Symbolic Computation in Spiking Neural Networks

Kraišniković Ceca
Institute of Theoretical Computer Science

Graz, June 18, 2019

On Symbolic Computation

- Abstract representations of tasks
- Computation using symbolic expressions
  - Arithmetic expressions:
    x + y, x – y, x * y, …

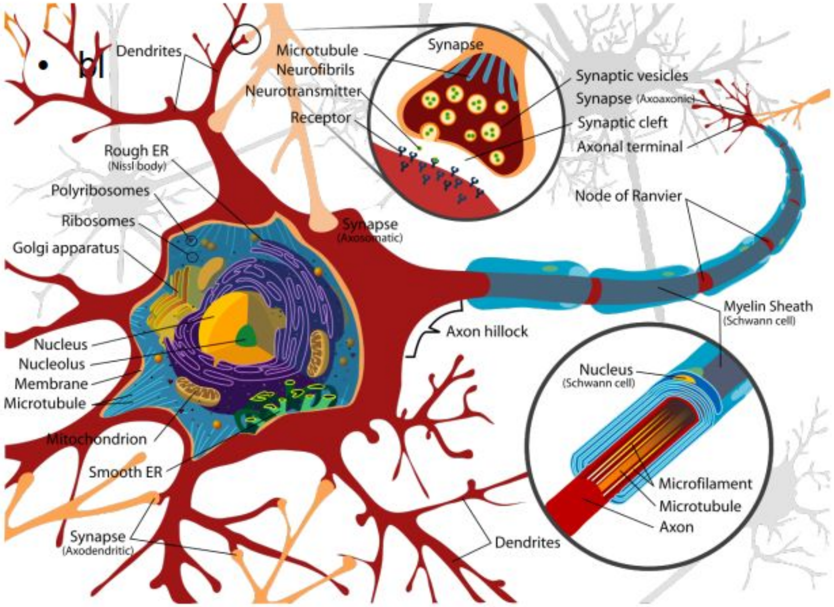
- Challenges for artificial neural networks:
  - Flexible cognitive control:
    RED, GREEN, BLUE, PURPLE
  - Free generalization
    e.g., wug - wugged

Motivation and Contribution

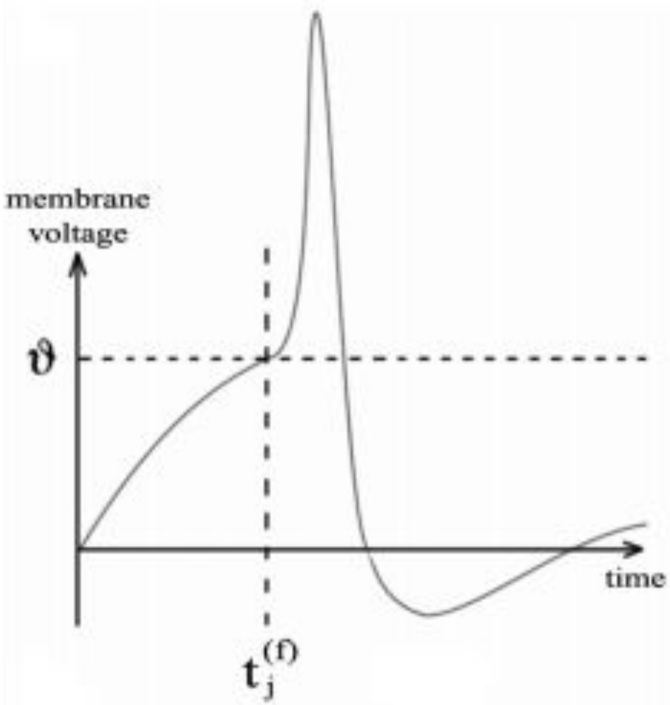
- Understanding intelligence
- Can symbolic computation be performed by (spiking) neural networks?

- **Goal:**
  - move towards computation using “higher-level” rules
- **Experiment:**
  - Learning to solve symbolic arithmetic expressions

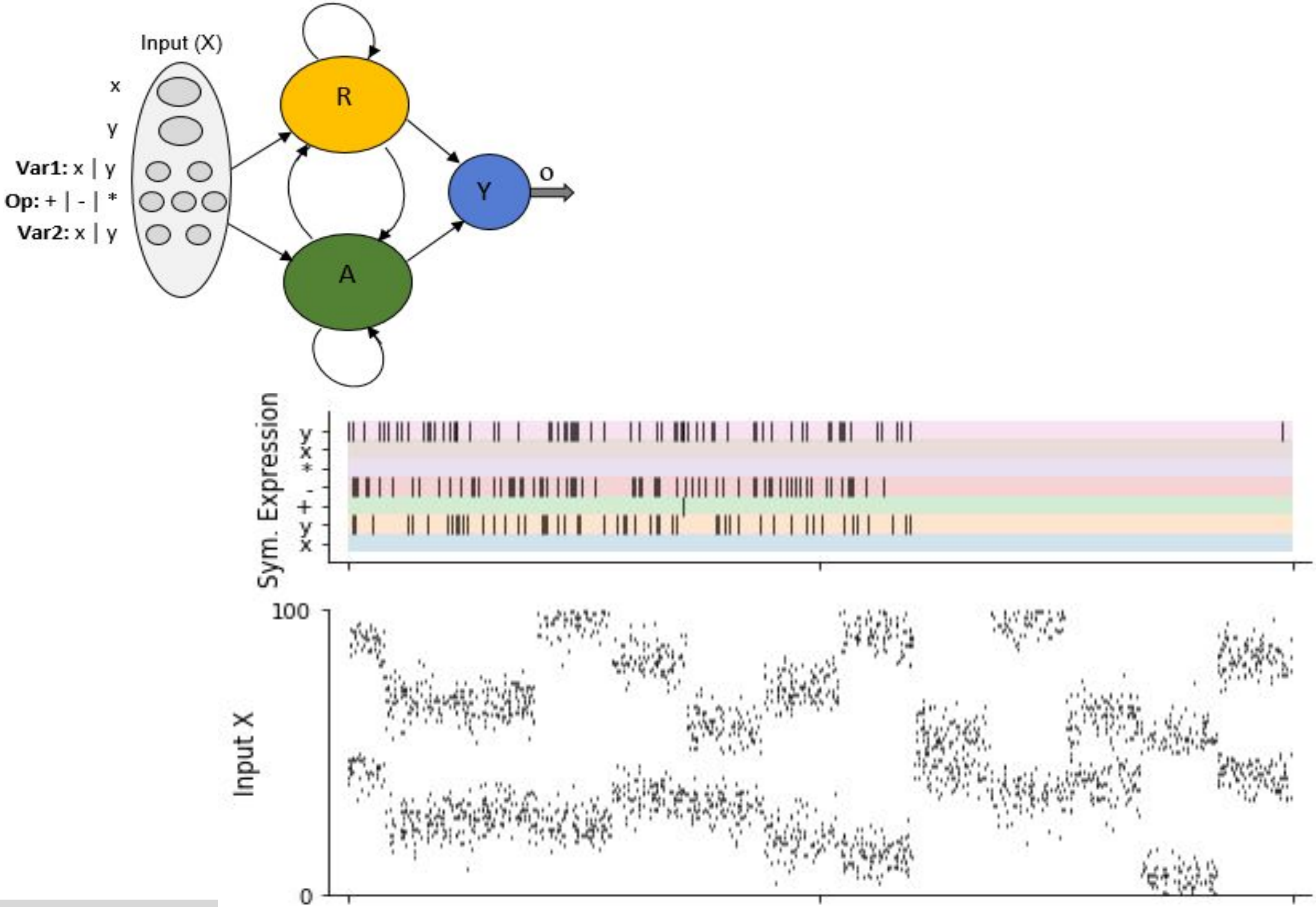
Biological neuron



Leaky Integrate-and-Fire Neuron



Learning to Solve Symbolic Arithmetic Expressions

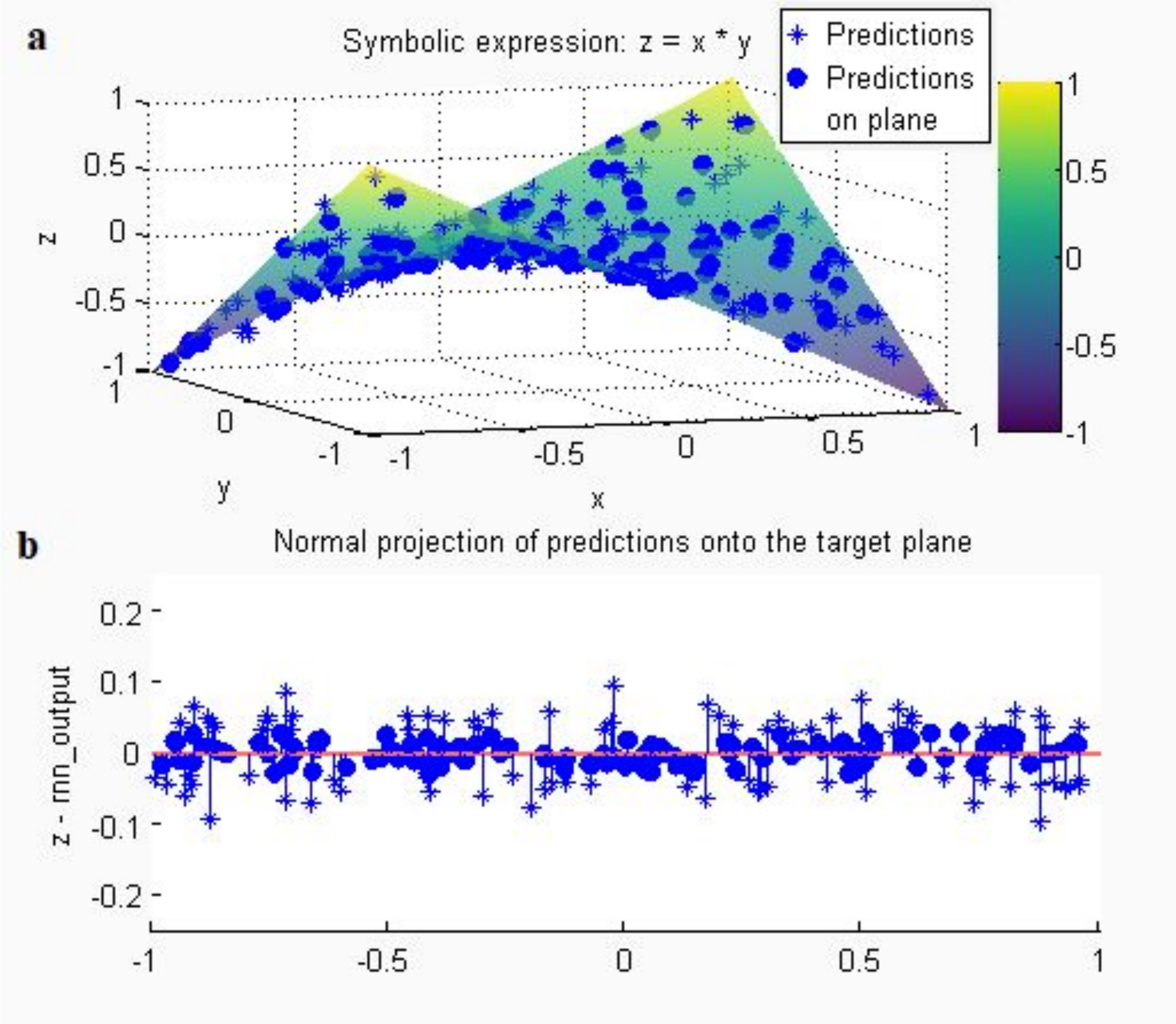


Calculation using arith. expression y - y

Step no.	Arith. expression y – y	Network output	Mean Squared Error (MSE)
1	-0.4705 – (0.4705) = 0	-0.0014	0.0000
2	0.9398 – 0.9398 = 0	-0.0887	0.0079
…	…	…	…
50	0.2152 – 0.2152 = 0	0.0279	0.0008
51	0.1826 – 0.1826 = 0	-0.0411	0.0017
…	…	…	…
99	0.2508 – 0.2508 = 0	-0.0196	0.0004
100	0.4771 – 0.4771 = 0	-0.0306	0.0009

Calculation using arith. expression y - y

Step no.	Arith. expression y – y	Network output	Mean Squared Error (MSE)
1	-0.4705 – (-0.4705) = 0	-0.0014	0.0000
2	0.9398 – 0.9398 = 0	-0.0887	0.0079
…	…	…	…
50	0.2152 – 0.2152 = 0	0.0279	0.0008
51	0.1826 – 0.1826 = 0	-0.0411	0.0017
…	…	…	…
99	0.2508 – 0.2508 = 0	-0.0196	0.0004
100	0.4771 – 0.4771 = 0	-0.0306	0.0009


a
Symbolic expression: z = x * y
Predictions
Predictions on plane
z
y
x
b
Normal projection of predictions onto the target plane
z - nn_output

Conclusion

- Our model is able to store and retrieve information, and to calculate arithmetic expressions.
- Our first step to show that symbolic computation is possible with connectionist model

[G. Bellec, D. Salaj, A. Subramoney, R. Legenstein, and W. Maass. Long short-term memory and learning-to-learn in networks of spiking neurons. *32nd Conference on Neural Information Processing Systems (NIPS 2018), Montreal, Canada*, 2018.]

Thank you for your attention!

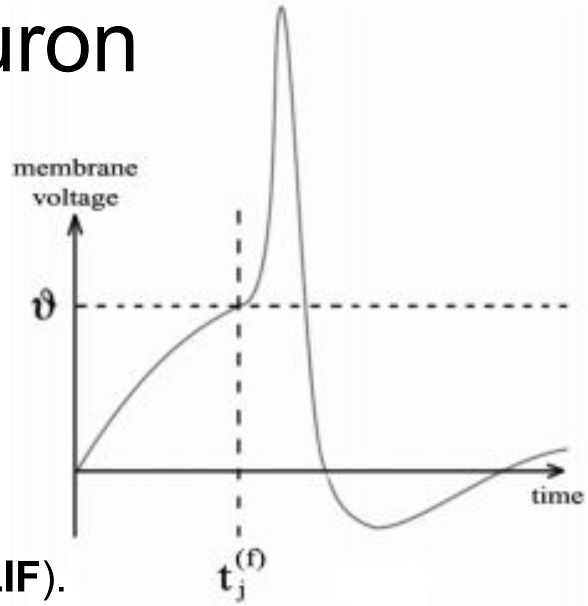
Questions?

Leaky Integrate-and-Fire Neuron

$$\tau_m \frac{du}{dt} = -u(t) + RI(t)$$

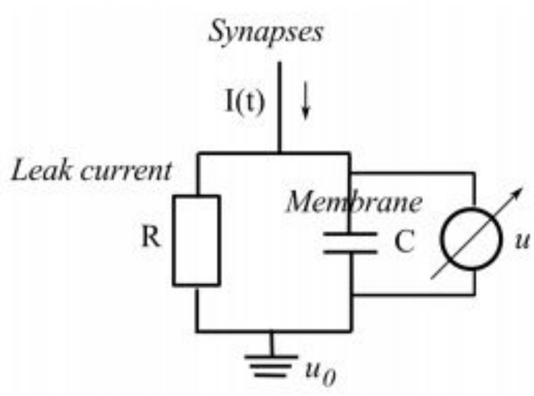
$$I_i(t) = \sum_j w_{ij} \sum_f \alpha(t - t_j^{(f)})$$

$$\alpha(s) = exp\left(-\frac{s}{\tau_s}\right)\Theta(s)$$



LIF neurons are augmented with an adaptive threshold (**ALIF**).

τ_m-membrane time constant,
$u(t)$ - membrane voltage,
$I(t)$ - current, w_{ij} - efficacy,
$t_j^{(f)}$- firing time of neuron j,
τ_s-time decay constant,
$\Theta(s)$ - Heaviside step function



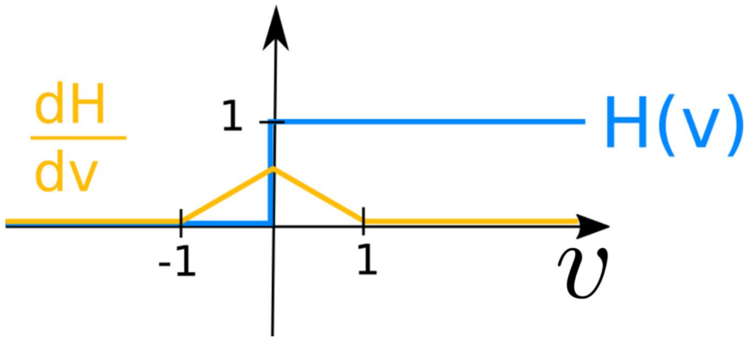
[Wulfram Gerstner and Werner Kistler. *Spiking Neuron Models: An Introduction*. Cambridge University Press, New York, NY, USA, 2002.]

Backpropagation Through Time (BPTT)

- BPTT is used to train RNNs.
  - Gradients are propagated through many steps.
- Outputs of spiking neurons are non-differentiable.
  - Dampened pseudo-derivative can be used instead.

$$\frac{dH}{dv} = \gamma \max\{0, 1 - |v|\},$$

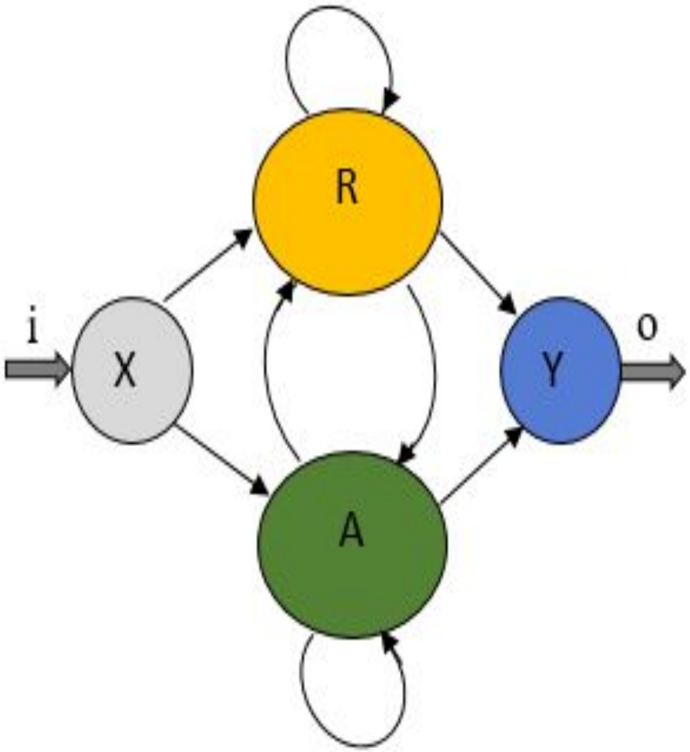
v – normalized membrane potential, γ – dampening factor.



[G. Bellec, D. Salaj, A. Subramoney, R. Legenstein, and W. Maass. Long short-term memory and learning-to-learn in networks of spiking neurons. *32nd Conference on Neural Information Processing Systems (NIPS 2018), Montreal, Canada*, 2018.]

Network Architecture

- Populations of neurons:
  - X – input,
  - Y – output,
  - R – regular spiking (LIF),
  - A – adaptive spiking (ALIF).
- Inputs i encoded to spike trains
- Output o: mean traces of spiking activity

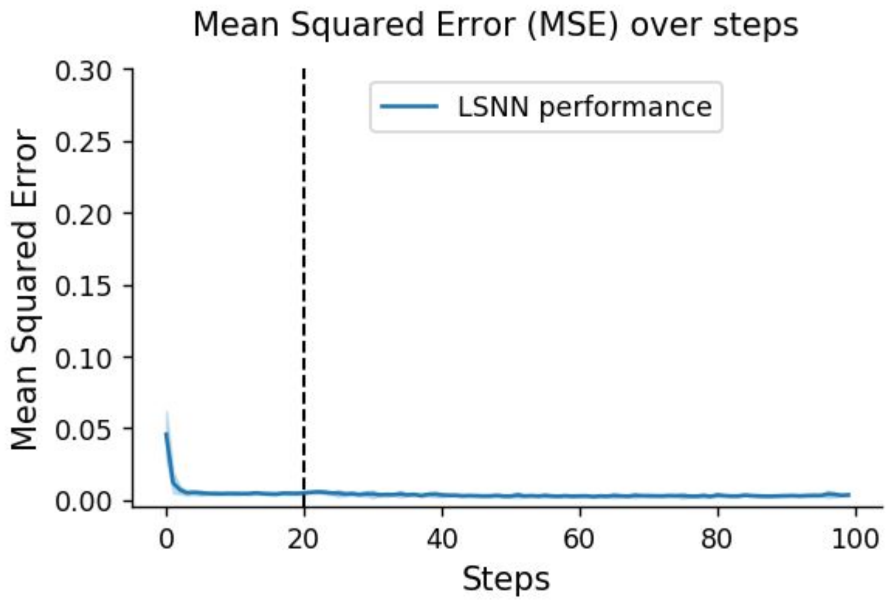


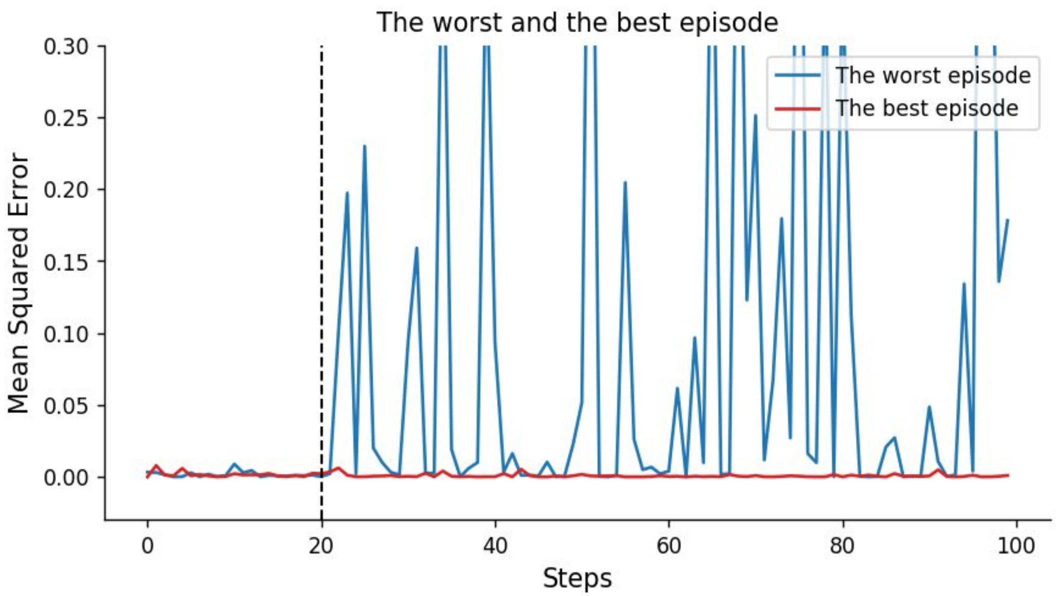
Experiment:
Learning to Solve Symbolic Arithmetic Expressions

$$Var_1 \;\; Operator \;\; Var_2 = Solution,$$

where $Var_1, Var_2 \in \{x, y\}, Operator \in \{+, -, *\},\; x, y \in [-1,1]$.

Expression	Target function
$z = x - x$, $z = y - y$	Horizontal line z = 0
$z = x + x = 2x$, $z = y + y = 2y$	Linear function (with a slope)
$z = x * x = x2$, $z = y * y = y2$	Quadratic function (parabola)
$z = x - y$, $z = y - x$, $z = x + y$,	Plane (3D space)
$z = x * y$	Hyperbolic paraboloid


Mean Squared Error (MSE) over steps
LSNN performance
Mean Squared Error
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
20
40
60
80
100
Steps


The worst and the best episode
The worst episode
The best episode
Mean Squared Error
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
20
40
60
80
100
Steps

Input Encoding

- Analog values from the input range [-1, 1]
- 100 input neurons, values from the input range equally distributed
- Gaussian response with particular mean on that analog value and a constant standard deviation $\sigma = 0.002$
- Firing rate of neuron i:

$$r_i = r_{max} \exp\left(-\frac{(m_i - z_i)^2}{2\sigma^2}\right),$$

$r_{max} = 200\ Hz$, m_i - *value for which neuron* i *is responsible*, z_i - value to encode.

Neuron Model

LIF

$$\tau_m \frac{du}{dt} = -u(t) + RI(t)$$

$$I_i(t) = \sum_j w_{ij} \sum_f \alpha(t - t_j^{(f)})$$

$$\alpha(s) = A\,exp\left(-\frac{s}{\tau_s}\right)\Theta(s)$$

τ_s - time constant for decay

Adaptive LIF

$$\tau_m \frac{du_j}{dt} = -u_j(t) + R_m I_j(t)$$

$$\tau_{a,j} \frac{db_j}{dt} = b_j^0 - b_j(t)$$

τ_m – membrane time constant

$\tau_{a,j}$ – adaptation time constant

In discrete time:

$$u_j(t+dt) = \alpha u_j(t) + (1-\alpha) R_m I_j(t) - b_j(t) z_j(t)$$

$$b_j(t+dt) = \rho_j b_j(t) + (1-\rho_j) z_j(t)$$

$$\alpha = exp\left(-\frac{dt}{\tau_m}\right)$$

$$\rho_j = \exp\left(-\frac{dt}{\tau_{a,j}}\right)$$

Network Parameters

Parameter	**Value**
maximum firing rate	200 Hz
threshold b_j	0.03 V
dt	1 timestep
refractory steps	5 timesteps
delay steps d	5 timesteps
τ_m	20 ms
dampening factor	0.4 or 0.3

Parameters for all spiking neurons

Parameter	**Value**
β	1.6
$\tau_{a,j}$	1-1000ms or 1-8000ms

Additional parameters for adaptive spiking neurons

1)

Parameter	**Value**
number of recurrent neurons	250
proportion of adaptive neurons	0.4
dampening factor	0.4
tau adaptation	1-1000 ms
number of steps in an episode	500
duration of steps	20 ms

2)

Parameter	**Value**
number of recurrent neurons	300
proportion of adaptive neurons	0.4
dampening factor	0.3
tau adaptation	1-8000 ms
number of steps in an episode	100
duration of steps	40 ms

3)

Parameter	**Value**
number of recurrent neurons	350
proportion of adaptive neurons	0.4
dampening factor	0.3
tau adaptation	1-8000 ms
number of steps in an episode	100
duration of steps	40 ms