduction
- Neuronal Network Components and Models
- Exemplary Large-Scale Networks
- Future Requirements
- Simulation Principles
  - Computation: Numerical ODE Solvers
  - Communication: AER
- **State-of-the Art**
- Brick Walls
  - Computation: …
  - Communication: …
- Conclusion

Neuroscience-Simulation Tools: Software

There's an algorithm to simulate our brains. Too bad no computer can run it

by TRISTAN GREENE — Mar 22, 2018 in ARTIFICIAL INTELLIGENCE

Scientists just created an algorithm capable of performing a complete human brain simulation. Now we just have to wait for someone to build a computer powerful enough to run it.

The team, comprised of researchers from Germany, Japan, Norway, and Sweden, recently published a white paper detailing the new algorithm, which connects virtual neurons with nodes. It's designed to simulate the brain's one billion connections between individual neurons and synapses.

A human brain's neuronal activity is incredibly complex and simulating it at a 1:1 ratio is impossible with current technology. Achieving just a 10 percent simulation rate maxes out the supercomputers that such limited simulations have been run on in the past. This is because the act of connecting neurons — crucial for every activity that happens in the brain — requires more power than today's hardware has. According to a *Kurzweil Network* article:

> *That process requires one bit of information per processor for every neuron in the whole network. For a network of one billion neurons, a large part of the memory in each node is consumed by this single bit of information per neuron. Of course, the amount of computer memory required per processor for these extra bits per neuron increases with the size of the neuronal network. To go beyond the 1 percent and simulate the entire human brain would require the memory available to each processor to be 100 times larger than in today's supercomputers.*

The new algorithm won't allow scientists to run those simulations now, but in theory it has "extreme scalability" that will work with future 'exascale' hardware. It was built using open source simulation software called neural simulation tool (NEST), which is widely used in the neuroscientific community.

By scaling the algorithm with future exascale supercomputers, researchers hope to reach 100 percent simulation. This would represent a watershed moment in several fields of scientific endeavor.

Such a simulation could change the course of research concerning brain disorders ranging from Parkinson's disease to multiple sclerosis. And the implications for artificial intelligence research and neural network design could involve an entirely new perspective on deep learning.

Scientists have worked for decades to simulate the human brain using computers and math. This algorithm is a bridge between what we knew about our minds yesterday, and what we'll know tomorrow.

frontiers in Neuroinformatics

ORIGINAL RESEARCH
published: 16 February 2018
doi: 10.3389/fninf.2018.00002

Extremely Scalable Spiking Neuronal Network Simulation Code: From Laptops to Exascale Computers

Jakob Jordan[1], Tammo Ippen[1,2], Moritz Helias[1,3], Itaru Kitayama[4], Mitsuhisa Sato[4], Jun Igarashi[5], Markus Diesmann[1,3,6] and Susanne Kunkel[7,8]*

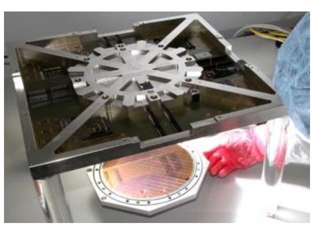
[1] Institute of Neuroscience and Medicine (INM-6) and Institute for Advanced Simulation (IAS-6) and JARA Institute Brain Structure-Function Relationships (INM-10), Jülich Research Centre, Jülich, Germany, [2] Faculty of Science and Technology, Norwegian University of Life Sciences, Ås, Norway, [3] Department of Physics, Faculty 1, RWTH Aachen University, Aachen, Germany, [4] Advanced Institute for Computational Science, RIKEN, Kobe, Japan, [5] Computational Engineering Applications Unit, RIKEN, Wako, Japan, [6] Department of Psychiatry, Psychotherapy and Psychosomatics, Medical Faculty, RWTH Aachen University, Aachen, Germany, [7] Department of Computational Science and Technology, School of Computer Science and Communication, KTH Royal Institute of Technology, Stockholm, Sweden, [8] Simulation Laboratory Neuroscience – Bernstein Facility for Simulation and Database Technology, Jülich Research Centre, Jülich, Germany

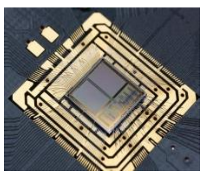
Neuroscience-Simulation Tools: Hardware

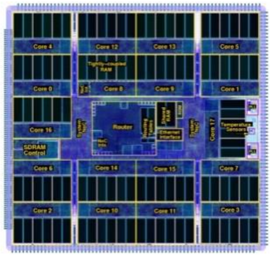
Human Brain Project Systems

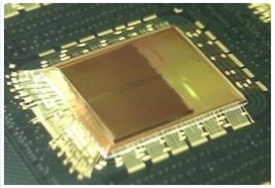
BrainScaleS **(Univ. Heidelberg)**	**SpiNNaker** **(Univ. Manchester)**
Analog Emulation / Digital Communication **„Physical-Model Emulator“**	**Fully Digital & Programmable** **„Many-DSP-Core“**
x 1,000 … 10,000 accelerated 4M Neurons / 1B Synapses Adaptive Exponential IF Wafer Scale / Ethernet 180-nm CMOS	„Real-Time Simulator“ 920M Neurons / 460B Synapses 1 Mio. ARM Cores, 16 bit 2D-Mesh Toroid Multicast 130-nm CMOS

4 Million Neurons
1 Billion Synapses









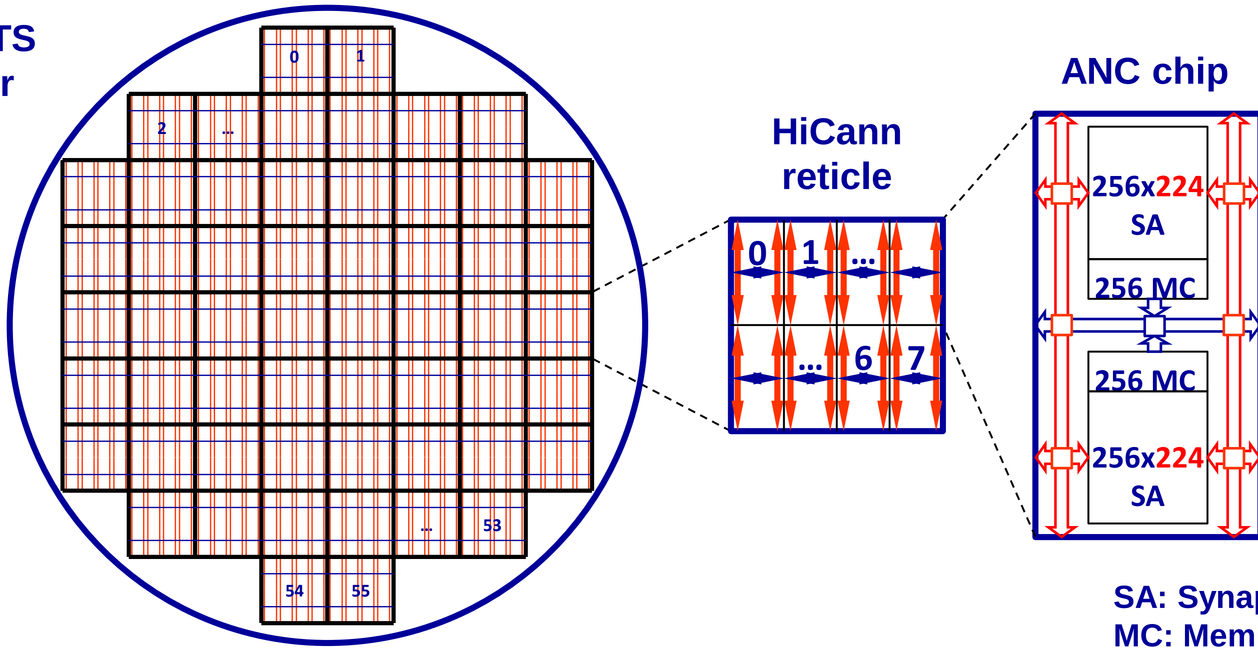
Neuroscience-Simulation Tools: Hardware

Wafer-Scale Integration of Analog Neural Networks

Johannes Schemmel, Johannes Fieres and Karlheinz Meier

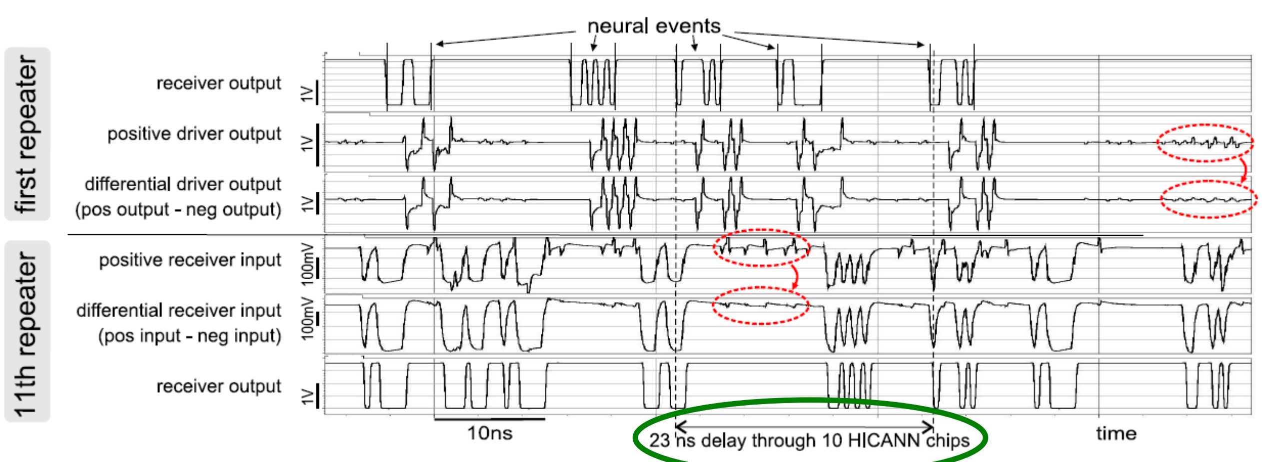
2008 International Joint Conference on Neural Networks (IJCNN 2008)

FACETS Wafer



SA: Synapse Array
MC: Membrane Circuit

Communication: Leased line with X64 TDM

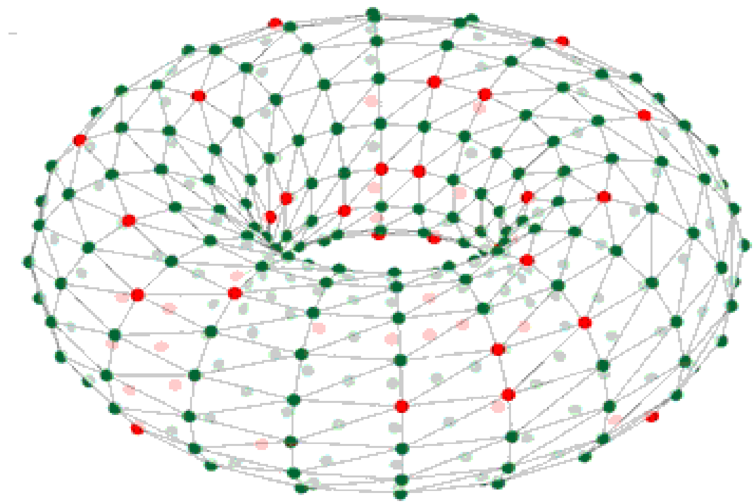


Neuroscience-Simulation Tools: Hardware

Understanding the Interconnection Network of SpiNNaker

Javier Navaridas†, Mikel Luján*, Jose Miguel-Alonso†, Luis A. Plana*, Steve Furber*

ICS'09, June 8–12, 2009, York Town Heights, New York, USA



Distance 256 x 256 Toroid

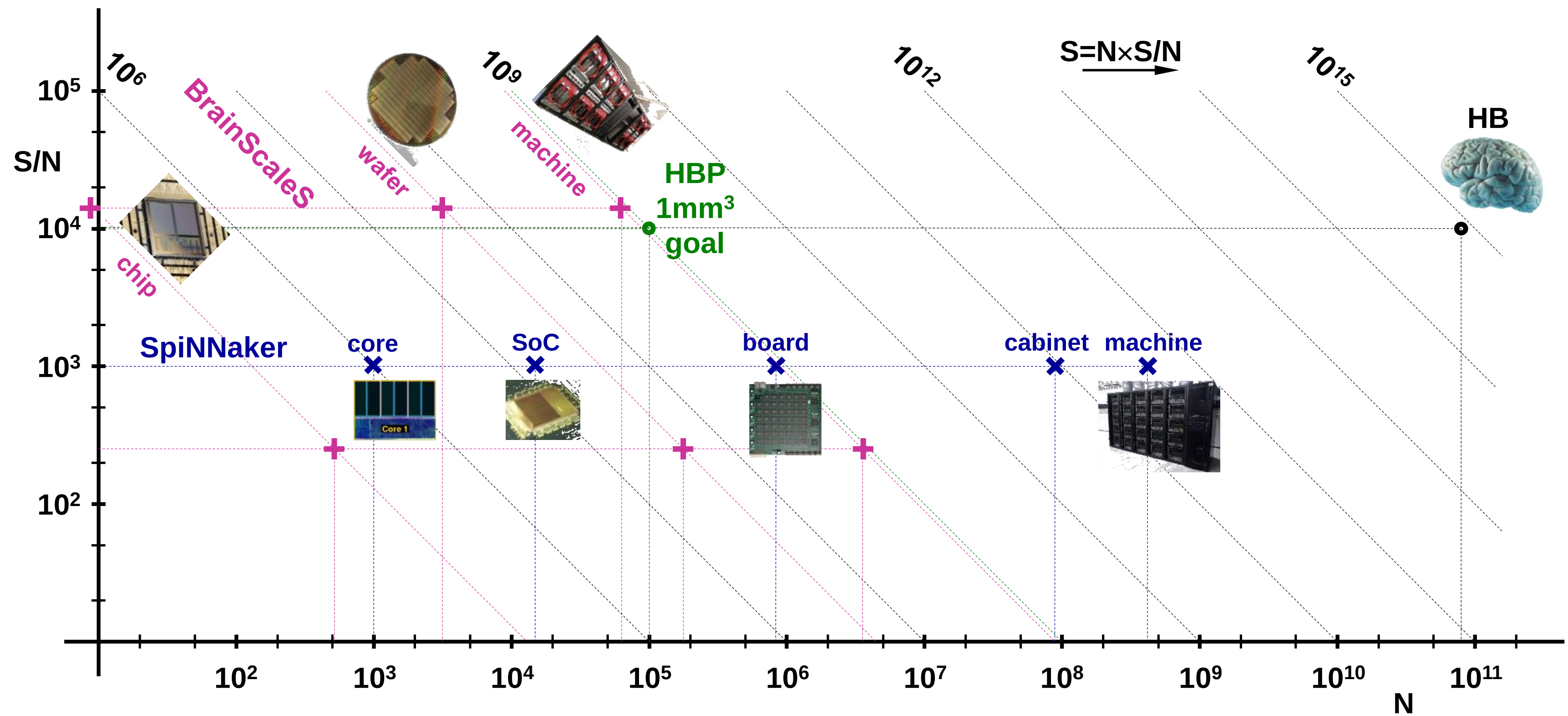
$$D = \frac{n}{2} + \left\lfloor \frac{n}{6} \right\rfloor = \left\lfloor \frac{2 \cdot n}{3} \right\rfloor$$ **= 170 hops**

Latency:
chip2chip 0.2 us/hop ⇨ 34 us (on board)
Board2board (SATA cable) 0.5 us / hop

Communication:
Packet switched with propriatary asynchronous interconnect

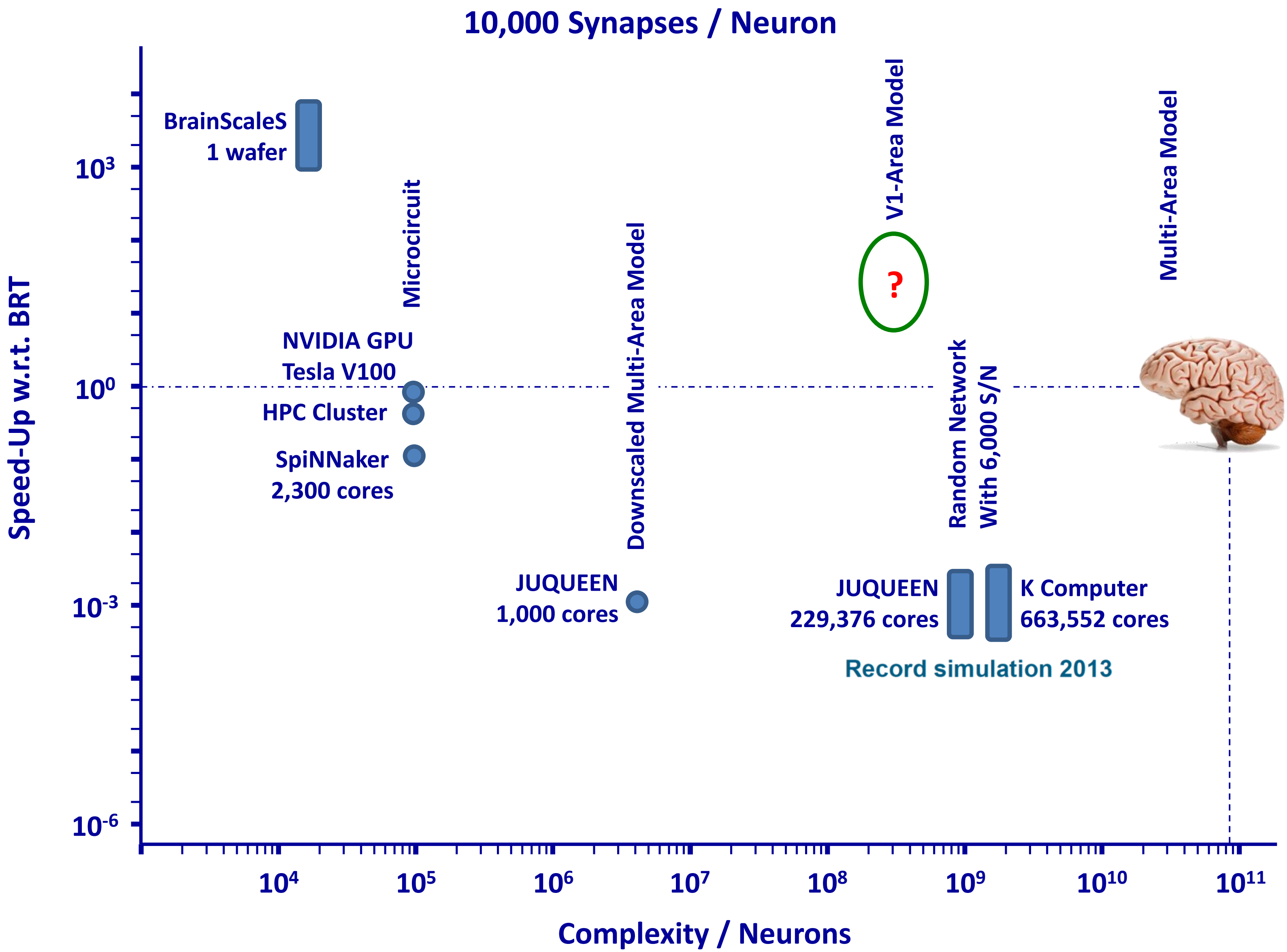
Neuroscience-Simulation Tools: Hardware

Neuroscience Simulation Platforms HBP, 2017



(theoretical capabilities)

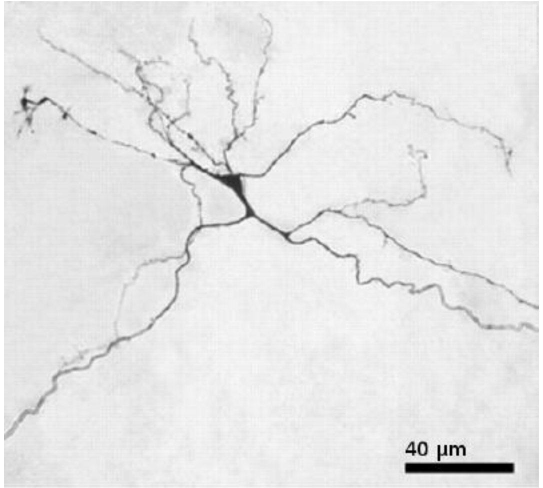
Neuroscience-Simulation Tools: Hardware



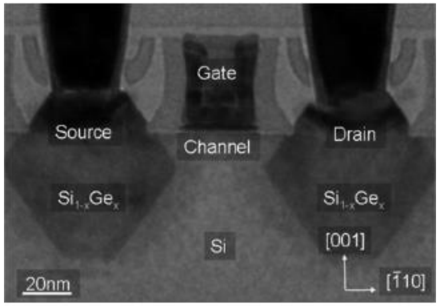
Outline

- ❒ Introduction
- ❒ Neuronal Network Components and Models
- ❒ Exemplary Large-Scale Networks
- ❒ Future Requirements
- ❒ Simulation Principles
  - Computation: Numerical ODE Solvers
  - Communication: AER
- ❒ State-of-the Art
- ❒ **Brick Walls**
  - **Computation: …**
  - **Communication: …**
- ❒ Conclusion

Biological Wetware vs. Nano-Scale CMOS



Lin et al. (2003), J. Neurophys., 90, 4
5



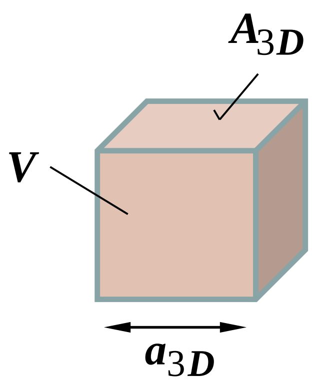
Imaging & Microscopy
(http://www.imaging-git.com

- Size of neurons: 10-100um
- Size of modern transistor: 10-100nm
  - in 2d, 1 million transistors fit into 1 neuron
- Number of neurons in cortex: about 10^{10}
- Number of transistors in modern microprocessor (Intel Broadwell-E5): about 10^{10}
  - ➢ Can we just scale this technology up?

❍ **So, why is it so difficult ?**

Biological Wetware vs. Nano-Scale CMOS

Mapping a Cube Model to a 2D Sheet of Neurons



Microcircuit (1 Microliter of Cortex):

Axonal diameter ≈ 0.3 μm

a_{3D} = 1 mm

$N = 10^5$ Neurons ⇨ $n_V = 10^5$ mm^{-3} Neuron density

l_a = 40 mm axonal length per neuron

Aggregated total axonal length

$$L_a = l_a \cdot N = 4{\cdot}10^3 \text{ m}$$

[*Dmitri B. Chklovskii,2004*]

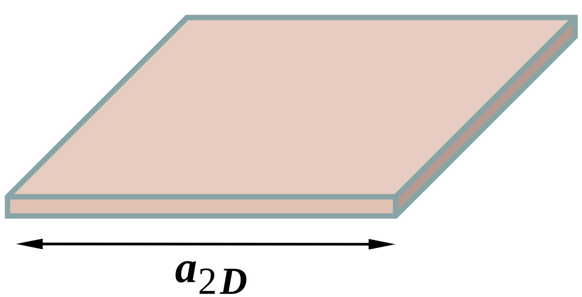
Generic Cube:

$$V = a_{3D}^3$$

$$N = a_{3D}^3 \cdot n_V$$

$$L_{a,3D} = l_a \cdot a_{3D}^3 \cdot n_V = a_{3D}^3 \cdot 4 \cdot 10^6 \text{ mm}^{-2}$$

2-Dimensional Sheet of Neurons:

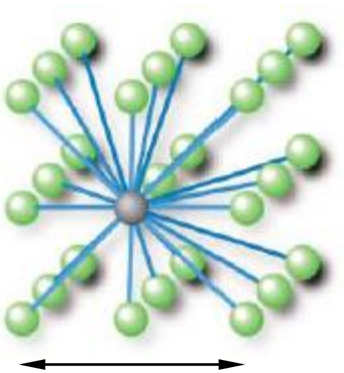


$$a_{2D} = a_{3D} \cdot \sqrt[6]{N} = a_{3D} \cdot \sqrt[6]{a_{3D}^3 \cdot n_V} = a_{3D}^{3/2} \cdot \sqrt[6]{n_V}$$

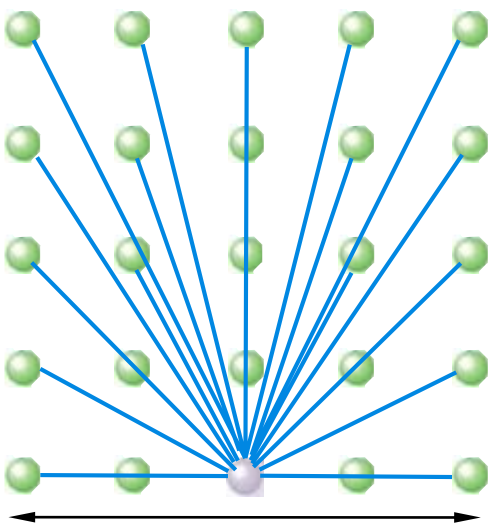
$$\sqrt[6]{n_V} = \sqrt[6]{10^5}\ \sqrt{1/\text{mm}} = 6.813 \text{ mm}^{-1/2}$$

$$a_{2D} = a_{3D}^{3/2} \cdot 6.813 \text{ mm}^{-1/2}$$

Biological Wetware vs. Nano-Scale CMOS



a_{3D}



a_{2D}

2D- vs 3D-Interconnect Wiring Lengths

$$l_{a,avg,3D} \sim a_{3D}$$

$$l_{a,avg,2D} \sim a_{2D}$$

$$L_{a,2D} = \frac{l_{a,avg,2D}}{l_{a,avg,3D}} \cdot L_{a,3D} \approx \frac{a_{2D}}{a_{3D}} \cdot a_{3D}^{3} \cdot 4 \cdot 10^{6}\ \text{mm}^{-2}$$

$$a_{2D} = a_{3D}^{3/2} \cdot 6.813\ \text{mm}^{-1/2}$$

$$L_{a,2D} \approx \frac{a_{3D}^{3/2}}{a_{3D}} \cdot a_{3D}^{3} \cdot 4 \cdot 10^{6}\ \text{mm}^{-2} \cdot 6.813\ \text{mm}^{-1/2}$$

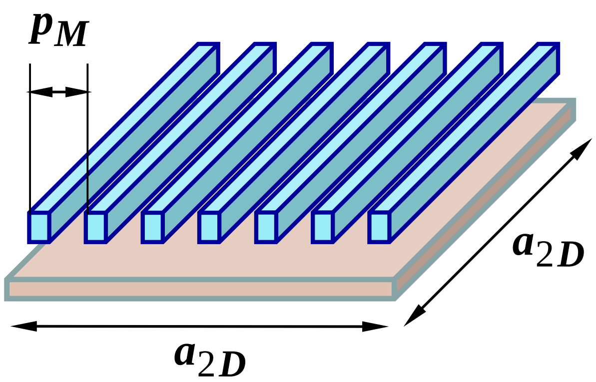
$$= a_{3D}^{3/2} \cdot a_{3D}^{2} \cdot 4 \cdot 10^{6}\ \text{mm}^{-2} \cdot 6.813\ \text{mm}^{-1/2}$$

$$L_{a,2D} \approx a_{3D}^{7/2} \cdot 27.252 \cdot 10^{6}\ \text{mm}^{-3/2}$$

Biological Wetware vs. Nano-Scale CMOS

Required 2D-Interconnect Density

M Layers of Metal Interconnect:
p_M Metal pitch
M Number of available metal layers
η Metal layer utilization



Available Aggregated Interconnect length

$$L_{int} = \eta \cdot \frac{M}{p_M} \cdot a_{2D}^2$$

Required Normalized Pitch

$$\pi_{req} := \frac{p_M}{M} = \eta \cdot \frac{a_{2D}^2}{L_{int}} = \eta \cdot \frac{a_{2D}^2}{L_{a,2D}} = \eta \cdot \frac{\left(a_{3D}^{3/2} \cdot 6.813\ \mathrm{mm}^{-1/2}\right)^2}{a_{3D}^{7/2} \cdot 27.252 \cdot 10^6\ \mathrm{mm}^{-3/2}}$$

$$\pi_{req} = \eta \cdot \frac{1.704 \cdot 10^{-6}\ \mathrm{mm}^{3/2}}{\sqrt{a_{3D}}}$$

Microcircuit with a_{3D} = 1 mm, assuming η = 0.5:

$L_{a,\,3D}$ = $4 \cdot 10^6$ mm
a_{2D} = 6.813 mm
$L_{a,\,2D}$ = $27.252 \cdot 10^6$ mm
π_{req} = $0.852 \cdot 10^{-6}$ mm = 0.852 nm

Biological Wetware vs. Nano-Scale CMOS

Available 2D-Interc. Densities in Select CMOS Nodes

CMOS Node	180 nm (BrainScaleS)	130 nm (SpiNNaker)	28 nm (Economic)	7 nm (2018 ?)	5 nm (Final ?)
N	1	10^2	10^5	$27 \cdot 10^5$	
M_{max} [1)]	6-4=2	6-4=2	10-4=6	12-4=8	14-4=10
$p_{M,\,min}$ [2)]	500 nm	340 nm	90 nm	38 nm	24 nm
$\pi_{available}$	250 nm	170 nm	15 nm	4.75 nm	2.4 nm

1) Assuming two metal layers for local interconnect and two layers for supply and clock distribution

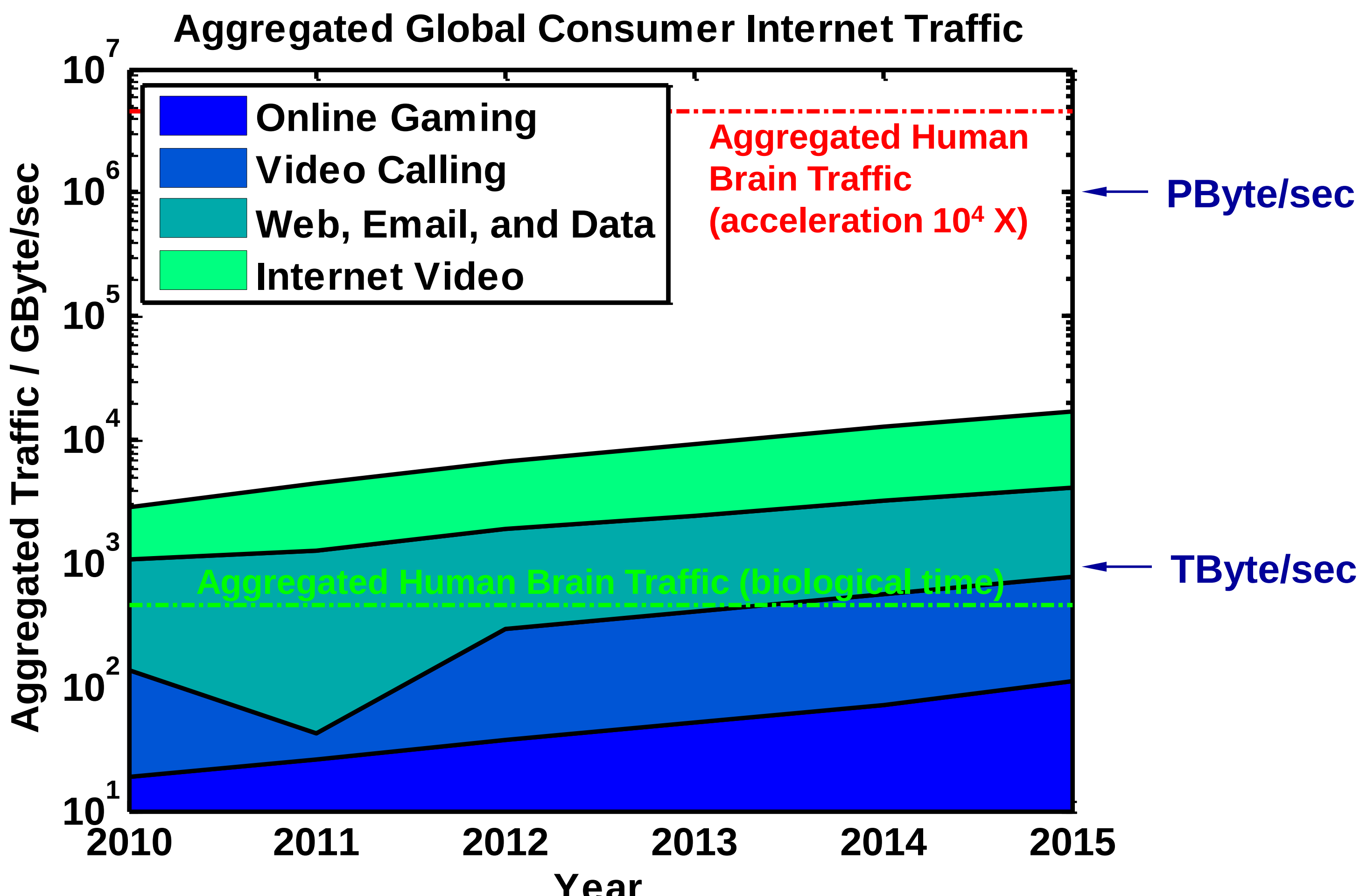
2) Actually, this is the pitch of the 1x-pitch layer(s), only

π_{req} = 0.852 nm

- **Even the interconnect density predicted for the future 5-nm CMOS node (possibly the end of scaling) will not match what would be required to mimic biological interconnect**

Limits to Speed-Up Communication

- **Human Brain Data Traffic vs. Global Consumer Internet Traffic**
  - **Human Brain**: Total neuron count: 10^{11}
    Avg. firing rate: 1Hz
    Source AER coding: $\lceil \log_2(10^{11}) \rceil = 37$ bits/spike
    Aggregated traffic: 10^{11} x 1Hz x 37bit = 0.4625 TByte/sec



Limits to Speed-Up Communication

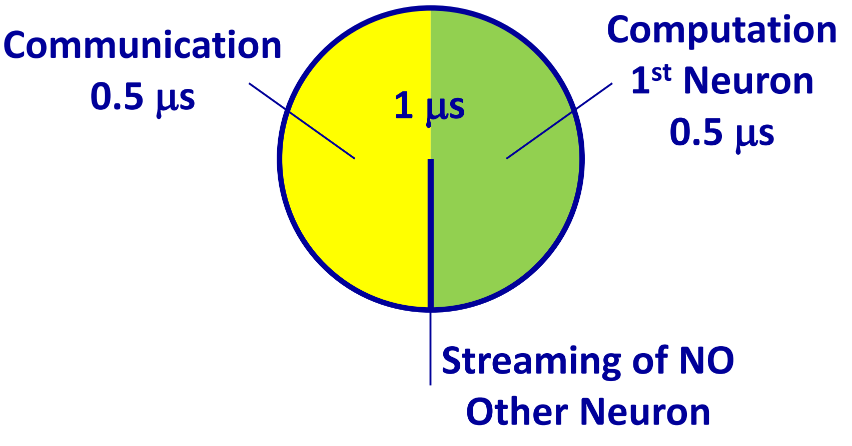
A Simple „*Gedankenexperiment*“

- **Assume:**
  - **Required Biological Real-Time Simulation Step Size (i.e., min. axon delay)**

    0.1 ms
  - **Intended Speed-Up Factor**

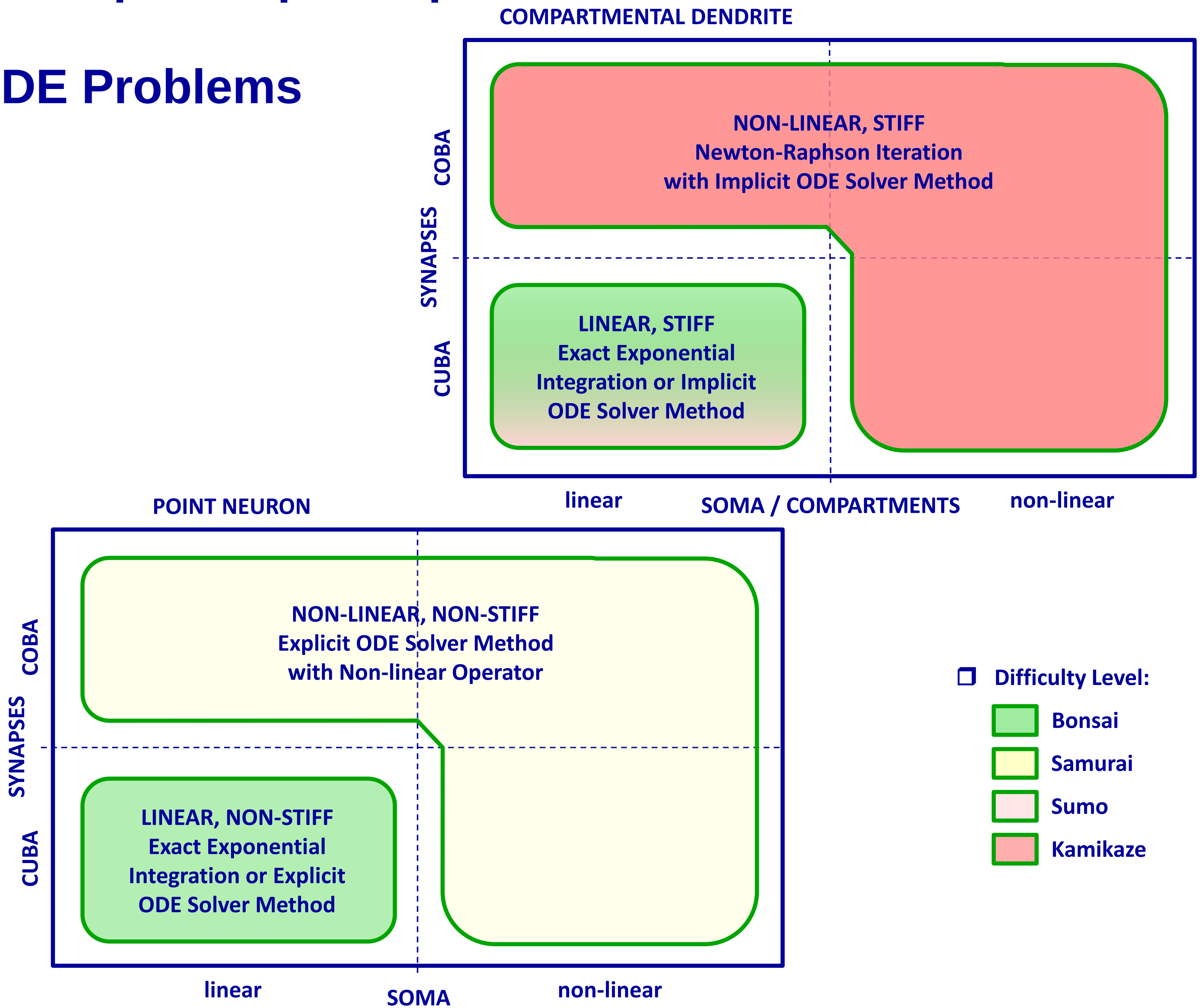
    x 100
  - ⇨ **Wall-Clock Time Simulation Step Size**

    1 μs
  - **„Fair“ 50-50% Allocation:**



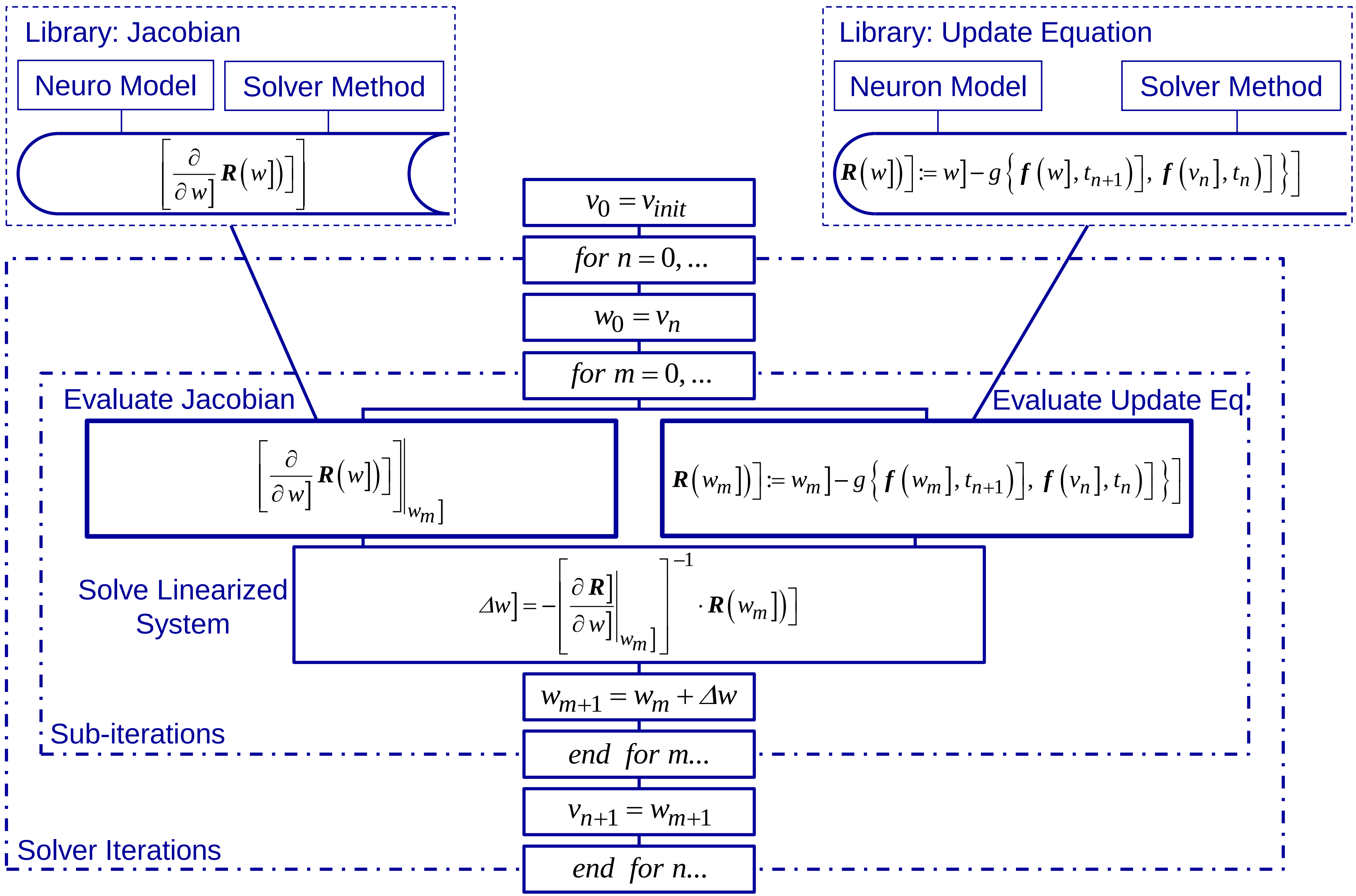
Limits to Speed-Up Computation

ODE Problems



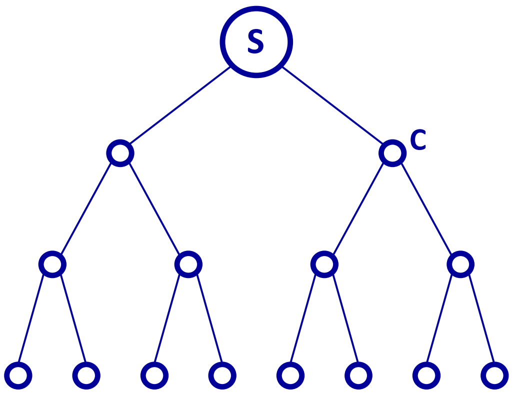
Limits to Speed-Up Computation

❒ **Newton-Raphson-based <u>Implicit</u> Single-step Solver** ❍ <u>**NL**</u> ODE System: $v' = \boldsymbol{f}(v,t)$



Limits to Speed-Up Computation

Single Neuron with Compartmental Dendrite



- ❒ Assume:
  - ❍ 14 Passive Compartments
  - ❍ 2 Lumped COBA Synapses per Compartment; (which can be integrated autonomously)
  - ❍ 2 Soma Equations

⇨ 16 x 16 Equation System

- ❒ Critical Path Length LU-based Linear Equation Solver w./o. any pivoting
  - ❍ Massively Parallel Solver / Maximally Sparse Tridiagonal Matrix
  - ❍ Without Pivoting and any „Bells and Whistles“ …

$\approx 3 \text{ x } n$ x (FDIV+FMAC) ≈ 50 x (FDIV+FMAC)

- ❒ Assume:
  - ❍ ≈ 10 Cycles per Dependant (FDIV+FMAC) Instructions

    ≈ 500 Cycles

  - ❍ 5-GHz Machine with 0.2-ns Clock Cycle

    ≈ 100 ns

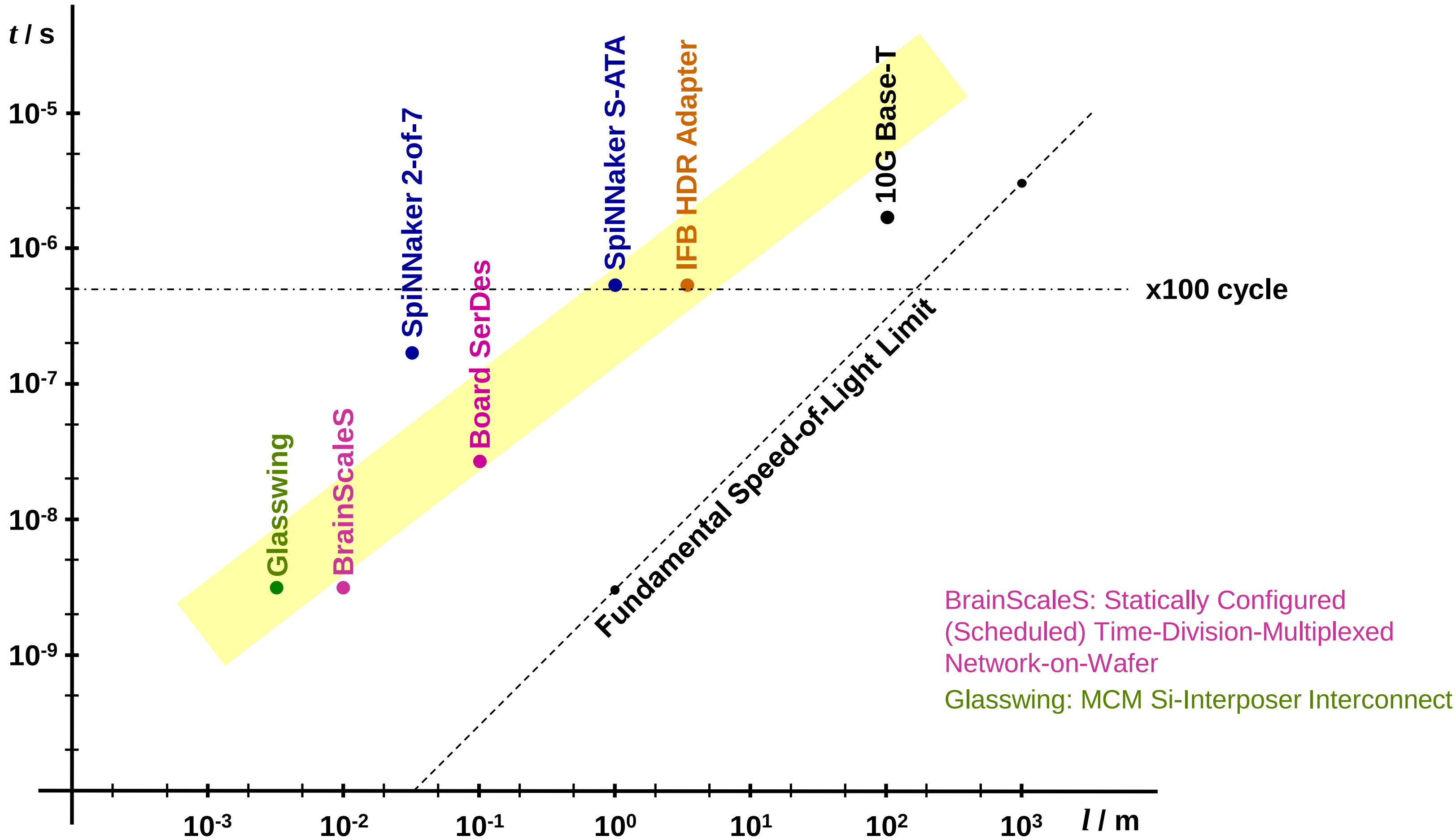
  - ❍ 5 Newton-Raphson Iterations

    ≈ 500 ns

Limits to Speed-Up Communication

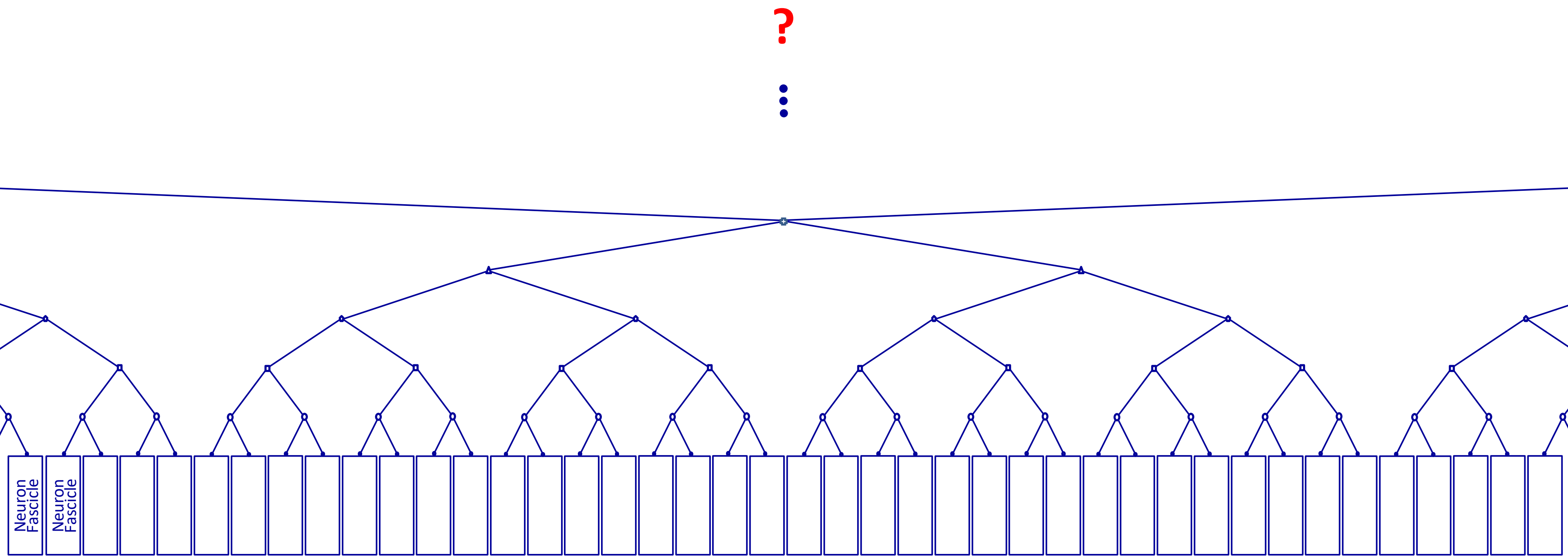
Half Round-Trip Time vs Link Distance

1-hop latency (1^{st}-bit-to-1^{st}-bit) i.e., one packet over one hop



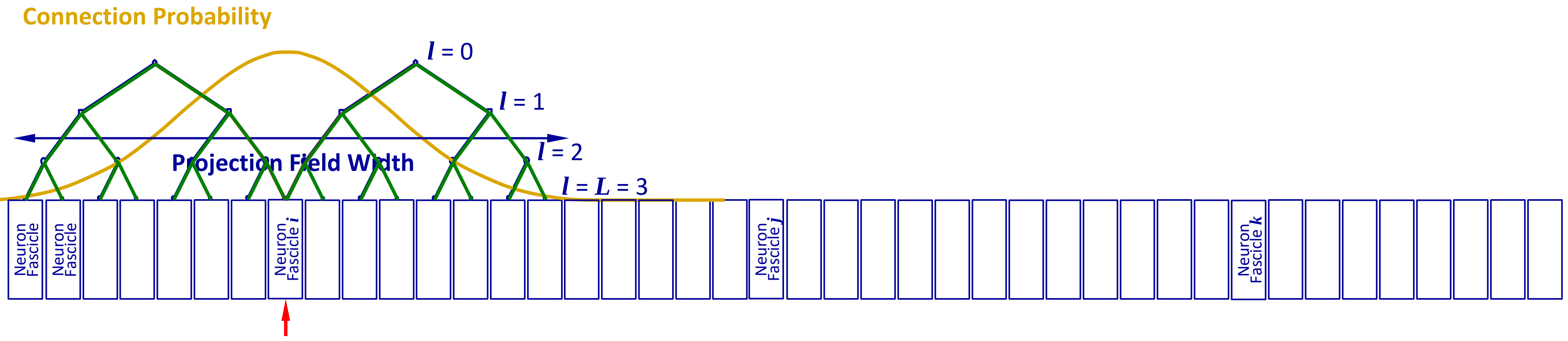
- ❒ **Latency vs. distance dependency transforms speed-up challenge into a integration-density challenge**

Section of Large One-Dimensional Array with Tree Topology



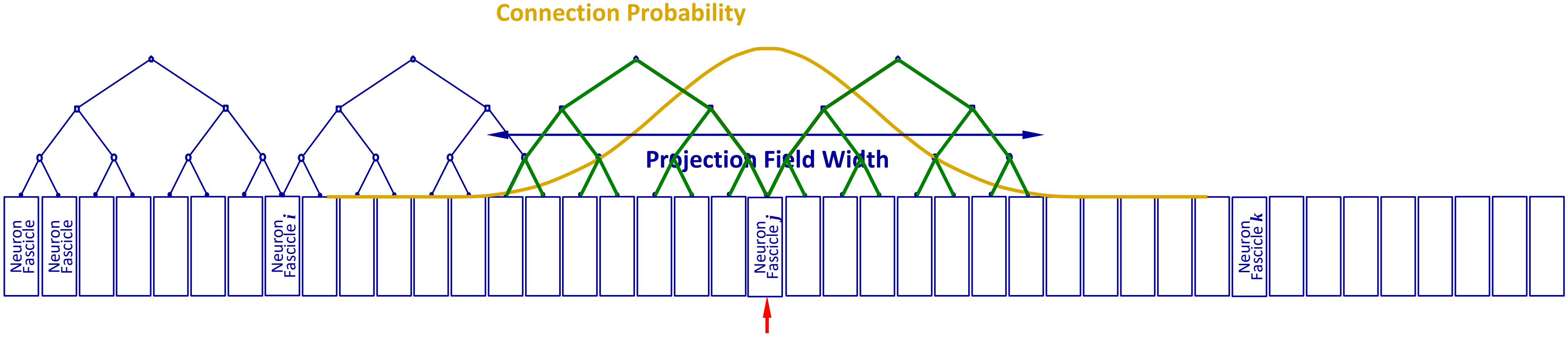
Section of Large One-Dimensional Array with Tree Topology

- ❒ **Localized Connection Probability limits "Projection Field": Need "Local Broadcast"**
  - ❍ **Use Pair of Limited-Order Trees**



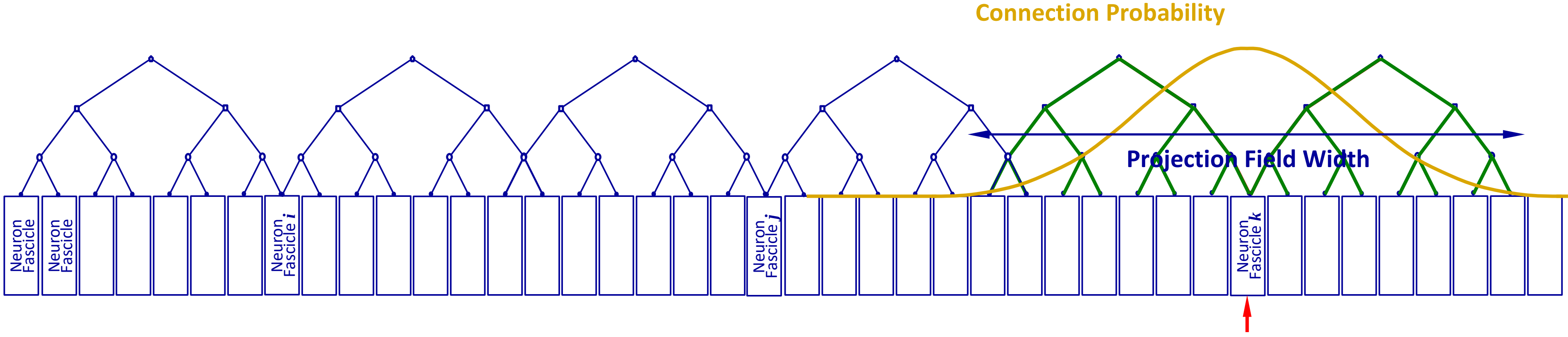
Section of Large One-Dimensional Array with Tree Topology

- ❒ **Localized Connection Probability limits "Projection Field": Need "Local Broadcast"**
  - ❍ **Use Pair of Limited-Order Trees**



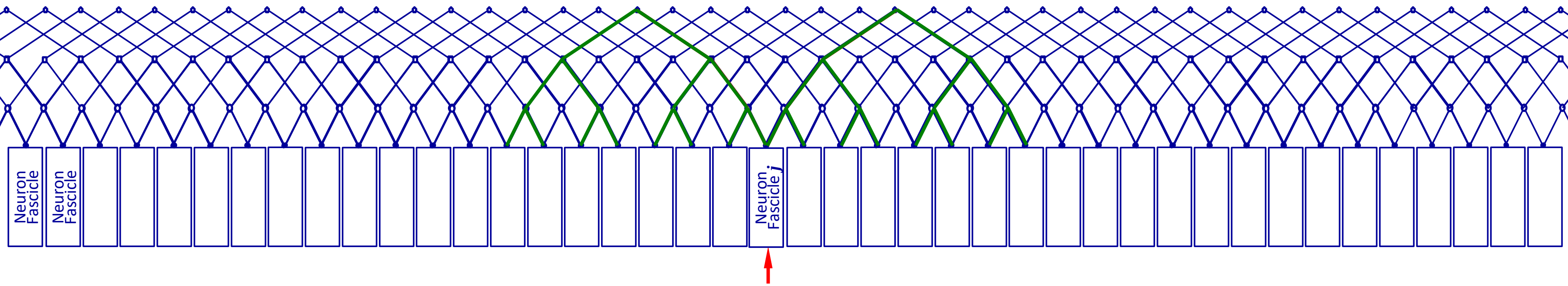
Section of Large One-Dimensional Array with Tree Topology

- ❒ **Localized Connection Probability limits "Projection Field": Need "Local Broadcast"**
  - ❍ **Use Pair of Limited-Order Trees**



Section of Large One-Dimensional Array with Tree Topology

- ❒ **Localized Connection Probability limits "Projection Field": Need "Local Broadcast"**
  - ❍ **Use Pair of Limited-Order Trees**

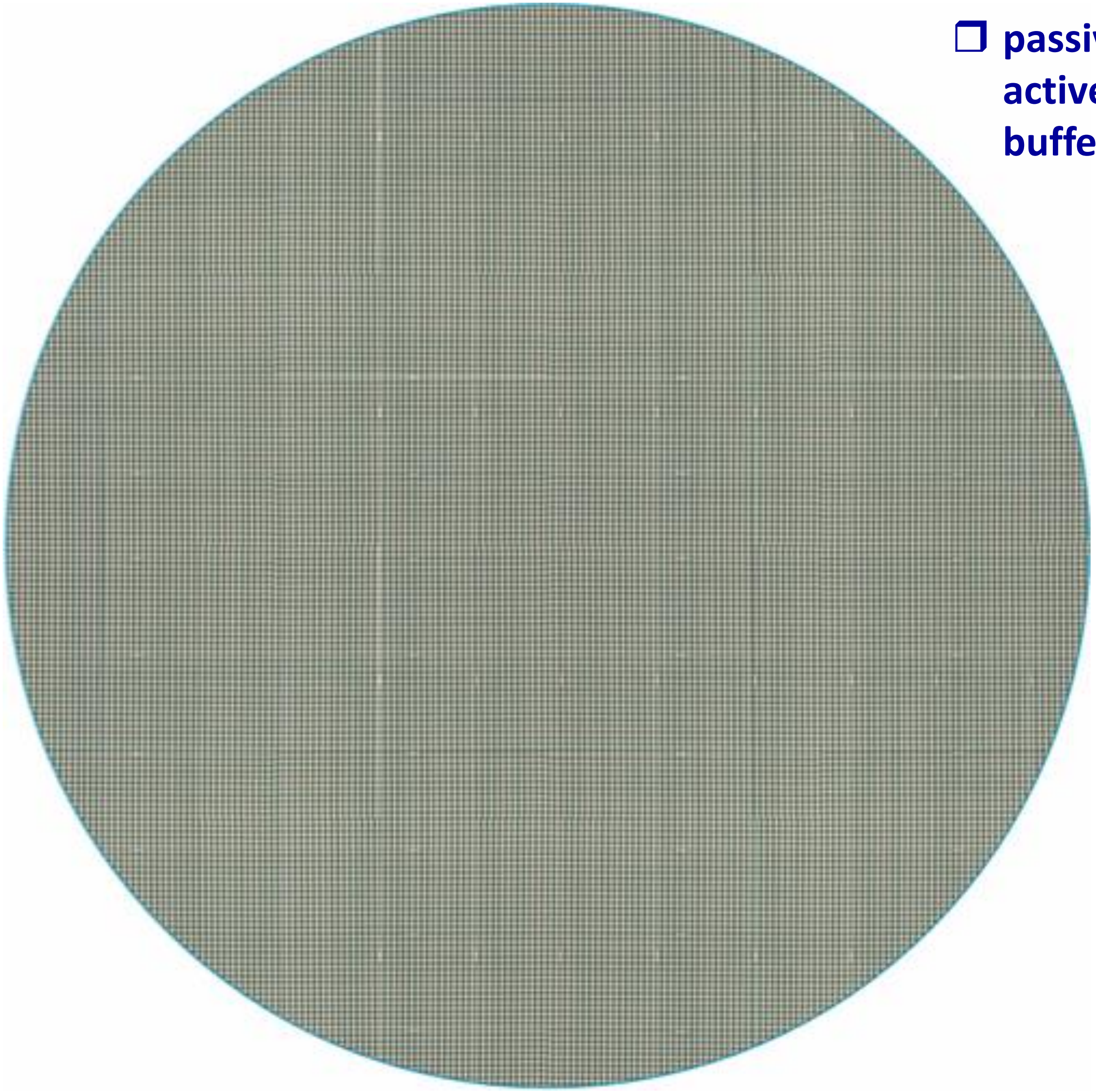


- ☑ Massively Parallel
- ☑ Ultra-Scalable
- ☑ Highly-Regular Interconnect
- ☑ Very Simple Route Protocol

Uplink: Unicast straight up, straight up, …

Downlink: Multicast down, down, …

Ultra-Large (Wafer-Scale) Silicon Interposer



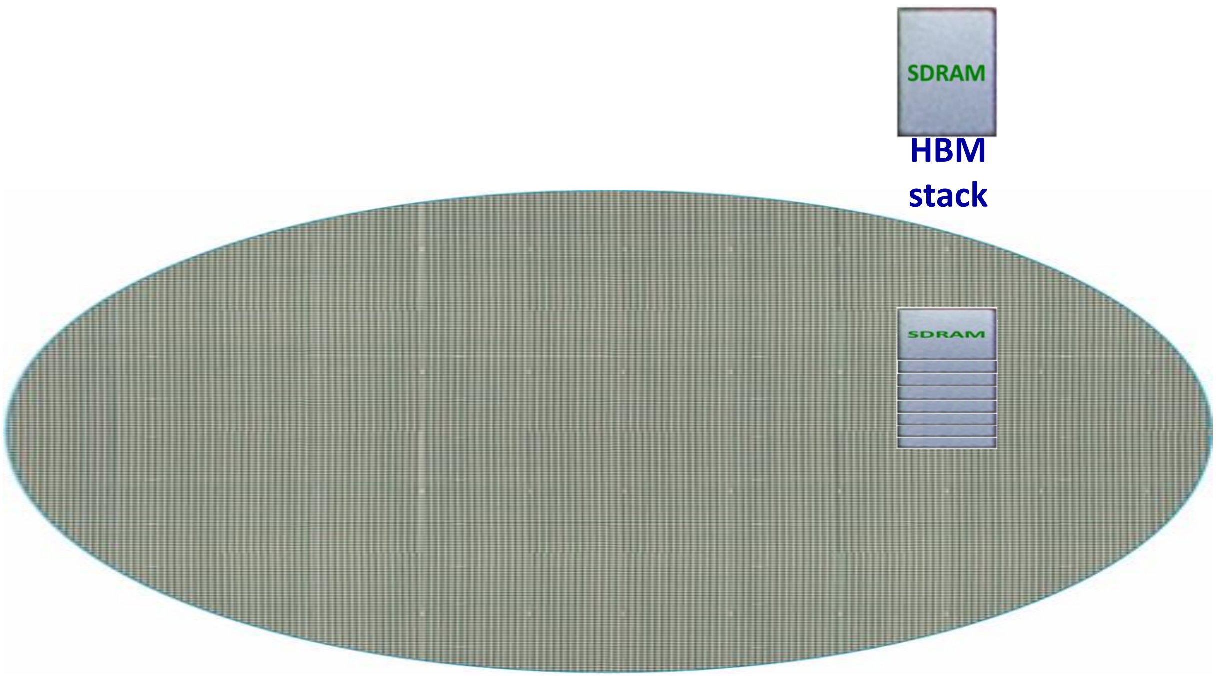
- passive / may be with active interconnect buffers

- First coarse-pitch interconnect metal layer
- Further coarse-pitch interconnect metal layer(s)

Flip-Chip-Die-Attach Memory Devices

Commodity (off-the-shelf devices)

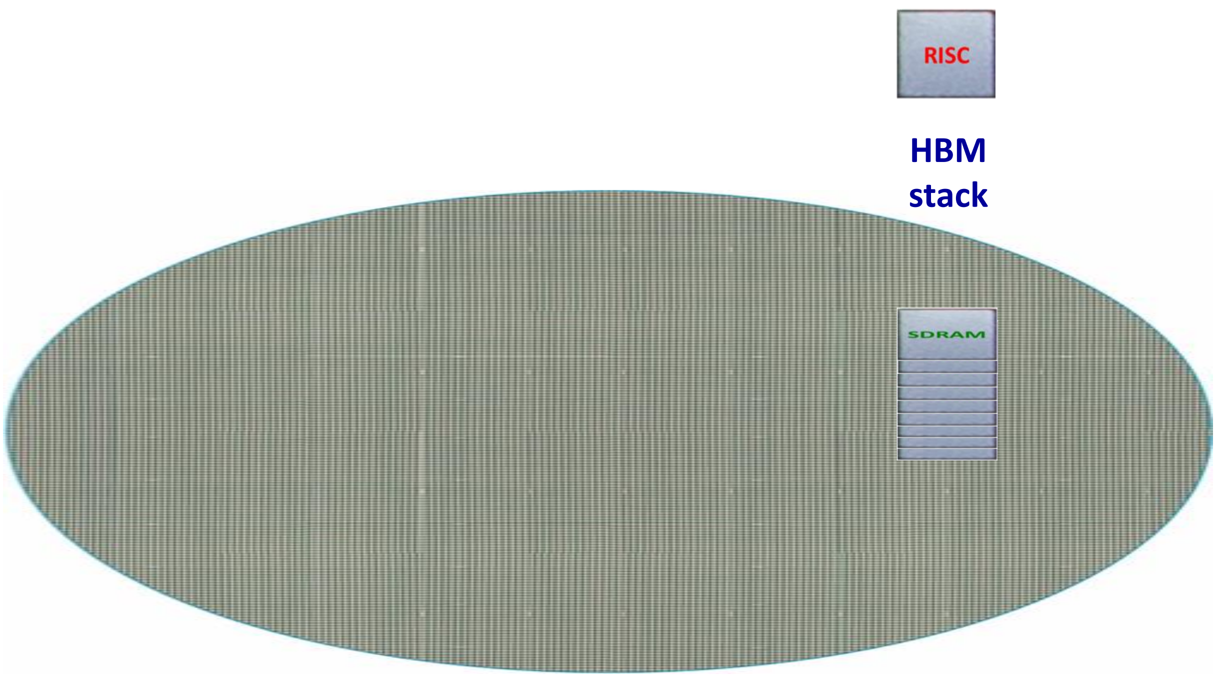
- Most advanced / best fitting technologies
- Relatively cheap
- Fully functional



Flip-Chip-Die-Attach Logic Devices (2.5-D Integration)

Commodity (off-the-shelf devices)

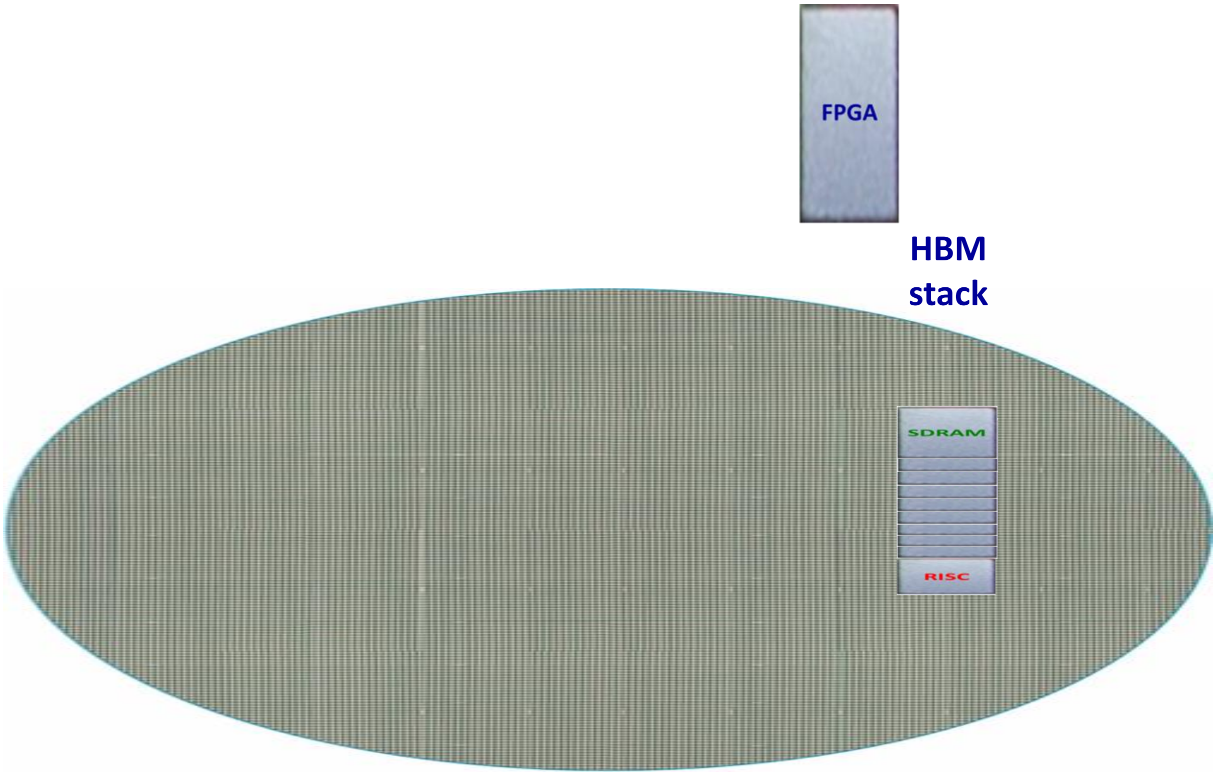
- Most advanced / best fitting technologies
- Relatively cheap
- Fully functional



Flip-Chip-Die-Attach Logic Devices (2.5-D Integration)

Commodity (off-the-shelf devices)

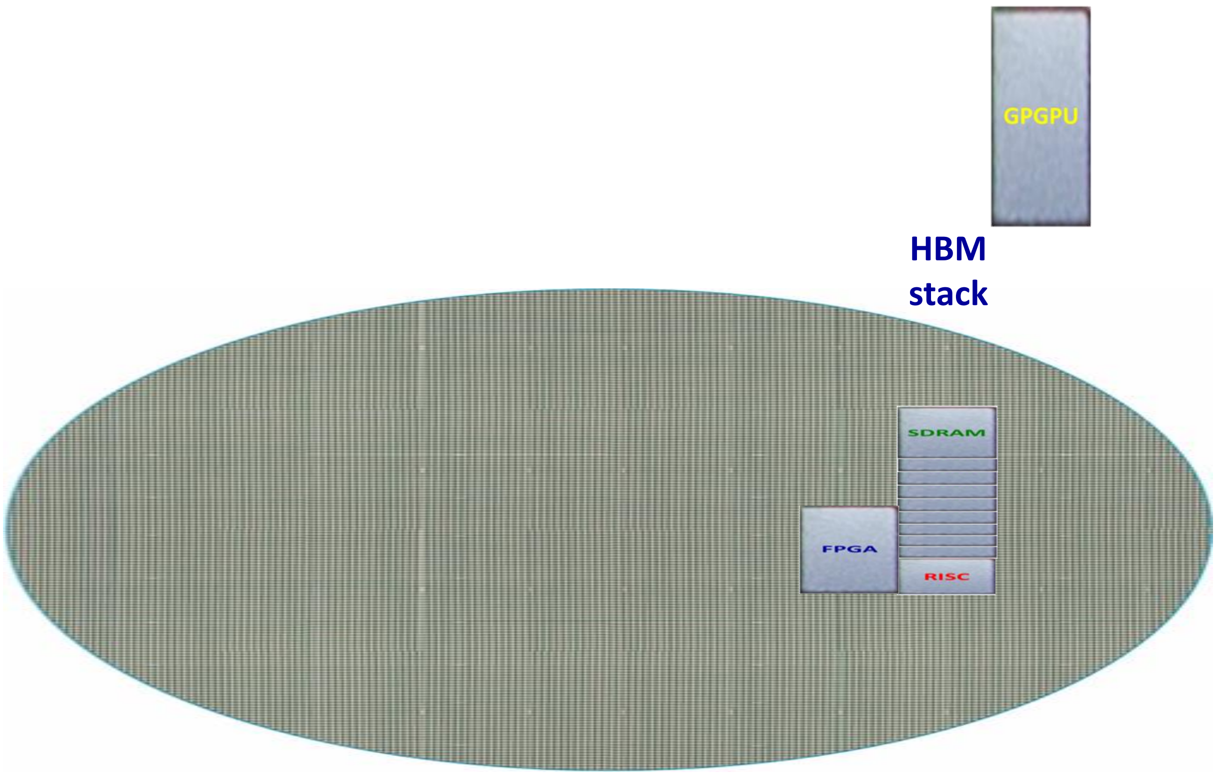
- Most advanced / best fitting technologies
- Relatively cheap
- Fully functional



Flip-Chip-Die-Attach Logic Devices (2.5-D Integration)

Commodity (off-the-shelf devices)

- Most advanced / best fitting technologies
- Relatively cheap
- Fully functional



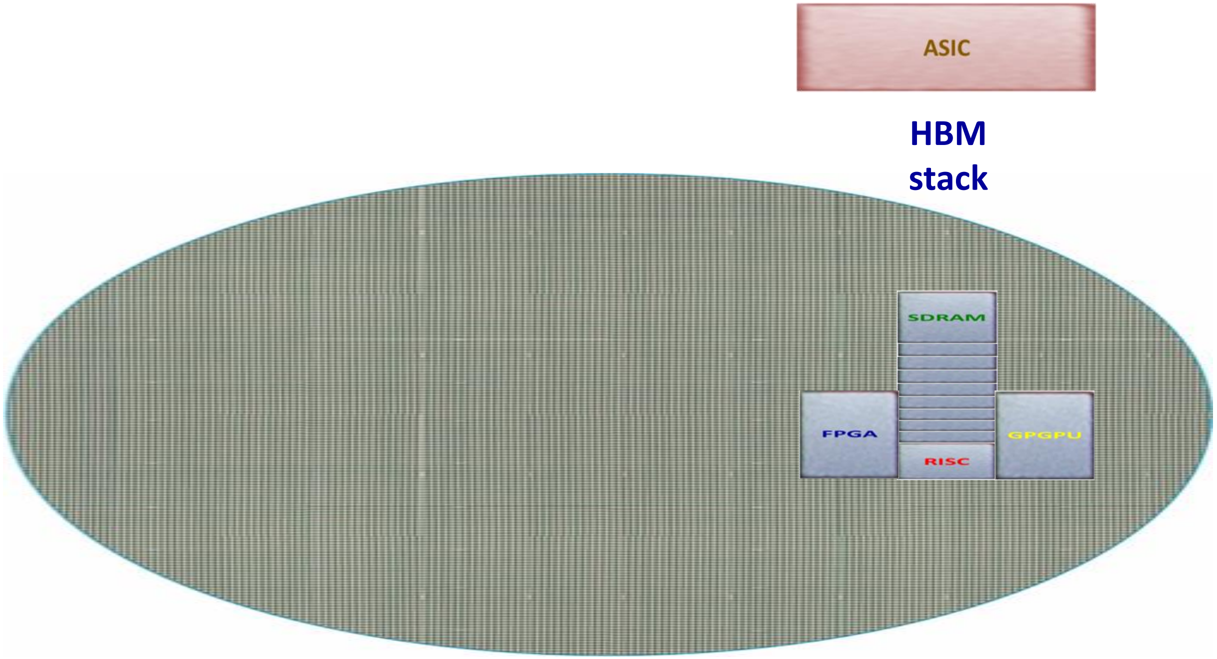
Flip-Chip-Die-Attach Logic Devices (2.5-D Integration)

Commodity (off-the-shelf) devices

- Most advanced / best fitting technologies
- Relatively cheap
- Fully functional

Proprietary dedicated devices

- Moderate technologies
- Relatively cheap



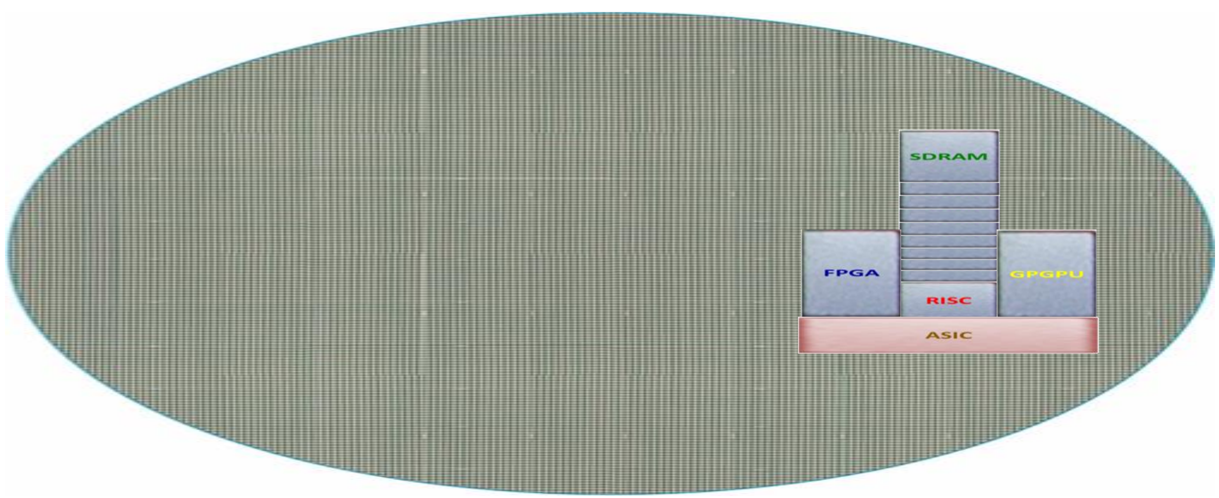
Single Hybrid Compute Node

Commodity (off-the-shelf) devices

- Most advanced / best fitting technologies
- Relatively cheap
- Fully functional

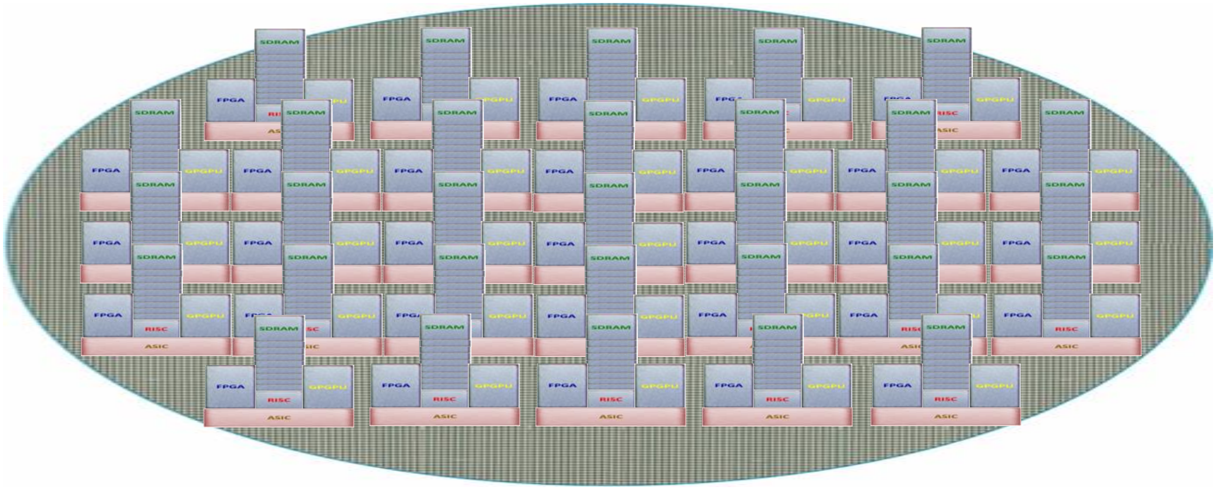
Proprietary devices

- Moderate technologies
- Relatively cheap
- Fully functional



Integrate Many Hybrid Compute Nodes

- **Challenges:**
  - **Mechanical stability**
  - **Thermal stability**



Outline

- Introduction
- Neuronal Network Components and Models
- Exemplary Large-Scale Networks
- Future Requirements
- Simulation Principles
  - Computation: Numerical ODE Solvers
  - Communication: AER
- State-of-the Art
- Brick Walls
  - Computation: …
  - Communication: …
- **Conclusion**

Conclusions

- **Requirements Future Neuroscience Simulation Platforms**
- **State-of-the Art not sufficient – by far**
- **Very challenging in today‘s technologies**
- **Some „brick walls“ ahead …**

Thank you very much for you kind attention !