

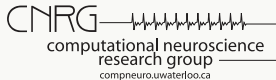
Nengo Summerschool 2019

Biologically Detailed Networks and Neuron Models

Peter Duggins, Andreas Stöckel



UNIVERSITY OF
WATERLOO



June 18, 2018

OUTLINE



I *Motivation & Background*

Why care about biological detail?

OUTLINE



I *Motivation & Background*

Why care about biological detail?

II *Conductance-based n -LIF neurons* ← Andreas

How to computationally exploit nonlinear dendritic interaction

OUTLINE



I *Motivation & Background*

Why care about biological detail?

II *Conductance-based n -LIF neurons* ← Andreas

How to computationally exploit nonlinear dendritic interaction

III *Detailed NEURON Models and the NEF* ← Peter

How to integrate detailed neuron models into the NEF

OUTLINE



I *Motivation & Background*

Why care about biological detail?

II *Conductance-based n -LIF neurons* ← Andreas

How to computationally exploit nonlinear dendritic interaction

III *Detailed NEURON Models and the NEF* ← Peter

How to integrate detailed neuron models into the NEF

IV *nengo-bio hands-on* ← Andreas

Build networks adhering to Dale's principle and exploit dendritic computation

PART I

Motivation & Background

Motivation — *Levels of Analysis*



Motivation — *Levels of Analysis*



Motivation — *Levels of Analysis*



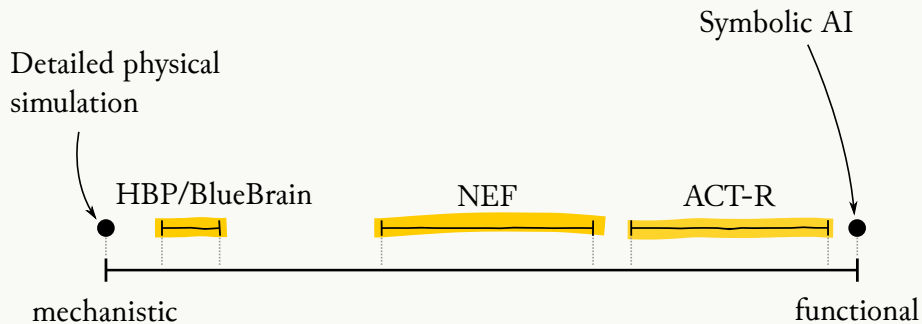
Motivation — *Levels of Analysis*



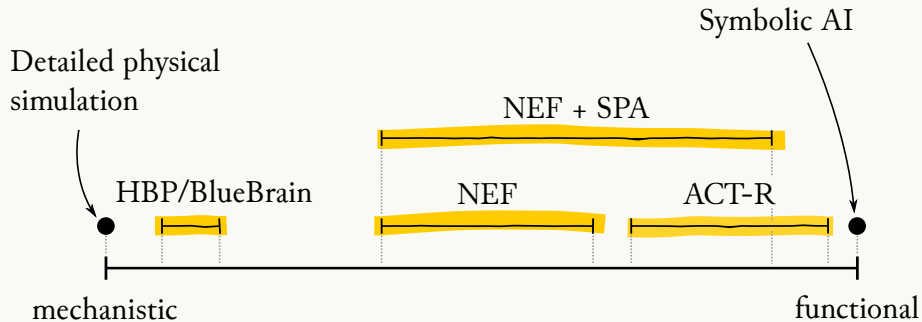
Motivation — *Levels of Analysis*



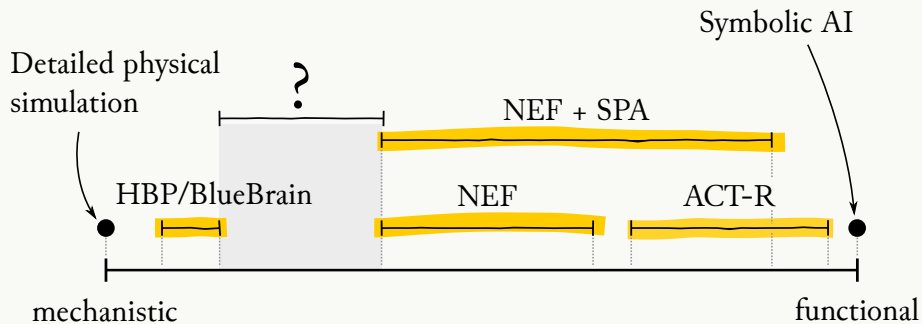
Motivation — *Levels of Analysis*



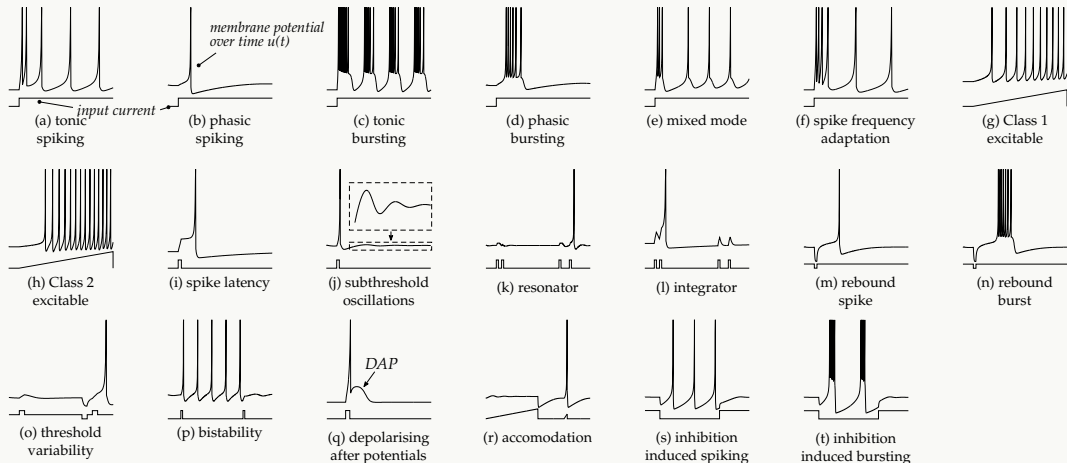
Motivation — *Levels of Analysis*



Motivation — *Levels of Analysis*

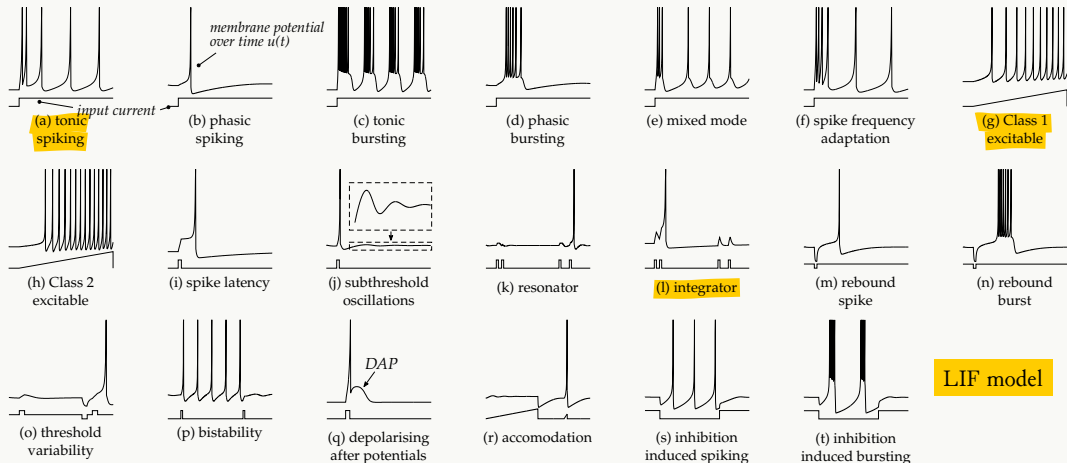


Motivation — *Computational Power of Biological Neurons*



Adapted from Izhikevich (2004), *Which Model to Use for Cortical Spiking Neurons?*

Motivation — *Computational Power of Biological Neurons*



Adapted from Izhikevich (2004), *Which Model to Use for Cortical Spiking Neurons?*

Motivation — *Computational Power of Biological Neurons*

- LIF model accounts for a fraction of the behaviour found in biological neurons

Motivation — *Computational Power of Biological Neurons*

- LIF model accounts for a fraction of the behaviour found in biological neurons
- **Questions:**
 - In how far is the additional detail functionally *relevant*?

Motivation — *Computational Power of Biological Neurons*

- LIF model accounts for a fraction of the behaviour found in biological neurons
- **Questions:**
 - In how far is the additional detail functionally *relevant*?
 - What is its role in terms of *computation*?


Motivation — *Computational Power of Biological Neurons*

- LIF model accounts for a fraction of the behaviour found in biological neurons
- **Questions:**
 - In how far is the additional detail functionally *relevant*?
 - What is its role in terms of *computation*?
 - Can we *harness* these details to perform computation?

Motivation — *Computational Power of Biological Neurons*

- LIF model accounts for a fraction of the behaviour found in biological neurons
- **Questions:**
 - In how far is the additional detail functionally *relevant*?
 - What is its role in terms of *computation*?
 - Can we *harness* these details to perform computation?
- **Example:**
 - Excitatory and inhibitory channels interact nonlinearly.

Motivation — *Computational Power of Biological Neurons*

- LIF model accounts for a fraction of the behaviour found in biological neurons
- **Questions:**
 - In how far is the additional detail functionally *relevant*?
 - What is its role in terms of *computation*?
 - Can we *harness* these details to perform computation?
- **Example:**
 - Excitatory and inhibitory channels interact nonlinearly.
 -  Can we exploit this nonlinearity systematically?

Motivation — *Recreating Idiosyncrasies of the Brain*

① Study Perturbations

- ▶ Drugs
- ▶ Neurological/mental disorders



Motivation — *Recreating Idiosyncrasies of the Brain*

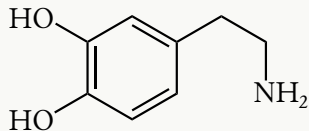
① Study Perturbations

- ▶ Drugs
- ▶ Neurological/mental disorders



② Neuromodulation

- ▶ Model additional pathways for information processing/learning



DOPAMINE

Motivation — *Recreating Idiosyncrasies of the Brain*

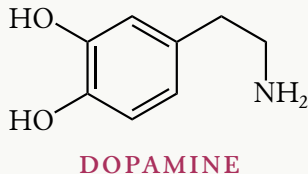
① Study Perturbations

- ▶ Drugs
- ▶ Neurological/mental disorders



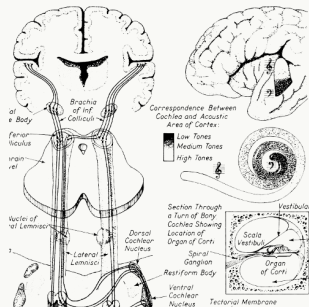
② Neuromodulation

- ▶ Model additional pathways for information processing/learning

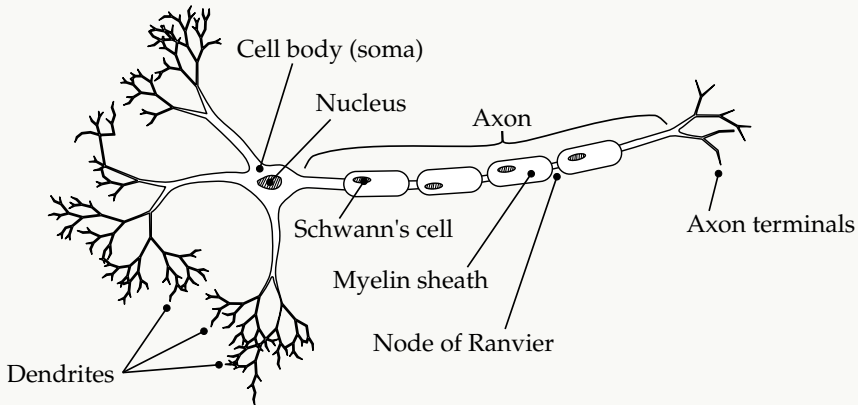


③ Constrain Models

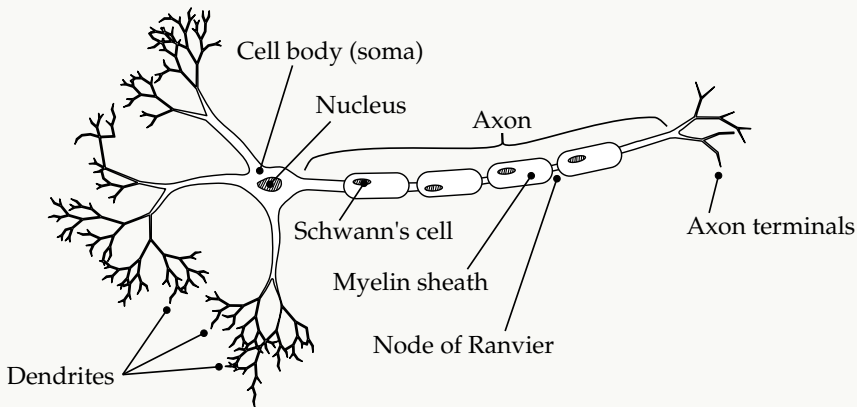
- ▶ Constrain models to available biophysical data



Neurobiology — *Idealized “Textbook” Neuron*

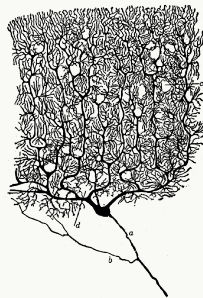
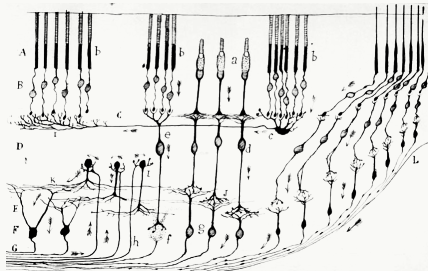
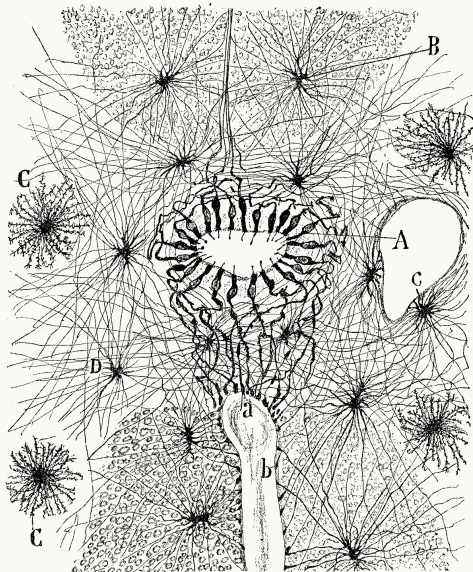


Neurobiology — *Idealized “Textbook” Neuron*



❶ Dendrites collect input → ❷ Integrated in soma → ❸ Output spikes travel along axon

Neurobiology — *Neural Heterogeneity*



Drawings by
Santiago Ramón y Cajal

Comparing LIF and Biology

	COMPLEXITY			NEF COMPATIBILITY	
	<i>LIF</i>	<i>BIO</i>		<i>LIF</i>	<i>BIO</i>
Geometry	point	compartmental	Tuning curve	$A = f(\mathbf{x})$	$A = f(\mathbf{x}, t)$
Synapse	current	conductance	Inputs	linear filter	synaptic nonlinearity
Dynamics	integrate, leak	ion channels	Dynamics	synapse dominates	neuron dominates
	voltage reset	cable equation	Decoders	static	time-dependent
Simulation	fast	slow	Estimates	$\sum_j a_i(t) * \mathbf{d}_i^f$?

PART II

Conductance-based n -LIF neurons

Reversal potentials

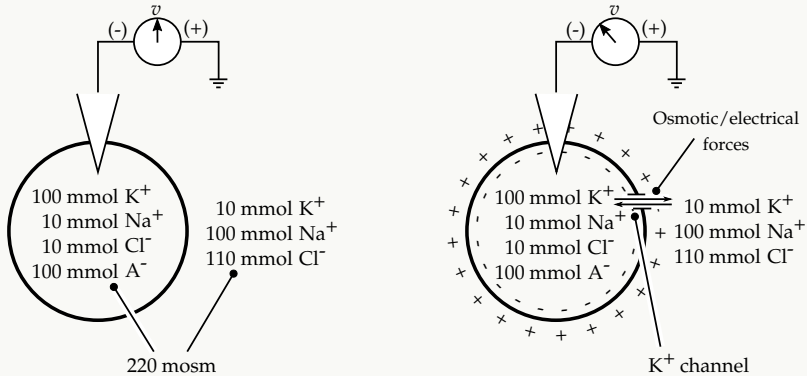
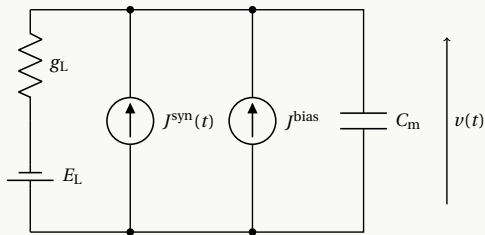


Illustration adapted from Reichert, 2000, Neurobiology.

Ion channels possess a specific *reversal potential* corresponding to the combination of *ion species* they are permeable for.

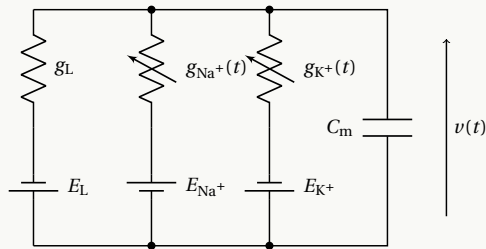
Current vs. Conductance-Based LIF

Current-based LIF



$$\begin{aligned} C_m \dot{u}(t) = & J^{\text{bias}} \\ & + \alpha J^{\text{syn}}(t) \\ & + g_L (E_L - u(t)) \end{aligned}$$

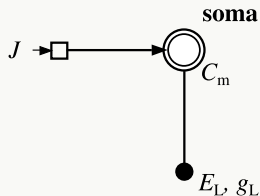
Conductance-based LIF



$$\begin{aligned} C_m \dot{u}(t) = & g_E(t) (E_E - u(t)) \\ & + g_I(t) (E_I - u(t)) \\ & + g_L (E_L - u(t)) \end{aligned}$$

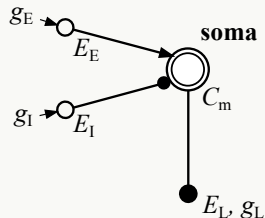
Current vs. Conductance-Based LIF

Current-based LIF



$$\begin{aligned} C_m \dot{u}(t) = & J^{\text{bias}} \\ & + \alpha J^{\text{syn}}(t) \\ & + g_L (E_L - u(t)) \end{aligned}$$

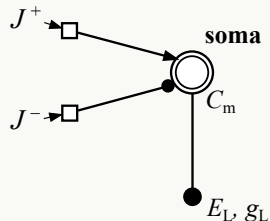
Conductance-based LIF



$$\begin{aligned} C_m \dot{u}(t) = & g_E(t) (E_E - u(t)) \\ & + g_I(t) (E_I - u(t)) \\ & + g_L (E_L - u(t)) \end{aligned}$$

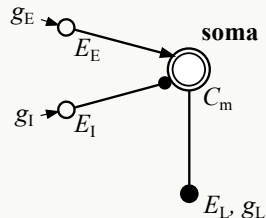
Current vs. Conductance-Based LIF

Current-based LIF



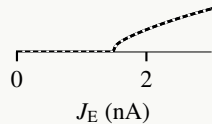
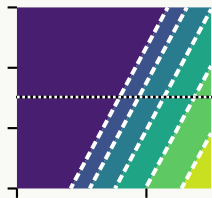
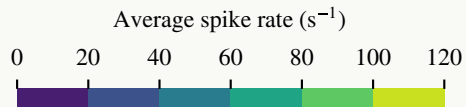
$$\begin{aligned} C_m \dot{u}(t) = & J^{\text{bias}} \\ & + \alpha (J^+(t) - J^-(t)) \\ & + g_L (E_L - u(t)) \end{aligned}$$

Conductance-based LIF

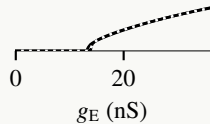
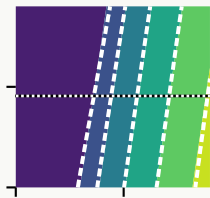


$$\begin{aligned} C_m \dot{u}(t) = & g_E(t) (E_E - u(t)) \\ & + g_I(t) (E_I - u(t)) \\ & + g_L (E_L - u(t)) \end{aligned}$$

Current vs. Conductance-Based LIF Tuning Curves



(a) Current-based



(b) Conductance-based

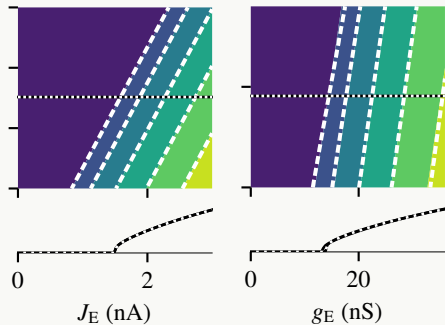
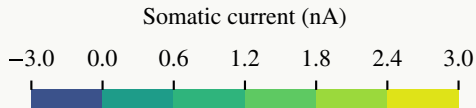
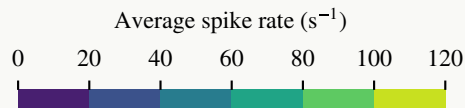


Model fit



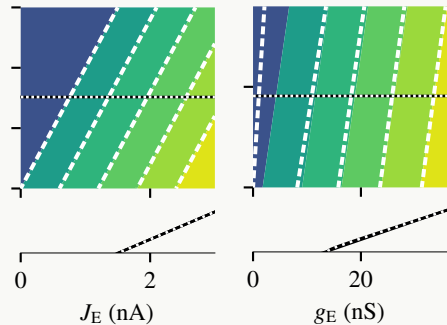
Numerical simulation

Current vs. Conductance-Based LIF Tuning Curves



(a) Current-based

(b) Conductance-based



(a) Current-based

(b) Conductance-based

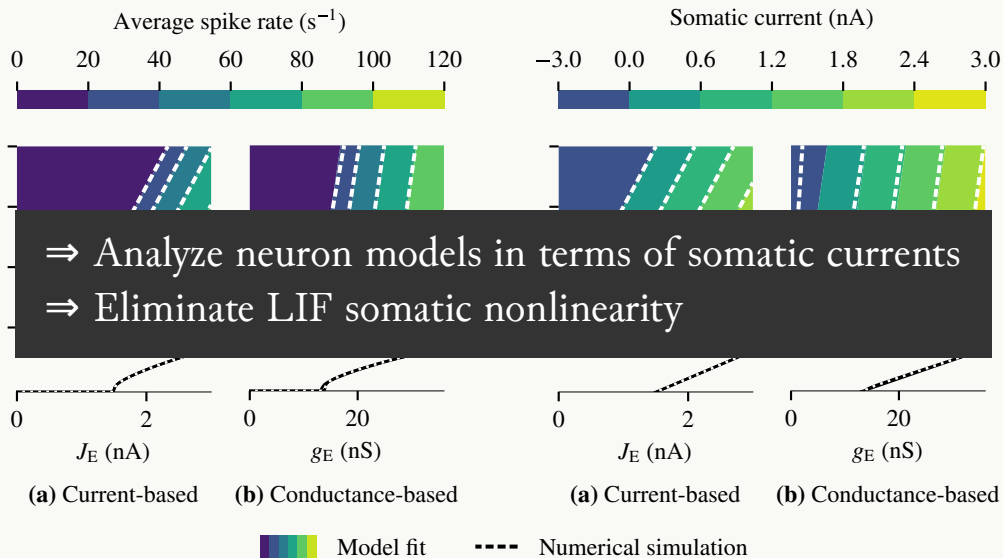


Model fit



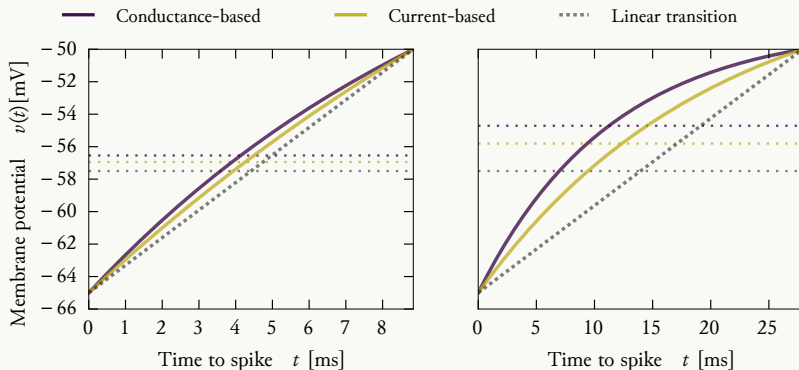
Numerical simulation

Current vs. Conductance-Based LIF Tuning Curves



Single-compartment conductance-based LIF

- For firing rates $\gg 0$, conductance-based LIF is boring

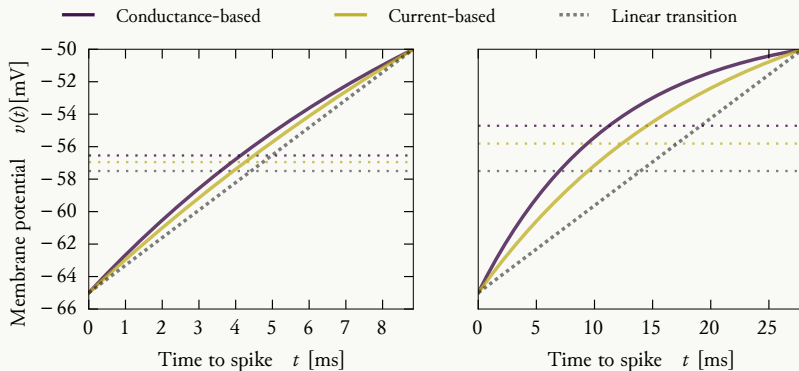


- Can assume average membrane potential \bar{u}

$$C_m \dot{u}(t) = g_E(t)(E_E - u(t)) + g_I(t)(E_I - u(t)) + g_L(E_L - u(t))$$

Single-compartment conductance-based LIF

- For firing rates $\gg 0$, conductance-based LIF is boring

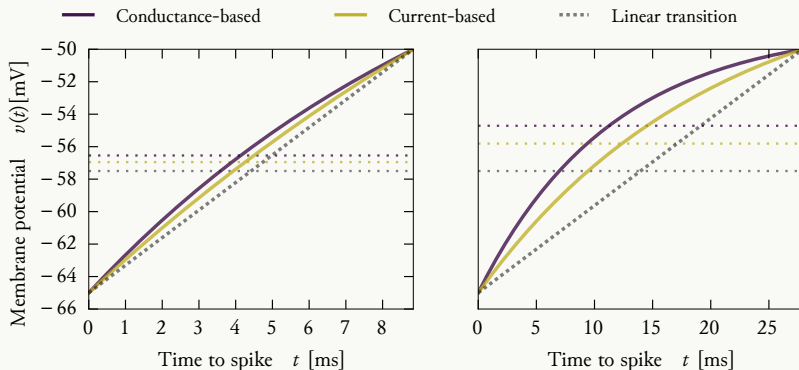


- Can assume average membrane potential \bar{u}

$$C_m \dot{u}(t) = g_E(t)(E_E - \bar{u}) + g_I(t)(E_I - \bar{u}) + g_L(E_L - u(t))$$

Single-compartment conductance-based LIF

- For firing rates $\gg 0$, conductance-based LIF is boring



- Can assume average membrane potential $\bar{u} \Rightarrow$ just a skewed current-based LIF!

$$C_m \dot{u}(t) = g_E(t)\alpha_E + g_I(t)\alpha_I + g_L(E_L - u(t))$$

Challenges and Extending the NEF

- **Challenges**

Challenges and Extending the NEF

- **Challenges**

- ① NEF assumes a constant *bias current*

- ⇒ Reinterpretation of tuning curves; optimize weights in *current-space*

Challenges and Extending the NEF

- **Challenges**

- 1 NEF assumes a constant *bias current*
⇒ Reinterpretation of tuning curves; optimize weights in *current-space*
- 2 NEF does not distinguish between *excitatory and inhibitory weights*
⇒ Solve *non-negative* optimization problems

Challenges and Extending the NEF

- **Challenges**

- ① NEF assumes a constant *bias current*
⇒ Reinterpretation of tuning curves; optimize weights in *current-space*
- ② NEF does not distinguish between *excitatory and inhibitory weights*
⇒ Solve *non-negative* optimization problems

- **Extensions**

Challenges and Extending the NEF

- **Challenges**

- ① NEF assumes a constant *bias current*
⇒ Reinterpretation of tuning curves; optimize weights in *current-space*
- ② NEF does not distinguish between *excitatory and inhibitory weights*
⇒ Solve *non-negative* optimization problems

- **Extensions**

- ③ Account for *sub-threshold currents* in the optimization process

Challenges and Extending the NEF

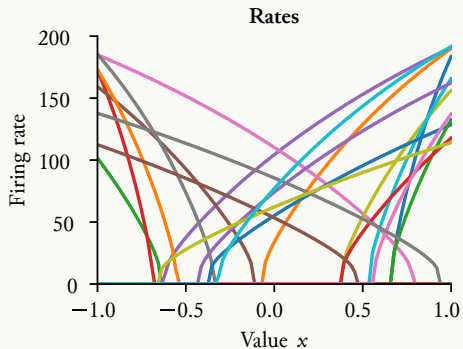
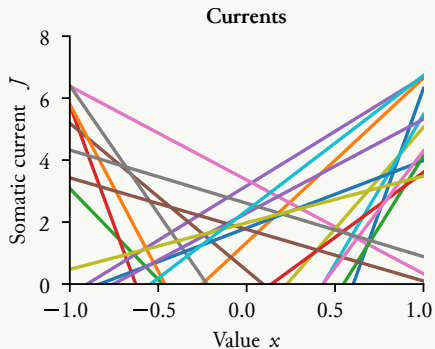
- **Challenges**

- ① NEF assumes a constant *bias current*
⇒ Reinterpretation of tuning curves; optimize weights in *current-space*
- ② NEF does not distinguish between *excitatory and inhibitory weights*
⇒ Solve *non-negative* optimization problems

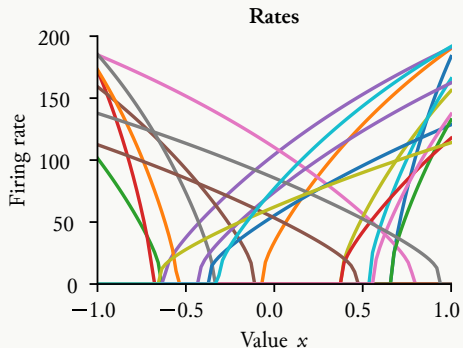
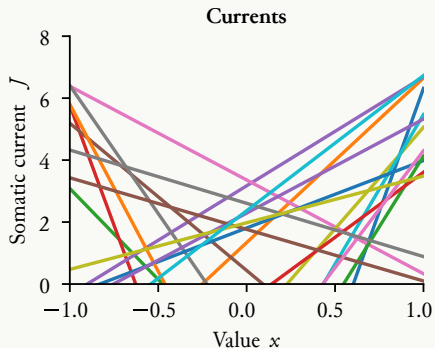
- **Extensions**

- ③ Account for *sub-threshold currents* in the optimization process
- ④ Take *dendritic non-linearity* into account

1 Eliminating the Bias Current

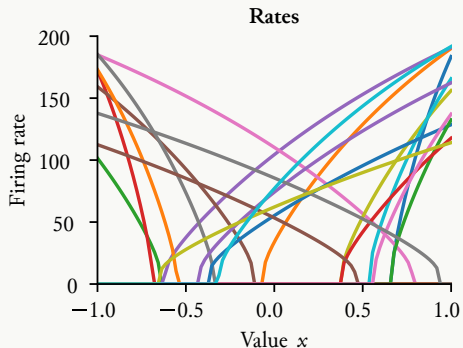
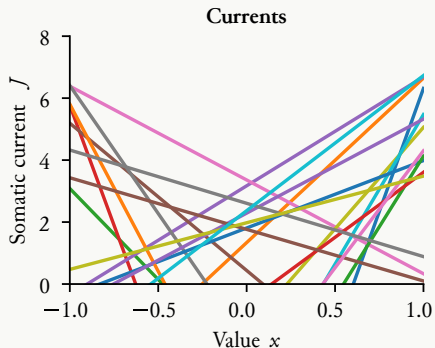


1 Eliminating the Bias Current



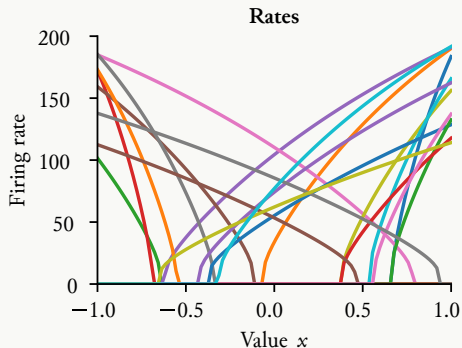
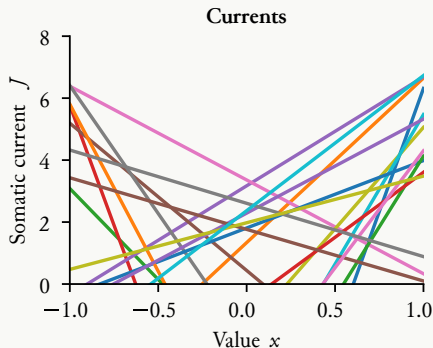
- Interpret tuning curve as normative: “We *want* firing rate a if the neuron represents x ”

1 Eliminating the Bias Current



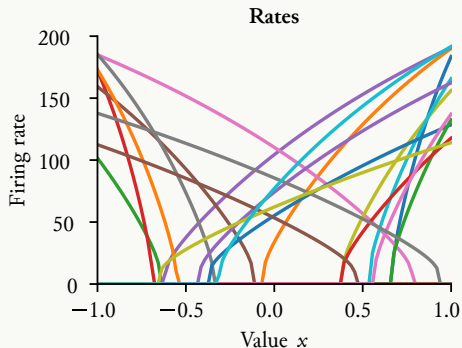
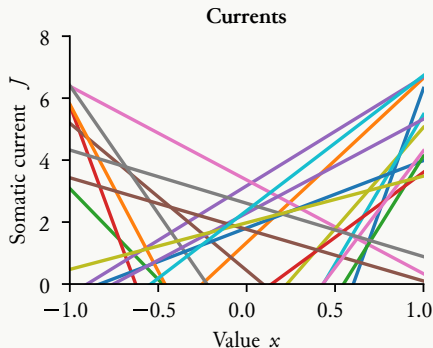
- Interpret tuning curve as normative: “We *want* firing rate a if the neuron represents x ”
- Firing rates correspond to a current J : “We *want* a current J if the neuron represents x ”

1 Eliminating the Bias Current



- Interpret tuning curve as normative: “We *want* firing rate a if the neuron represents x ”
 - Firing rates correspond to a current J : “We *want* a current J if the neuron represents x ”
- ⇒ Find a current-decoder for each individual post-neuron

1 Eliminating the Bias Current



- Interpret tuning curve as normative: “We *want* firing rate a if the neuron represents x ”
 - Firing rates correspond to a current J : “We *want* a current J if the neuron represents x ”
- ⇒ Find a current-decoder for **each individual post-neuron** (instead of population-wise)

2 Nonnegative Weight Optimization

- Assume a post-neuron receives both excitatory and inhibitory input from each pre-neuron $i \Rightarrow$ weight vectors \vec{w}_i^+ , \vec{w}_i^- .

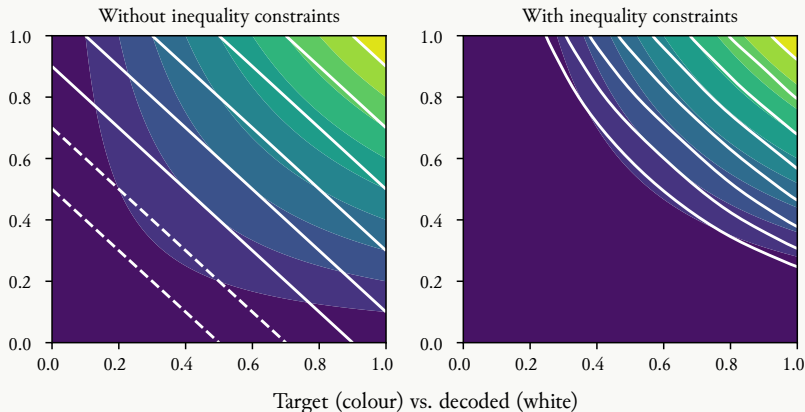
$$\begin{aligned} \min_{\vec{w}_i^+, \vec{w}_i^-} \quad & \frac{1}{2} \sum_{k=1}^N \left\| \vec{w}_i^+ \vec{a}_k^+ - \vec{w}_i^- \vec{a}_k^- - J(\langle \vec{e}_i, f(\vec{x}_k) \rangle) \right\|_2^2 \\ &= \frac{1}{2} \left\| \vec{w}'_i A' - \vec{j} \right\|_2^2 \quad \text{where } \vec{w}'_i = (\vec{w}_i^+, \vec{w}_i^-), A' = (A^+, -A^-)^T, \\ &\quad \text{and } (\vec{j})_k = J(\langle \vec{e}_i, f(\vec{x}_k) \rangle), \end{aligned}$$

subject to $\vec{w}_i^+ \geq 0, \vec{w}_i^- \geq 0$

- Can remove rows/columns from A' and \vec{w}_i to account for Dale's principle (e.g. only 20% of pre-neurons are inhibitory, 80% are excitatory)

3 Account for sub-threshold currents

- Tuning curves are not injective (one-to-one): multiple x map onto a zero output rate
- ⇒ If the target rate is zero, we do not care about the current, as long as the rate is zero
- ⇒ Turn zero-rates into inequality instead of equality constraints (quadratic programming)



4 Take *dendritic nonlinearity* into account

- Generalise dendritic interaction between excitation and inhibition
- Dendritic nonlinearity H converts synaptic state to somatic current

4 Take *dendritic nonlinearity* into account

- Generalise dendritic interaction between excitation and inhibition
- Dendritic nonlinearity H converts synaptic state to somatic current
- **Example:**

Current-based model: $H(J^+, J^-) = J^+ - J^-$

$$\min_{\vec{w}_i^+, \vec{w}_i^-} \quad \frac{1}{2} \sum_{k=1}^N \left\| H(\vec{w}_i^+ \vec{a}_k^+, \vec{w}_i^- \vec{a}_k^-) - J(\langle \vec{e}_i, f(\vec{x}_k) \rangle) \right\|_2^2$$

subject to $\vec{w}_i^+ \geq 0, \vec{w}_i^- \geq 0$

4 Take *dendritic nonlinearity* into account

- Generalise dendritic interaction between excitation and inhibition
- Dendritic nonlinearity H converts synaptic state to somatic current
- **Example:**

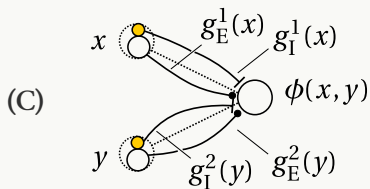
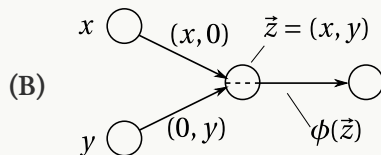
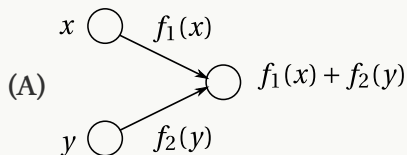
Current-based model: $H(J^+, J^-) = J^+ - J^-$

$$\min_{\vec{w}_i^+, \vec{w}_i^-} \frac{1}{2} \sum_{k=1}^N \|H(\vec{w}_i^+ \vec{a}_k^+, \vec{w}_i^- \vec{a}_k^-) - J(\langle \vec{e}_i, f(\vec{x}_k) \rangle)\|_2^2$$

subject to $\vec{w}_i^+ \geq 0, \vec{w}_i^- \geq 0$

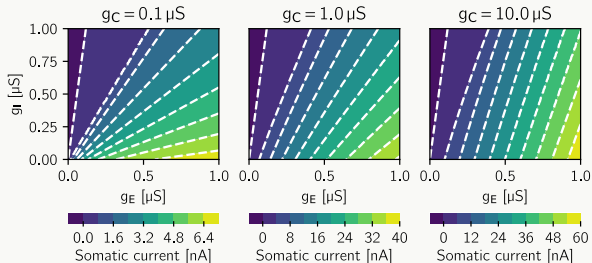
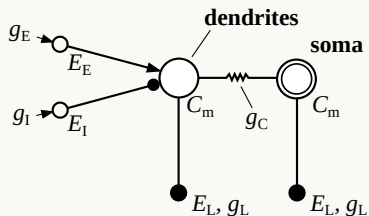
- **Questions:**
 - Can we find more interesting H ?
 - Can we exploit H as a computational resource?
 - Under which conditions can we efficiently solve the optimization problem?

Computing Multivariate Nonlinear Functions



Example: Multiplication, compute $\phi(\vec{z}) = \phi(x, y) = x \cdot y$

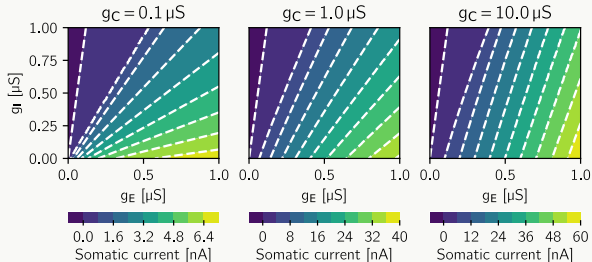
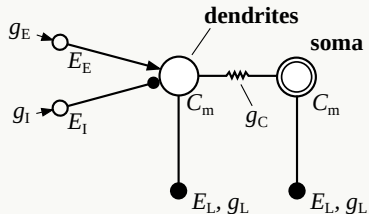
Two-compartment LIF neuron



- Dendritic nonlinearity:

$$H(g_E, g_I) = \frac{b_1 + b_2 g_E + b_3 g_I}{a_1 + a_2 g_E + a_3 g_I}$$

Two-compartment LIF neuron

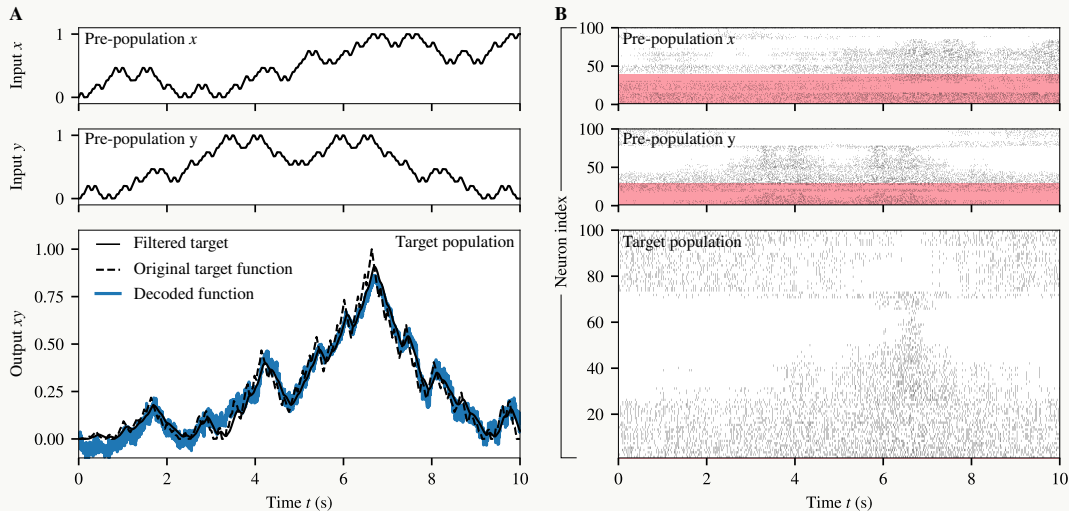


- Dendritic nonlinearity:

$$H(g_E, g_I) = \frac{b_1 + b_2 g_E + b_3 g_I}{a_1 + a_2 g_E + a_3 g_I}$$

- Can still formalize weight-optimization as convex quadratic programming problem, guaranteed to find global optimum

Results – *Dendritic computation* (I)

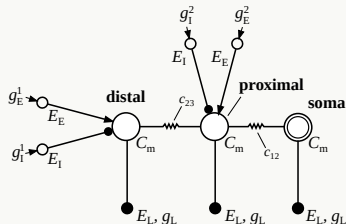


Results – *Dendritic computation* (II)

Target	Experiment setup						
	Standard LIF			Two comp. LIF $g_C = 50\text{ nS}$		Two comp. LIF $g_C = 100\text{ nS}$	
	no relaxation	A standard	B two-layer	C standard	noise model	standard	noise model
$x + y$	$5.1 \pm 0.6\%$	$5.5 \pm 1.1\%$	$11.0 \pm 1.3\%$	$3.2 \pm 1.1\%$	$9.1 \pm 1.2\%$	$5.1 \pm 1.2\%$	$11.5 \pm 1.3\%$
$x \times y$	$26.2 \pm 0.4\%$	$21.5 \pm 6.6\%$	$15.4 \pm 4.0\%$	$13.9 \pm 2.9\%$	$11.9 \pm 1.8\%$	$18.2 \pm 4.0\%$	$14.3 \pm 2.1\%$
$\sqrt{x \times y}$	$14.1 \pm 0.4\%$	$19.7 \pm 6.1\%$	$16.3 \pm 3.0\%$	$9.7 \pm 2.6\%$	$7.1 \pm 1.0\%$	$13.3 \pm 4.2\%$	$8.9 \pm 1.7\%$
$(x \times y)^2$	$44.5 \pm 0.6\%$	$33.0 \pm 6.6\%$	$18.7 \pm 6.7\%$	$27.7 \pm 4.1\%$	$27.4 \pm 4.1\%$	$34.3 \pm 5.3\%$	$30.3 \pm 4.3\%$
$x/(1+y)$	$6.0 \pm 0.4\%$	$5.2 \pm 0.7\%$	$9.5 \pm 0.8\%$	$3.4 \pm 1.0\%$	$10.0 \pm 1.6\%$	$5.3 \pm 1.3\%$	$14.0 \pm 1.9\%$
$\ (x, y)\ $	$8.0 \pm 0.4\%$	$5.7 \pm 1.1\%$	$10.5 \pm 1.0\%$	$3.1 \pm 1.3\%$	$8.9 \pm 1.2\%$	$4.3 \pm 1.8\%$	$12.3 \pm 1.8\%$
$\text{atan}(x, y)$	$10.3 \pm 0.3\%$	$8.6 \pm 1.0\%$	$13.4 \pm 1.1\%$	$5.8 \pm 1.3\%$	$8.4 \pm 1.0\%$	$7.0 \pm 1.2\%$	$12.7 \pm 1.6\%$
$\max(x, y)$	$14.9 \pm 0.3\%$	$10.0 \pm 0.9\%$	$11.3 \pm 1.4\%$	$5.5 \pm 0.9\%$	$7.7 \pm 0.9\%$	$7.3 \pm 0.9\%$	$9.7 \pm 1.0\%$

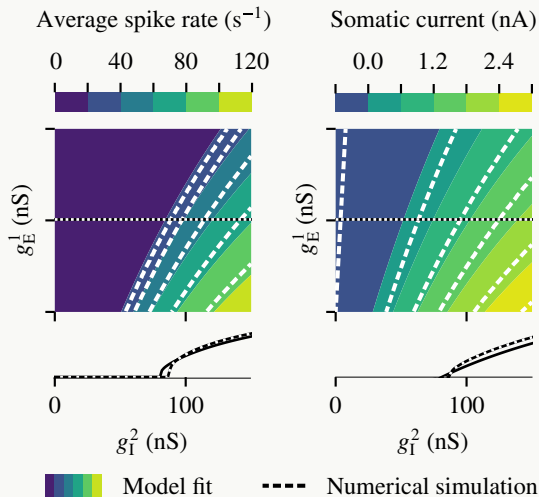
<http://arxiv.org/abs/1904.11713>

Three-compartment LIF neuron

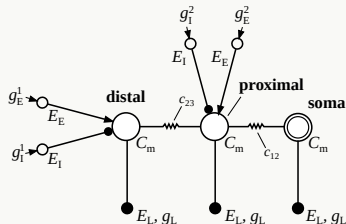


- Dendritic nonlinearity:

$$\left(\frac{b_1^1 + b_2^1 g_E + b_3^1 g_I}{a_1^1 + a_2^1 g_E + a_3^1 g_I} \right) \cdot \left(\frac{b_1^2 + b_2^2 g_E + b_3^2 g_I}{a_1^2 + a_2^2 g_E + a_3^2 g_I} \right)$$



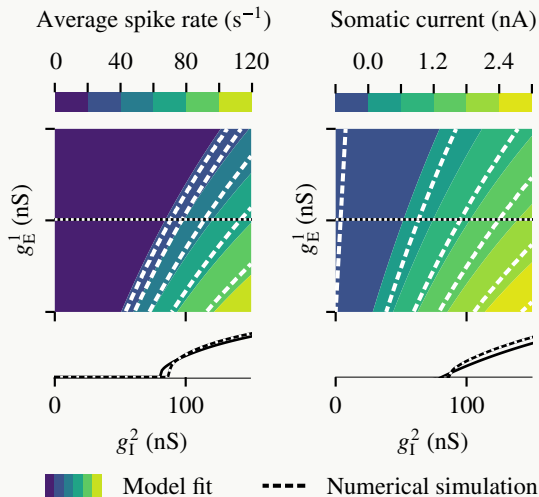
Three-compartment LIF neuron



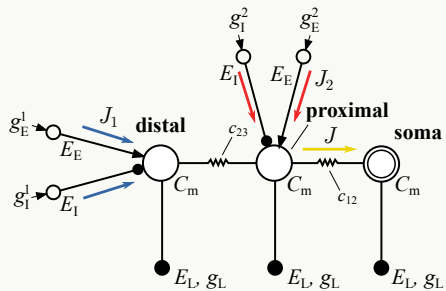
- Dendritic nonlinearity:

$$\left(\frac{b_1^1 + b_2^1 g_E + b_3^1 g_I}{a_1^1 + a_2^1 g_E + a_3^1 g_I} \right) \cdot \left(\frac{b_1^2 + b_2^2 g_E + b_3^2 g_I}{a_1^2 + a_2^2 g_E + a_3^2 g_I} \right)$$

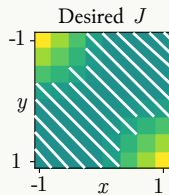
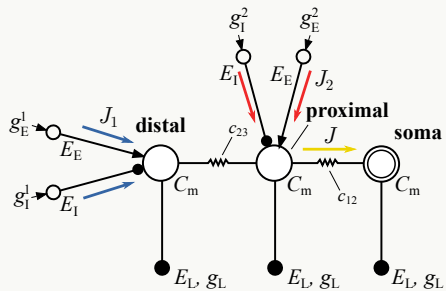
- For n -LIF, $n \geq 3$: can formalize weight-optimization as iterative trust-region-based optimization problem;
not guaranteed to find global optimum



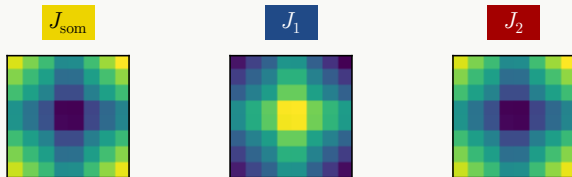
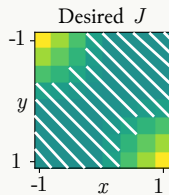
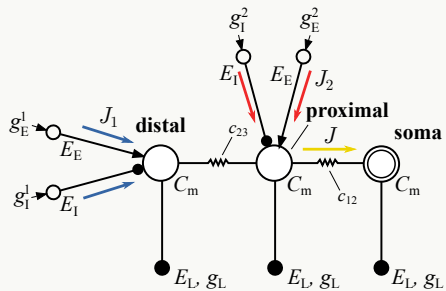
Three-compartment LIF neuron – *Example*



Three-compartment LIF neuron – *Example*

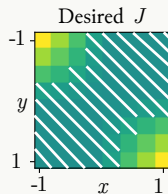
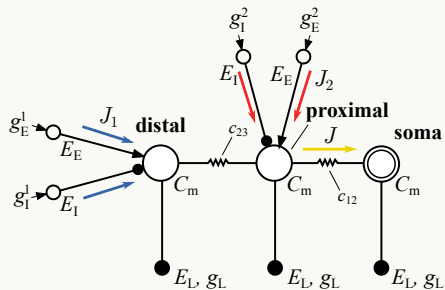


Three-compartment LIF neuron – *Example*

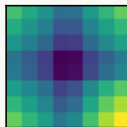


ITERATION 0

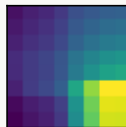
Three-compartment LIF neuron – *Example*



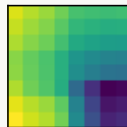
J_{som}



J_1

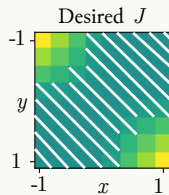
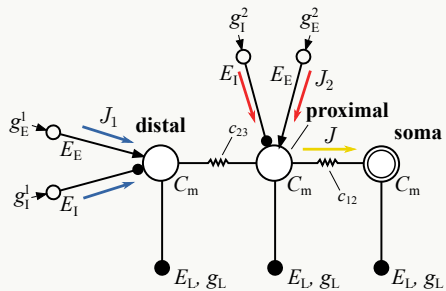


J_2



ITERATION 1

Three-compartment LIF neuron – *Example*



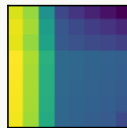
J_{som}



J_1

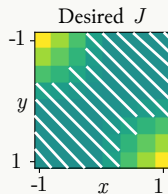
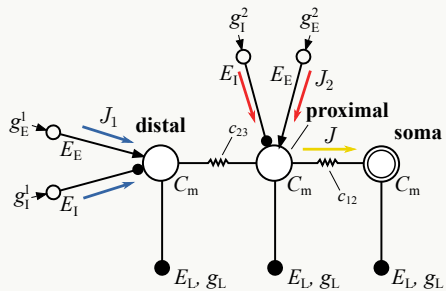


J_2

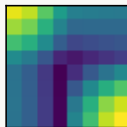


ITERATION 2

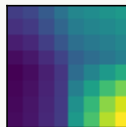
Three-compartment LIF neuron – *Example*



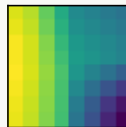
J_{som}



J_1

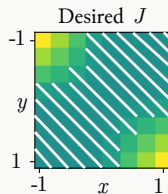
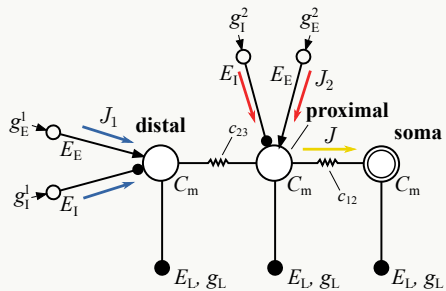


J_2

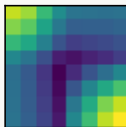


ITERATION 3

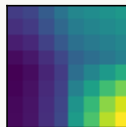
Three-compartment LIF neuron – *Example*



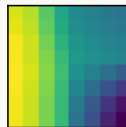
J_{som}



J_1

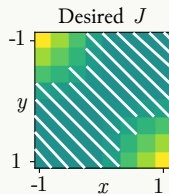
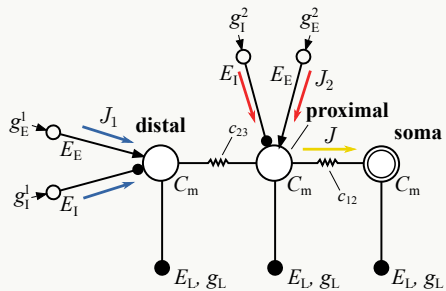


J_2

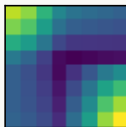


ITERATION 4

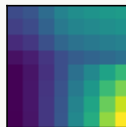
Three-compartment LIF neuron – *Example*



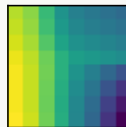
J_{som}



J_1

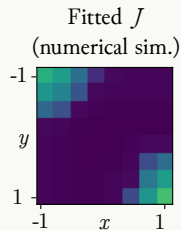
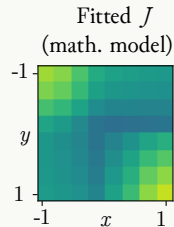
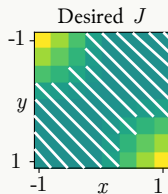
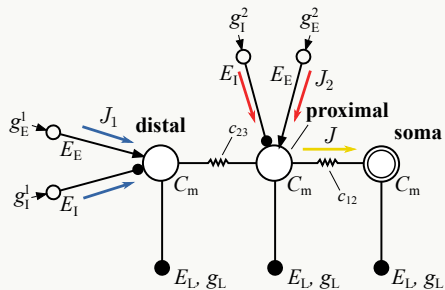


J_2

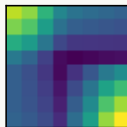


ITERATION 5

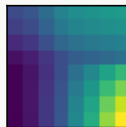
Three-compartment LIF neuron – *Example*



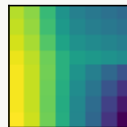
J_{som}



J_1



J_2



ITERATION 5

PART III

Detailed Neuron Models and the NEF

Neural Engineering Framework

state space

\mathbf{x}



neuron space

\mathbf{A}

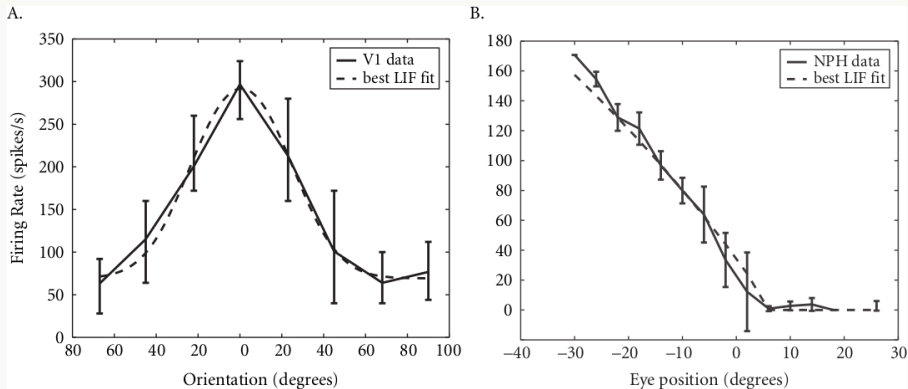


Figure: Tuning curves from monkey visual cortex (A) and human NPH (B)

Problem Statement

Tuning-curve-based methods assume non-adaptive neurons:

$$a_i = F(\mathbf{x})$$

For adaptive neurons, activity depends on signal history

$$a_i = F(\mathbf{x}, t)$$

Goal: extend NEF methods to account for cellular dynamics

Approach

Assume neuron model is a “black box”

Use time-varying signals $\mathbf{x}(t)$ to train neural connection parameters

- Encoding: iterative optimization of gain, bias
- Decoding: least squares optimization of decoders, synapses

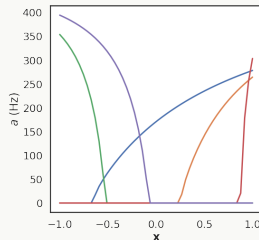
Considerations:

- $\mathbf{x}(t)$ spans state space and activates appropriate dynamics
- $\mathbf{x}(t)$ passes through preliminary filters (spikes, dendrites, etc.)

Static Encoding

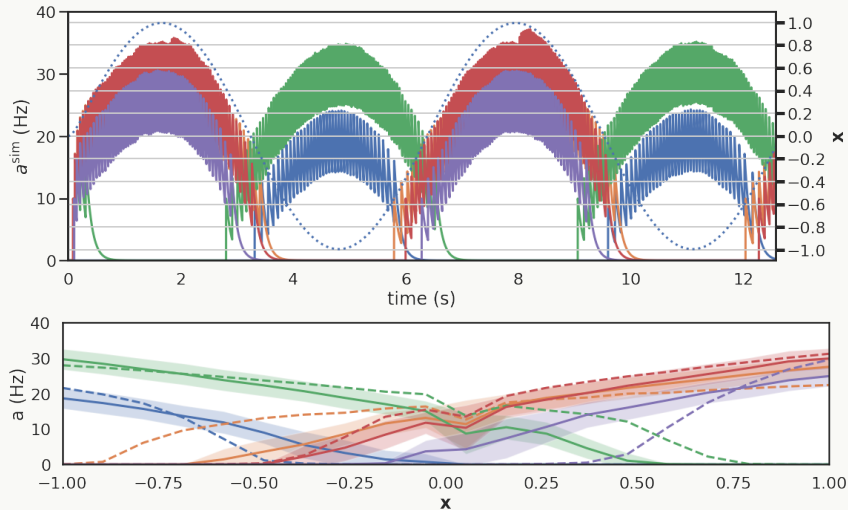
Find gain and bias that achieve desired max_rates and intercepts

1. Find $J^{\text{test}}(x)$ for $\mathbf{x} \in [-1, 1]$ assuming $\alpha = 1, \beta = 0$
2. Calculate $a(J^{\text{test}})$ using rate-approximation
3. Interpolate $a = G(J(\mathbf{x}))$
4. Use G^{-1} to solve for α and β



Temporal Encoding

Find gain and bias that achieve desired max_rates and intercepts



Static Decoding

Minimizes state-space error over \mathbf{x}

$$E = \frac{1}{2} \int_{-1}^1 (\mathbf{x} - \hat{\mathbf{x}})^2 dx$$

$$\hat{\mathbf{x}} = \sum_i a_i \mathbf{d}_i$$

Solve using least-squares

$$A \mathbf{d} = \mathbf{x}$$

$$\mathbf{d} = (A^T A + \sigma^2)^{-1} A^T \mathbf{x}$$

Static Decoding

Minimizes state-space error over \mathbf{x}

$$E = \frac{1}{2} \int_{-1}^1 (\mathbf{x} - \hat{\mathbf{x}})^2 dx$$

$$\hat{\mathbf{x}} = \sum_i a_i \mathbf{d}_i$$

Solve using least-squares

$$A \mathbf{d} = \mathbf{x}$$

$$\mathbf{d} = (A^T A + \sigma^2)^{-1} A^T \mathbf{x}$$

Activity matrix calculated by static encoding

$$A = \begin{bmatrix} a_1(\mathbf{x} = -1) & \dots & a_n(\mathbf{x} = -1) \\ \vdots & \ddots & \vdots \\ a_1(\mathbf{x} = 1) & \dots & a_n(\mathbf{x} = 1) \end{bmatrix}$$

$$a_i(\mathbf{x}) = G_i(\mathbf{x}, \alpha_i, \beta_i)$$

Temporal Decoding

Minimize state-space error over t

$$E = \frac{1}{2} \int_0^{t'} (\mathbf{x}(t) - \hat{\mathbf{x}}(t))^2 dt$$

$$\hat{\mathbf{x}}(t) = \sum_i a_i(t) \mathbf{d}_i$$

Solve using least-squares

$$\begin{aligned} A \mathbf{d} &= \mathbf{x}(t) \\ \mathbf{d} &= (A^T A)^{-1} A^T \mathbf{x}(t) \end{aligned}$$

Temporal Decoding

Minimize state-space error over t

$$E = \frac{1}{2} \int_0^{t'} (\mathbf{x}(t) - \hat{\mathbf{x}}(t))^2 dt$$

$$\hat{\mathbf{x}}(t) = \sum_i a_i(t) \mathbf{d}_i$$

Solve using least-squares

$$\begin{aligned} A \mathbf{d} &= \mathbf{x}(t) \\ \mathbf{d} &= (A^T A)^{-1} A^T \mathbf{x}(t) \end{aligned}$$

Activity matrix collected by simulating over time

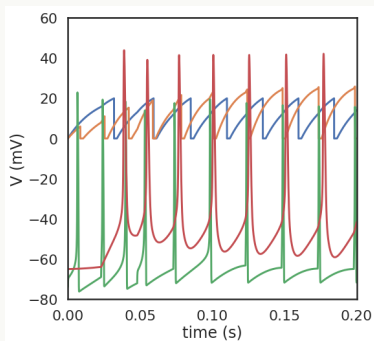
$$A = \begin{bmatrix} a_1(t=0) & \dots & a_n(t=0) \\ \vdots & \ddots & \vdots \\ a_1(t=t') & \dots & a_n(t=t') \end{bmatrix}$$

$$\mathbf{x}(t) = [\text{computed directly}]$$

Neuron Models

Adaptive LIF

- spikes increase V^{thr}
- divisive effect on J

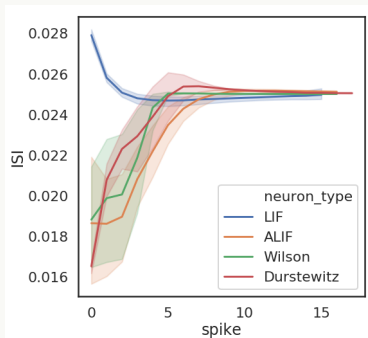


Wilson

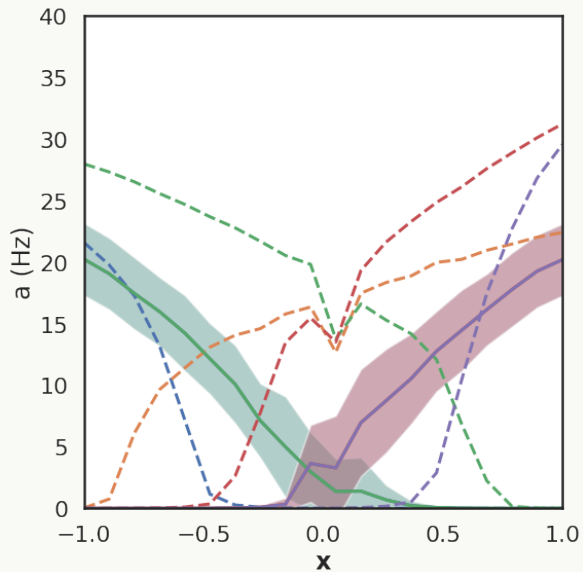
- mammalian neocortex
- 3 coupled ODEs for V, R, C

Durstewitz

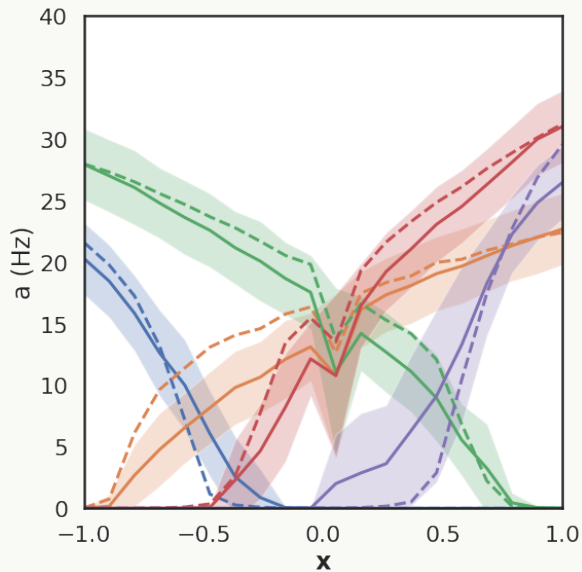
- Layer-V PC
- conductance syn.
- 4 geo. compartments
- 6 HH ion channels



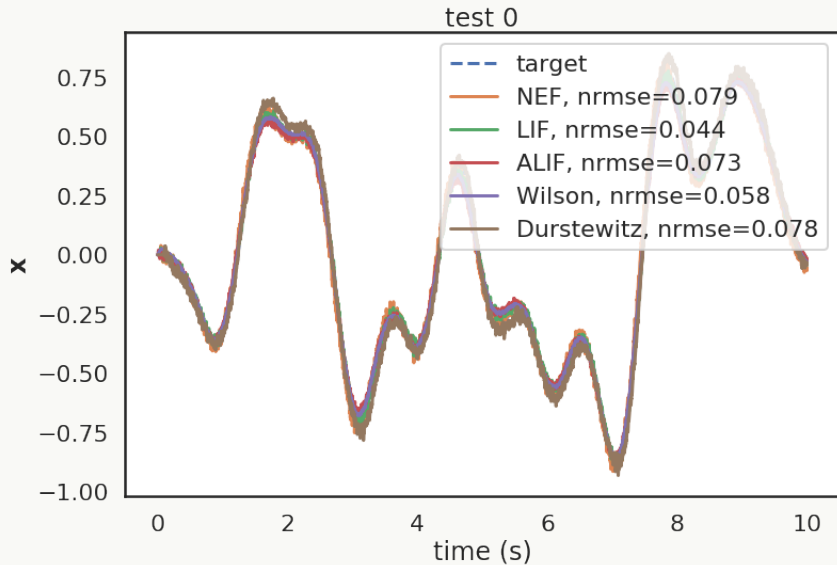
Encoding: Training α and β



Encoding: Training α and β



Decoding: $f(x) = x$



Decoding: $f(x) = x$

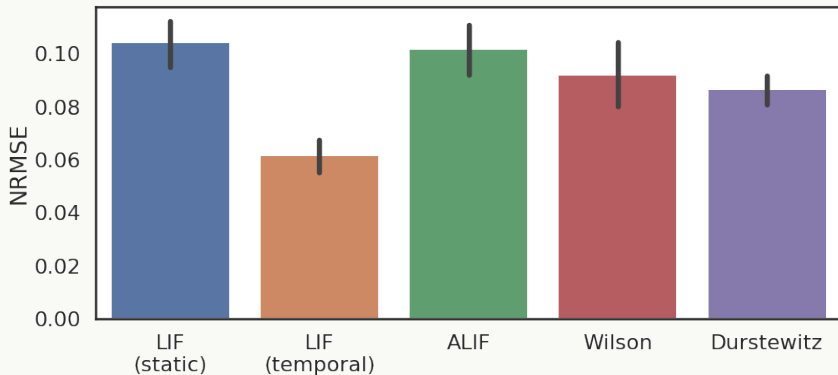
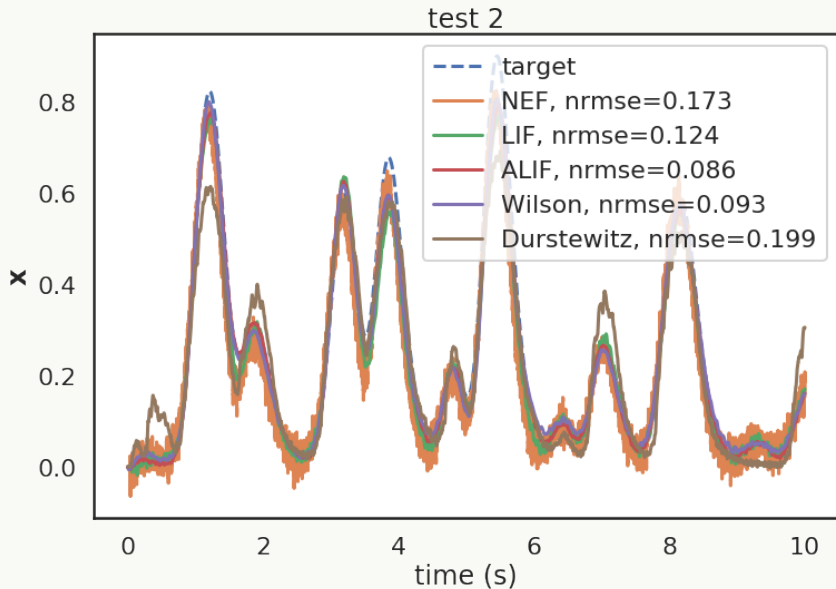


Figure: Average performance across neuron models

Decoding: $f(x) = x^2$



Decoding: $f(x) = x^2$

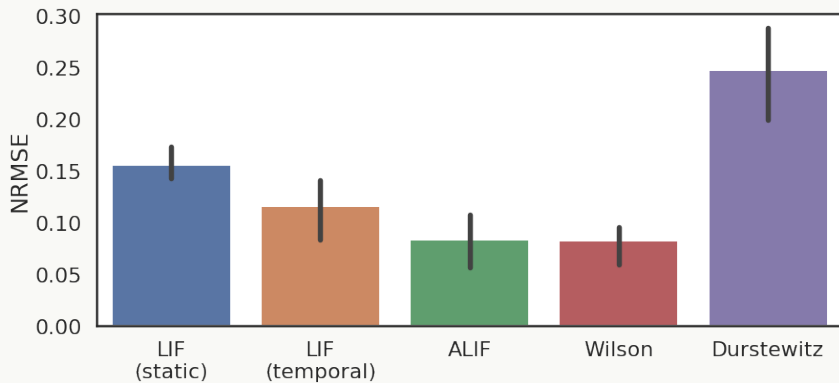


Figure: Average performance across neuron models

Dynamical System: 2D Oscillator

Figure: Implementing a two-dimensional oscillator using 200 recurrently-connected neurons

Dynamical System: Lorenz Attractor

Figure: Implementing a three-dimensional chaotic attractor using 2000 recurrently-connected neurons

Future Work

Nengo Integration

- Online learning

Applications

- Harder dynamics
- Cognitive models
- Drug effects

Extensions

- Dale's principle
- Functional neuromodulation

Summary

Training α, β using $\mathbf{x}(t)$ distributes tuning curves of complex neurons

Training \mathbf{d} using $\mathbf{x}(t)$ decodes activities of adaptive neurons

Training $h(t)$ using $\mathbf{x}(t)$ controls for cell dynamics in recurrent networks

Thanks to

Aaron Voelker Terry Stewart Andreas Stöckel Chris Eliasmith

PART IV

nengo-bio hands-on