Variational ground state search on the BrainScaleS-2 neuromorphic hardware

15th March 2021, Virtual APS March Meeting

Robert Klassert[1], Andreas Baumbach[1], Stefanie Czischek[1,2] and Martin Gärttner[1]

1 Kirchhoff-Institut für Physik, Ruprecht-Karls-Universität Heidelberg, Im NeuenheimerFeld 227, 69120 Heidelberg, Germany

2 Department of Physics and Astronomy, University of Waterloo, Ontario, N2L 3G1, Canada

UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386

KIRCHHOFF-INSTITUTE FOR PHYSICS

BrainScaleS

STRUCTURES CLUSTER OF EXCELLENCE

Simulating many-body systems with neural quantum states

Simulating many-body systems with neural quantum states

$\uparrow\uparrow\downarrow\uparrow$

system of N qubits: $|\psi\rangle = \sum_{v_1,\ldots,v_N} c_{v_1,\ldots,v_N} |v_1 \ldots v_N\rangle$

Simulating many-body systems with neural quantum states

$\uparrow\uparrow\downarrow\uparrow$

system of N qubits: $|\psi\rangle = \sum_{v_1,\ldots,v_N} c_{v_1,\ldots,v_N} |v_1 \ldots v_N\rangle$

$|\psi_\theta\rangle$

desired variational ansatz:

- efficient representation $\dim(\theta) = \mathrm{poly}(N)$
- efficient sample generation

Simulating many-body systems with neural quantum states

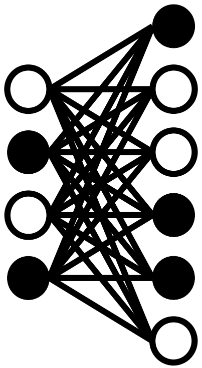
$\uparrow\uparrow\downarrow\uparrow$

system of N qubits: $|\psi\rangle = \sum_{v_1,\ldots,v_N} c_{v_1,\ldots,v_N} |v_1 \ldots v_N\rangle$

$|\psi_\theta\rangle$

desired variational ansatz:

- efficient representation $\dim(\theta) = \mathrm{poly}(N)$
- efficient sample generation



synthetic measurement distribution p_θ

expectation values $\langle O \rangle_{p_\theta}$

optimization $\Delta\theta$

Simulating many-body systems with neural quantum states

↑↑↓↑

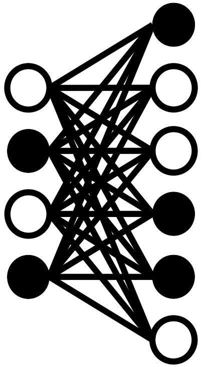
system of N qubits: $|\psi\rangle = \sum_{v_1,\ldots,v_N} c_{v_1,\ldots,v_N} |v_1 \ldots v_N\rangle$

$|\psi_\theta\rangle$

desired variational ansatz:

- efficient representation $\dim(\theta) = \mathrm{poly}(N)$
- efficient sample generation

Restricted Boltzmann machine



→ synthetic measurement p_θ distribution

→ expectation values $\langle O \rangle_{p_\theta}$

← optimization $\Delta\theta$ ←

Simulating many-body systems with neural quantum states

$\uparrow\uparrow\downarrow\uparrow$

system of N qubits: $|\psi\rangle = \sum_{v_1,\ldots,v_N} c_{v_1,\ldots,v_N} |v_1 \ldots v_N\rangle$

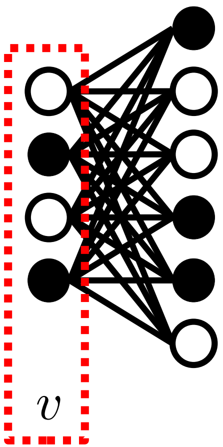
$|\psi_\theta\rangle$

desired variational ansatz:

- efficient representation $\dim(\theta) = \mathrm{poly}(N)$
- efficient sample generation

Restricted Boltzmann machine

$v = (v_1, \ldots, v_N)$



synthetic measurement p_θ distribution

expectation values $\langle O \rangle_{p_\theta}$

optimization $\Delta\theta$

Simulating many-body systems with neural quantum states

$\uparrow\uparrow\downarrow\uparrow$

system of N qubits: $|\psi\rangle = \sum_{v_1,\ldots,v_N} c_{v_1,\ldots,v_N} |v_1 \ldots v_N\rangle$

$|\psi_\theta\rangle$

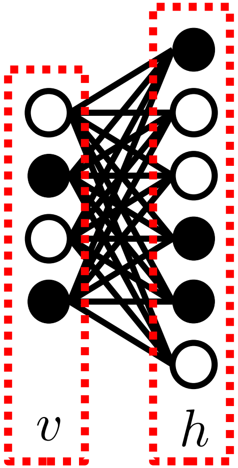
desired variational ansatz:

- efficient representation $\dim(\theta) = \text{poly}(N)$
- efficient sample generation

Restricted Boltzmann machine

$v = (v_1, \ldots, v_N)$

$[v, h] \in \{0,1\}^{N_v + N_h}$



synthetic measurement p_θ distribution

expectation values $\langle O \rangle_{p_\theta}$

optimization $\Delta\theta$

Simulating many-body systems with neural quantum states

$\uparrow\uparrow\downarrow\uparrow$

system of N qubits: $|\psi\rangle = \sum_{v_1,\ldots,v_N} c_{v_1,\ldots,v_N} |v_1 \ldots v_N\rangle$

$|\psi_\theta\rangle$

desired variational ansatz:

- efficient representation $\dim(\theta) = \mathrm{poly}(N)$
- efficient sample generation

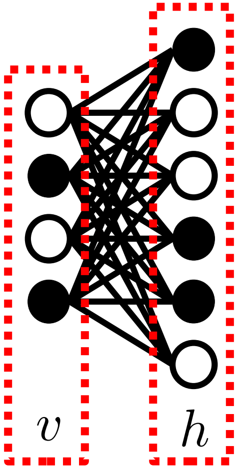
Restricted Boltzmann machine

$v = (v_1, \ldots, v_N)$

$[v, h] \in \{0, 1\}^{N_v + N_h}$

$\theta = [W, b]$

$\dim(\theta) = N_v N_h + N_v + N_h$



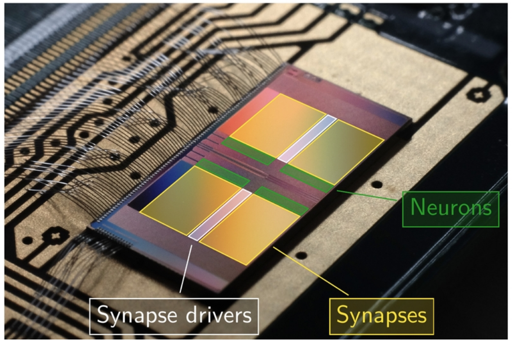
synthetic measurement distribution p_θ

expectation values $\langle O \rangle_{p_\theta}$

optimization $\Delta\theta$

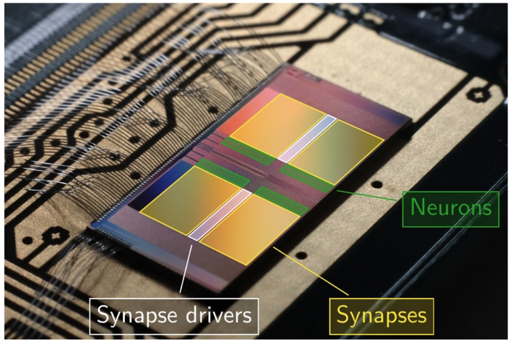
Why neuromorphic hardware for neural quantum states?

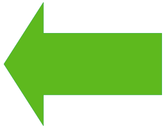
BrainScaleS-2 (BSS2)
neuromorphic chip



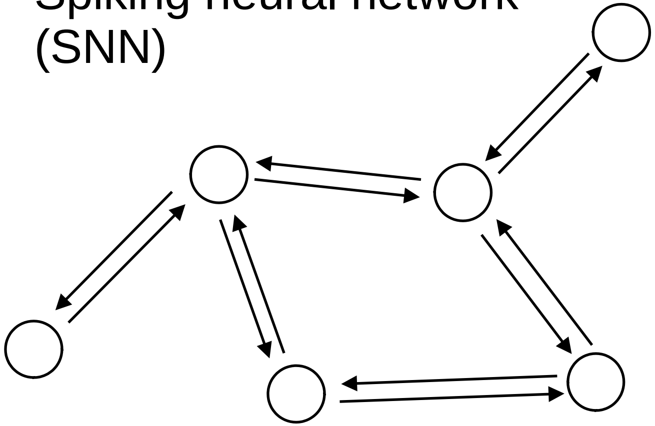
Why neuromorphic hardware for neural quantum states?

BrainScaleS-2 (BSS2) neuromorphic chip





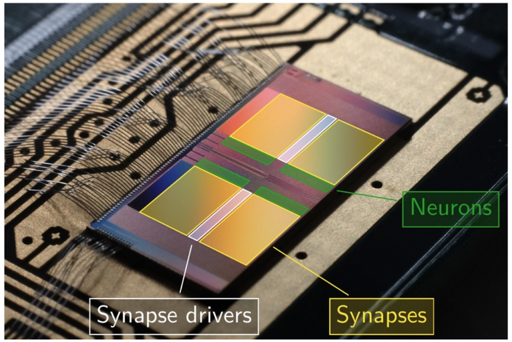
Spiking neural network (SNN)



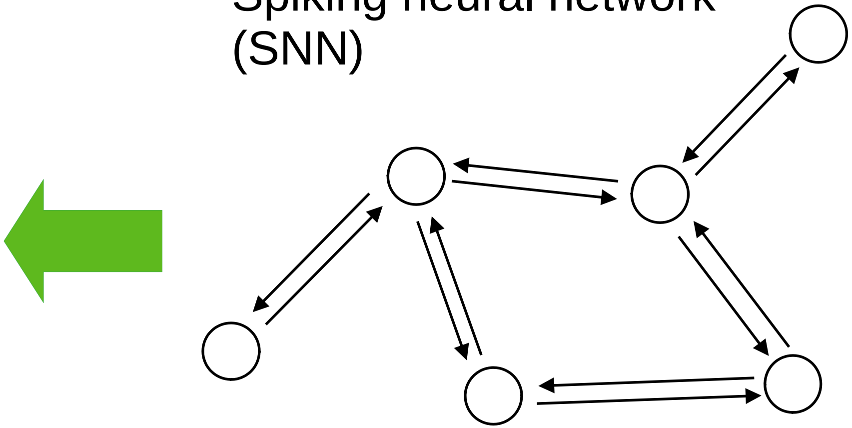
- BSS2 implements physical SNN as continuous dynamical system

Why neuromorphic hardware for neural quantum states?

BrainScaleS-2 (BSS2)
neuromorphic chip



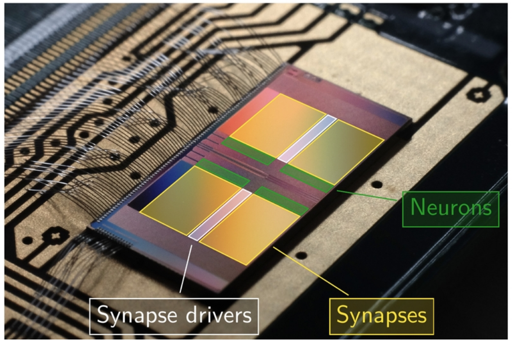
Spiking neural network
(SNN)



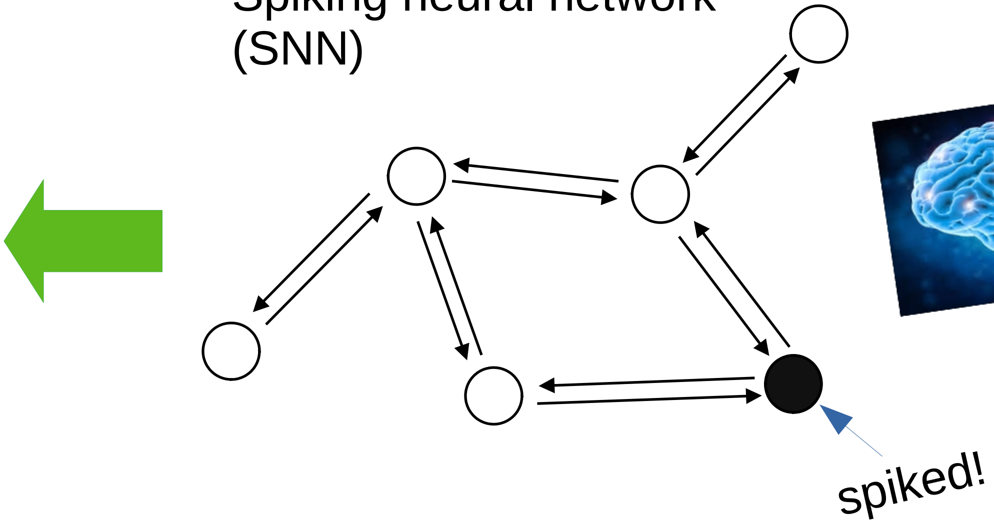
- BSS2 implements physical SNN as continuous dynamical system

Why neuromorphic hardware for neural quantum states?

BrainScaleS-2 (BSS2)
neuromorphic chip



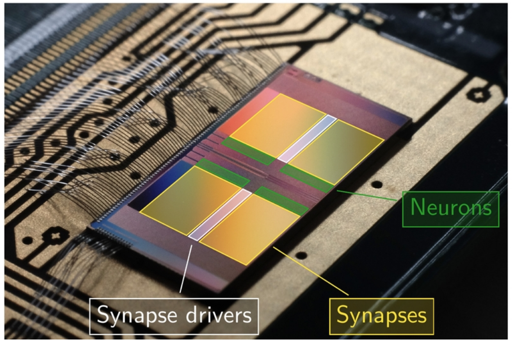
Spiking neural network
(SNN)



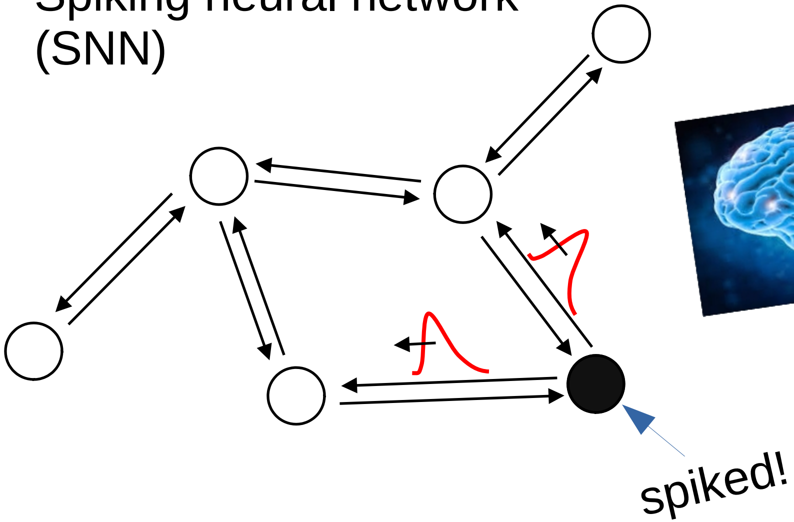
- BSS2 implements physical SNN as continuous dynamical system

Why neuromorphic hardware for neural quantum states?

BrainScaleS-2 (BSS2)
neuromorphic chip



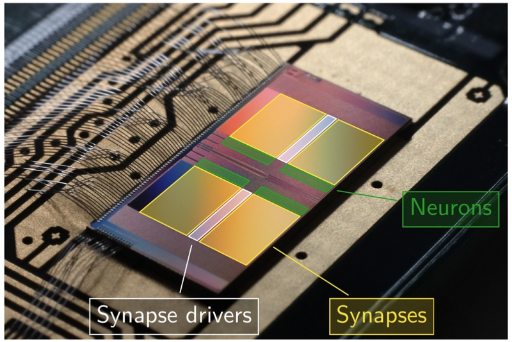
Spiking neural network
(SNN)



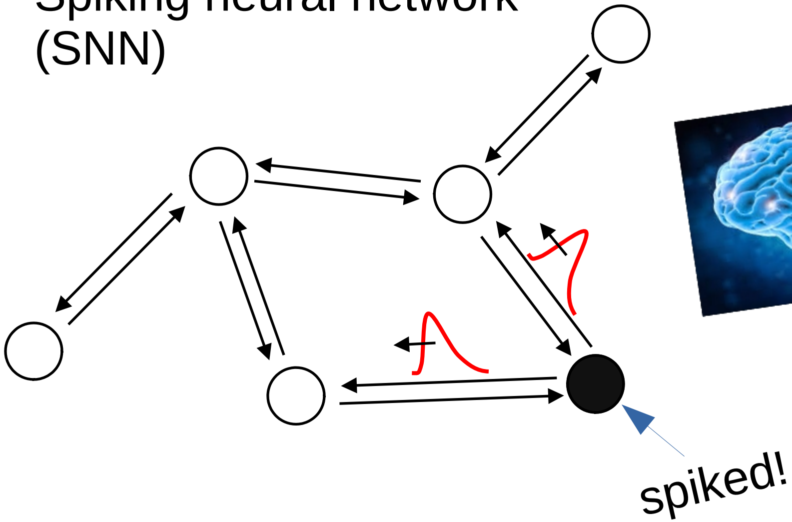
- BSS2 implements physical SNN as continuous dynamical system

Why neuromorphic hardware for neural quantum states?

BrainScaleS-2 (BSS2) neuromorphic chip



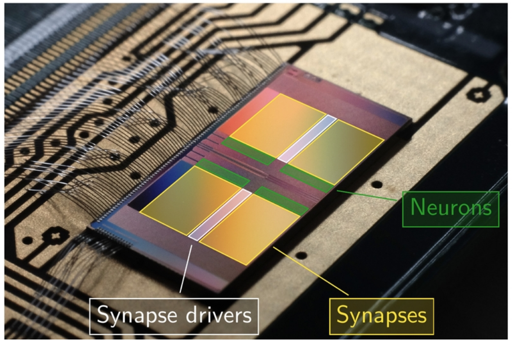
Spiking neural network (SNN)



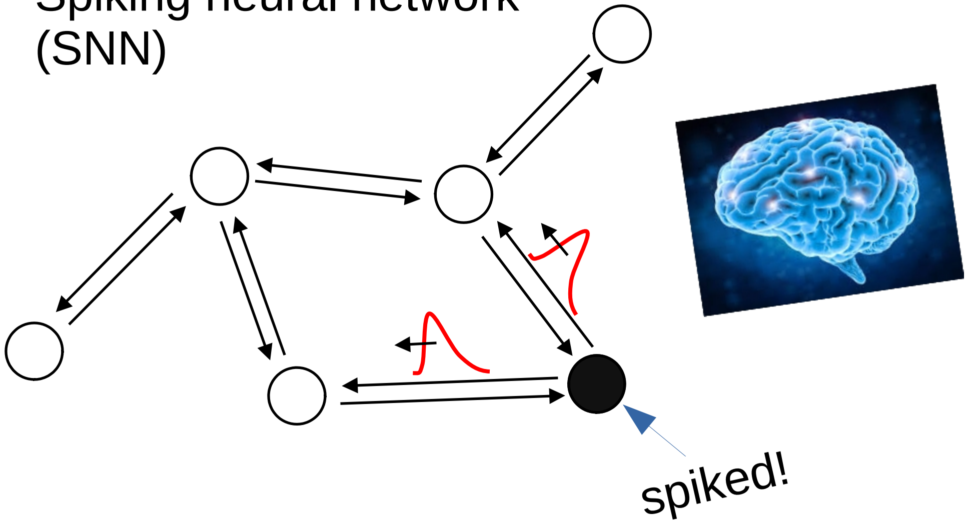
- BSS2 implements physical SNN as continuous dynamical system
- inherent parallelism enables fast sampling independent of network size

Why neuromorphic hardware for neural quantum states?

BrainScaleS-2 (BSS2)
neuromorphic chip



Spiking neural network
(SNN)

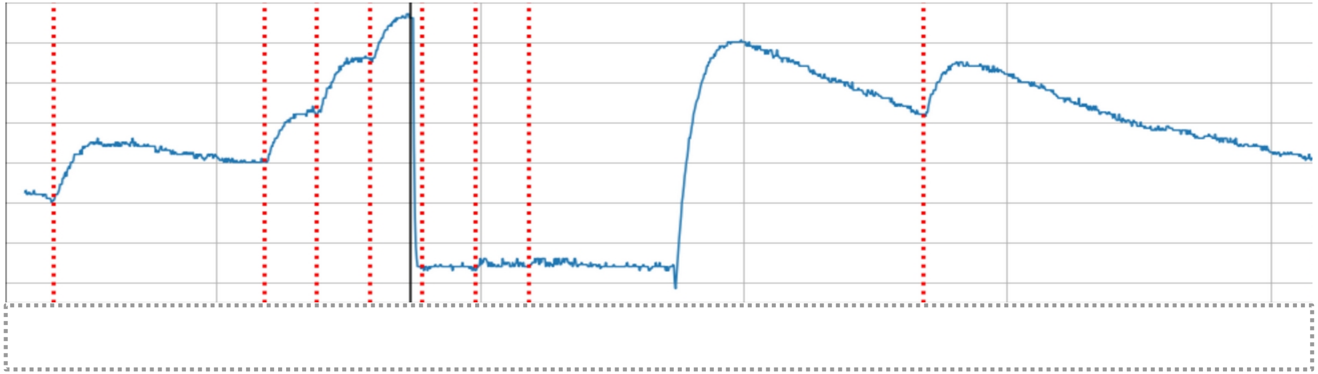


- BSS2 implements physical SNN as continuous dynamical system
- inherent parallelism enables fast sampling independent of network size
- recently *Czischek et al., arXiv:2008.01039 (2020)*, show high-fidelity encoding of Bell pairs

Neural sampling with dynamic spiking neuron models

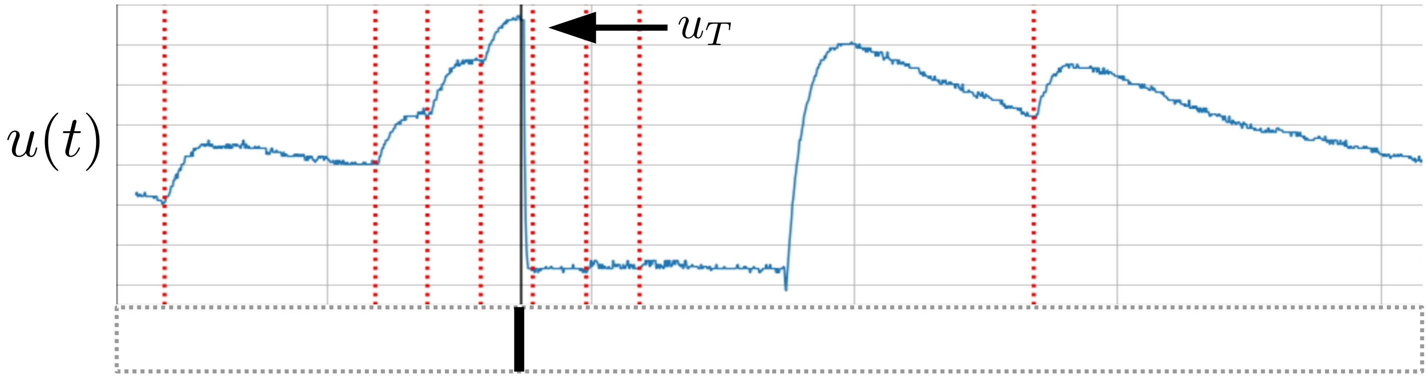
dynamics of single neuron's membrane potential u

$u(t)$



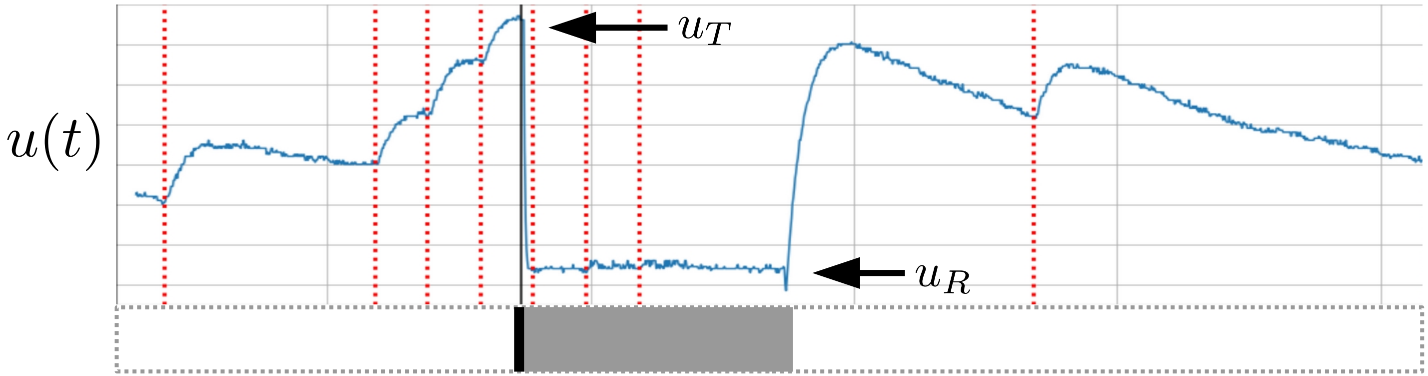
Neural sampling with dynamic spiking neuron models

dynamics of single neuron's membrane potential u



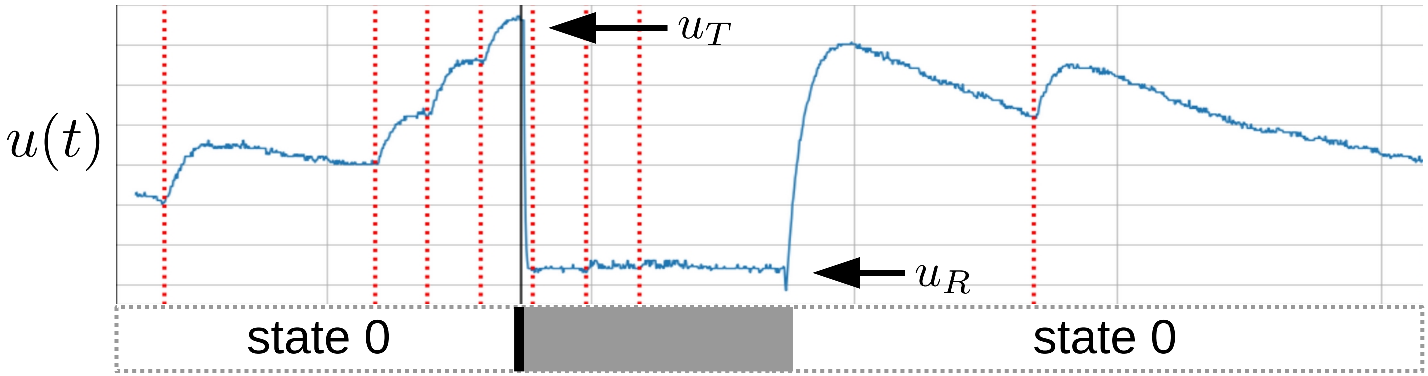
Neural sampling with dynamic spiking neuron models

dynamics of single neuron's membrane potential u



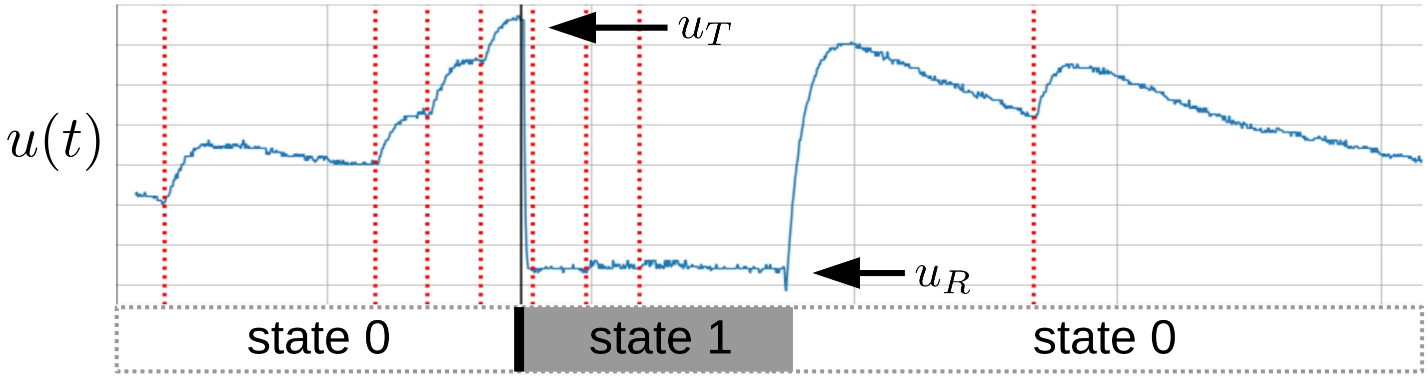
Neural sampling with dynamic spiking neuron models

dynamics of single neuron's membrane potential u



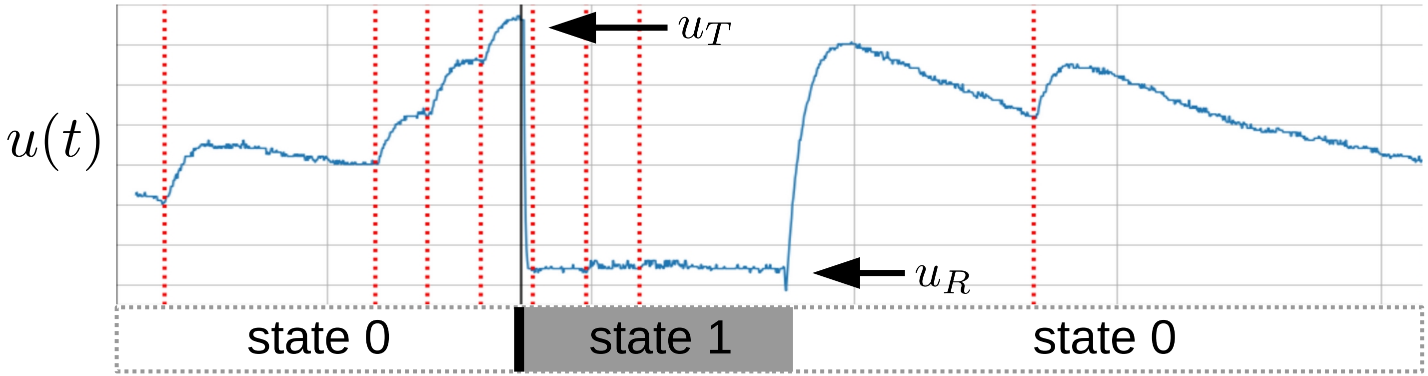
Neural sampling with dynamic spiking neuron models

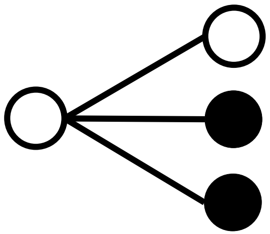
dynamics of single neuron's membrane potential u



Neural sampling with dynamic spiking neuron models

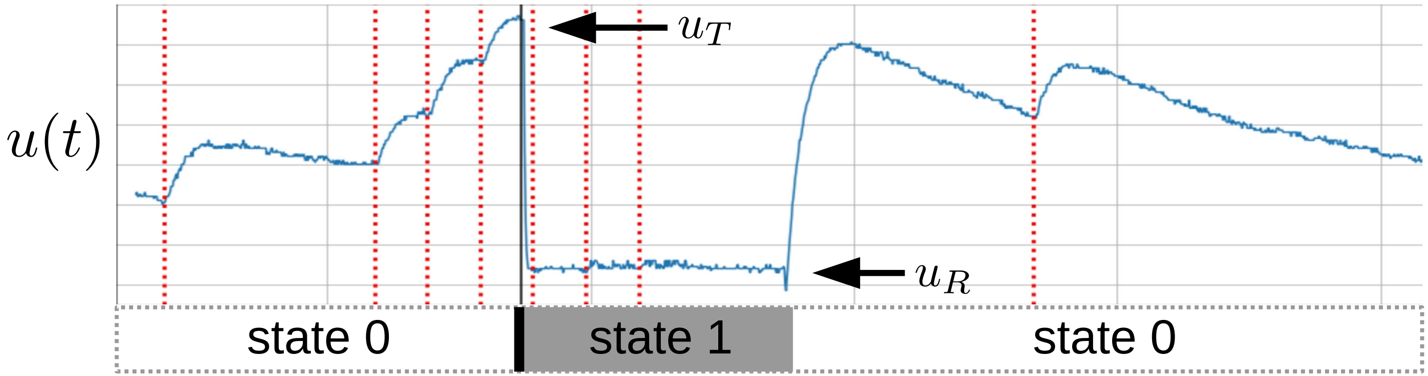
dynamics of single neuron's membrane potential u

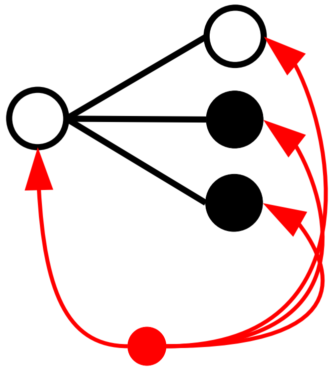




Neural sampling with dynamic spiking neuron models

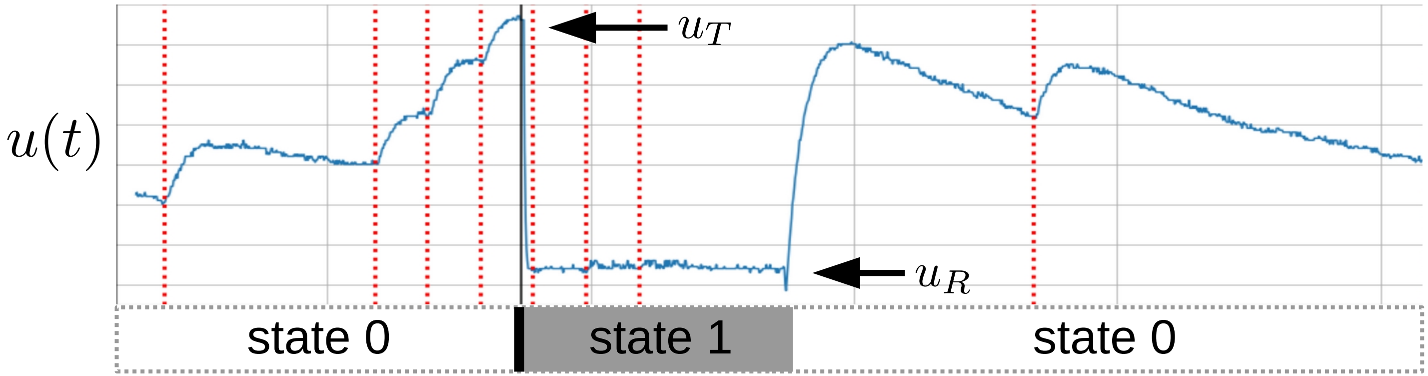
dynamics of single neuron's membrane potential u

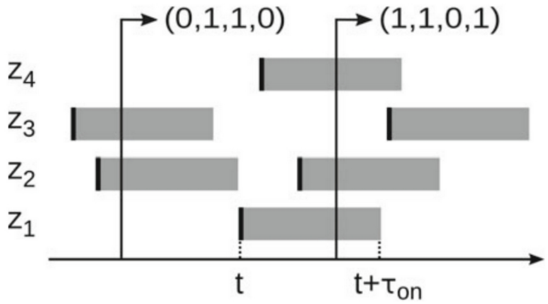


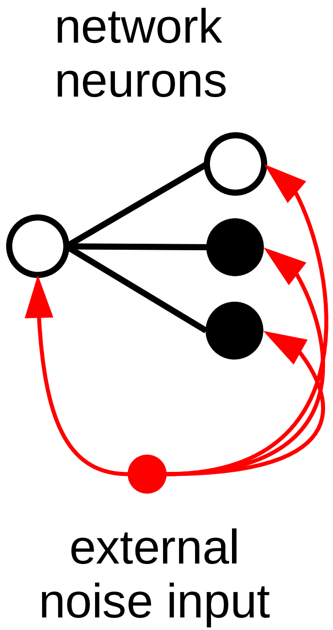


Neural sampling with dynamic spiking neuron models

dynamics of single neuron's membrane potential u

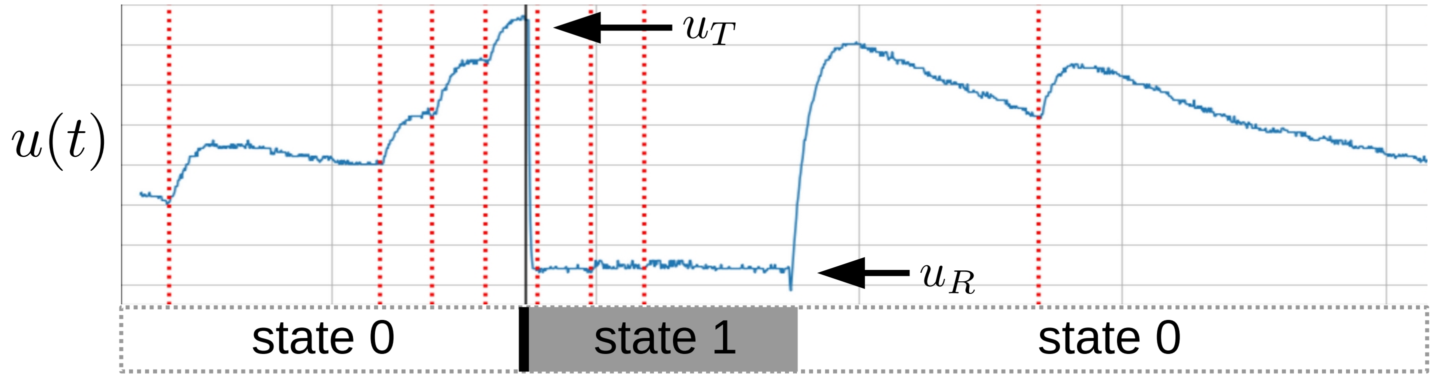




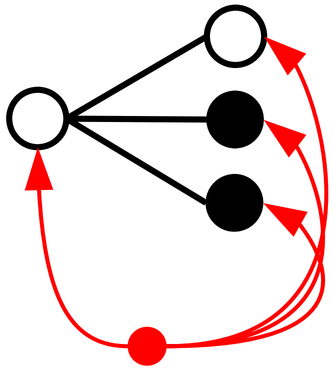


Neural sampling with dynamic spiking neuron models

dynamics of single neuron's membrane potential u

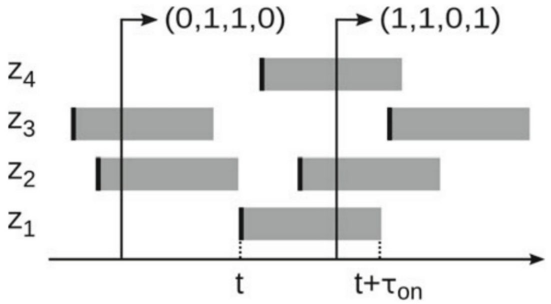


network neurons



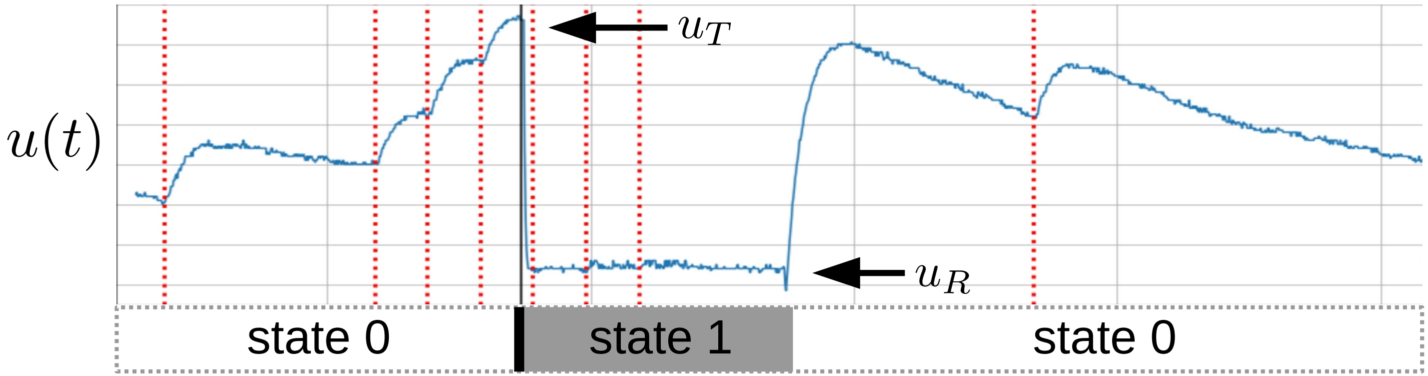
external noise input

- sparse spike data represent samples

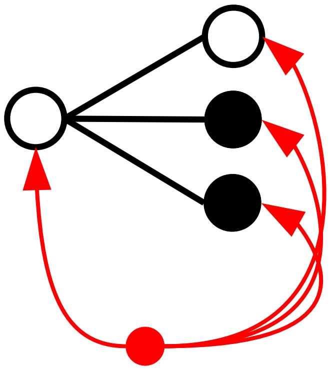


Neural sampling with dynamic spiking neuron models

dynamics of single neuron's membrane potential u

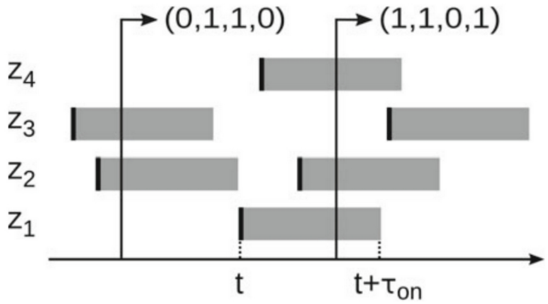


network
neurons



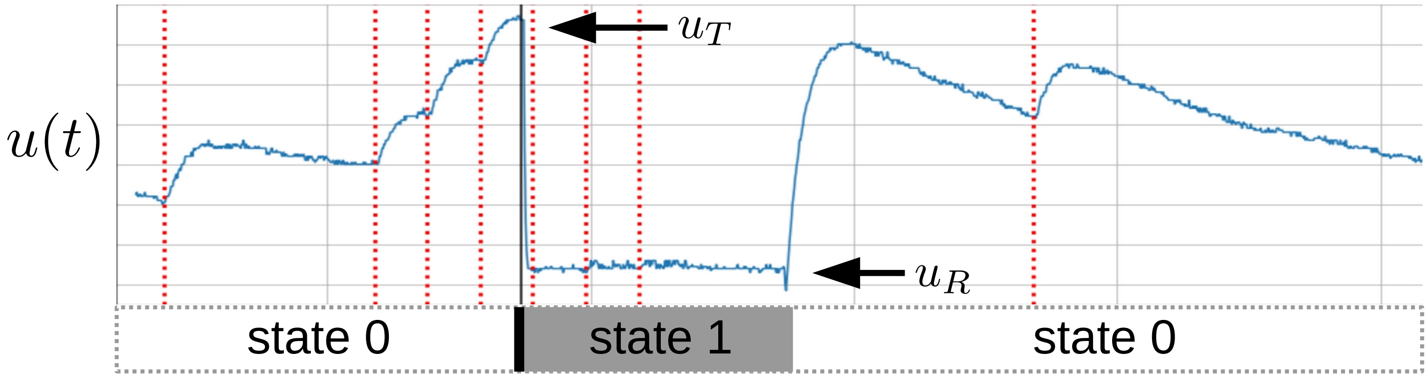
external
noise input

- sparse spike data represent samples
- 10^5 samples / second

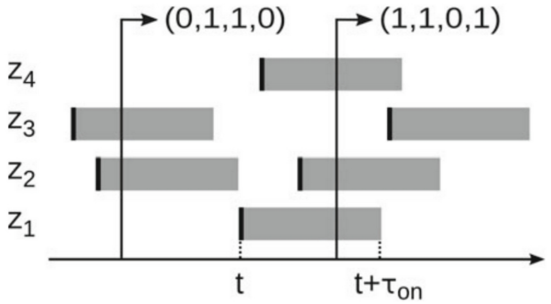


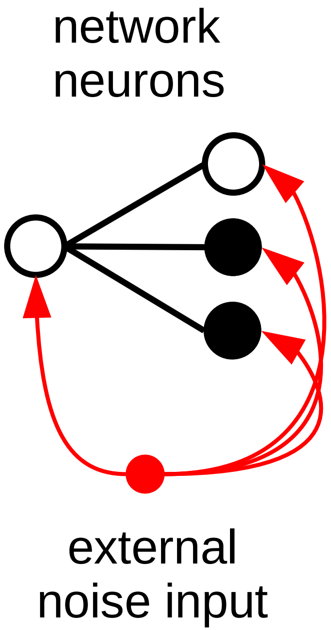
Neural sampling with dynamic spiking neuron models

dynamics of single neuron's membrane potential u



- sparse spike data represent samples
- 10^5 samples / second
- independent of network size





Encoding the quantum states

$$H_{\text{TFIM}} = -J \sum_{\langle i,j \rangle}^{N} \sigma_z^i \sigma_z^j - h \sum_{i}^{N} \sigma_x^i$$

1D, periodic boundary conditions

Encoding the quantum states

$$H_{\text{TFIM}} = -J \sum_{\langle i,j \rangle}^{N} \sigma_z^i \sigma_z^j - h \sum_{i}^{N} \sigma_x^i$$

1D, periodic boundary conditions

stoquastic H → $c_v \in \mathbb{R}_{\geq 0}$

Encoding the quantum states

$$H_{\mathrm{TFIM}} = -J \sum_{\langle i,j \rangle}^{N} \sigma_z^i \sigma_z^j - h \sum_{i}^{N} \sigma_x^i$$

1D, periodic boundary conditions

stoquastic H → $c_v \in \mathbb{R}_{\geq 0}$

ansatz: $c_v = \sqrt{p_\theta(v)}$

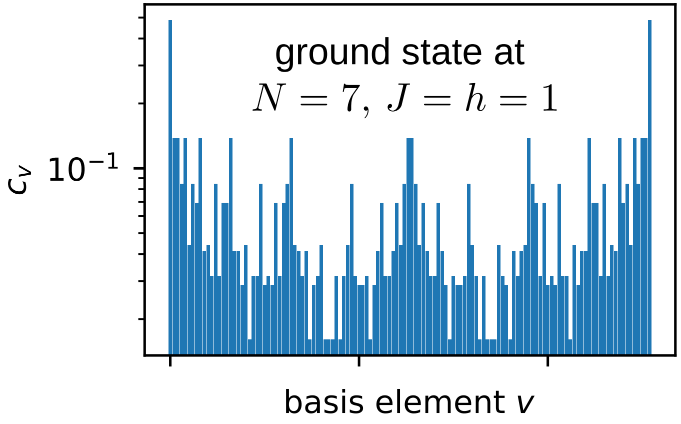
Encoding the quantum states

$$H_{\text{TFIM}} = -J \sum_{\langle i,j \rangle}^{N} \sigma_z^i \sigma_z^j - h \sum_{i}^{N} \sigma_x^i$$

1D, periodic boundary conditions

stoquastic H → $c_v \in \mathbb{R}_{\geq 0}$

ansatz: $c_v = \sqrt{p_\theta(v)}$



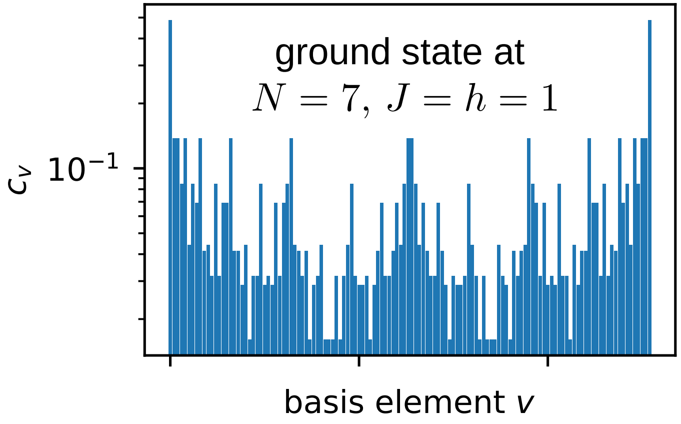
Encoding the quantum states

$$H_{\mathrm{TFIM}} = -J \sum_{\langle i,j \rangle}^{N} \sigma_z^i \sigma_z^j - h \sum_{i}^{N} \sigma_x^i$$

1D, periodic boundary conditions

stoquastic H → $c_v \in \mathbb{R}_{\geq 0}$

ansatz: $c_v = \sqrt{p_\theta(v)}$



Learning algorithm

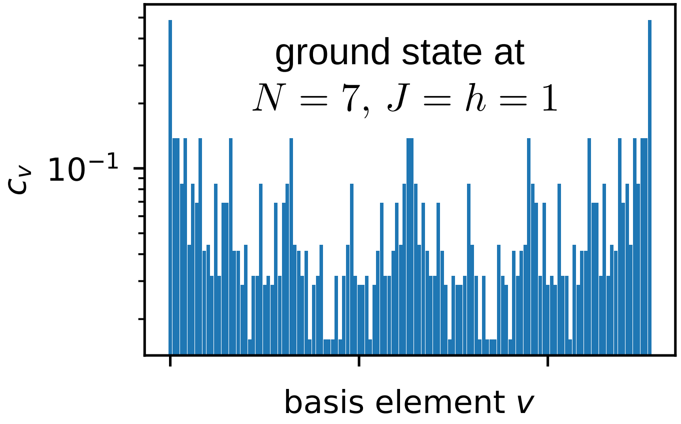
Encoding the quantum states

$$H_{\mathrm{TFIM}} = -J \sum_{\langle i,j \rangle}^{N} \sigma_z^i \sigma_z^j - h \sum_{i}^{N} \sigma_x^i$$

1D, periodic boundary conditions

stoquastic H → $c_v \in \mathbb{R}_{\geq 0}$

ansatz: $c_v = \sqrt{p_\theta(v)}$



Learning algorithm

(1) Calibrate chip, initialize θ

(2) Sample $v, h \sim p_\theta$

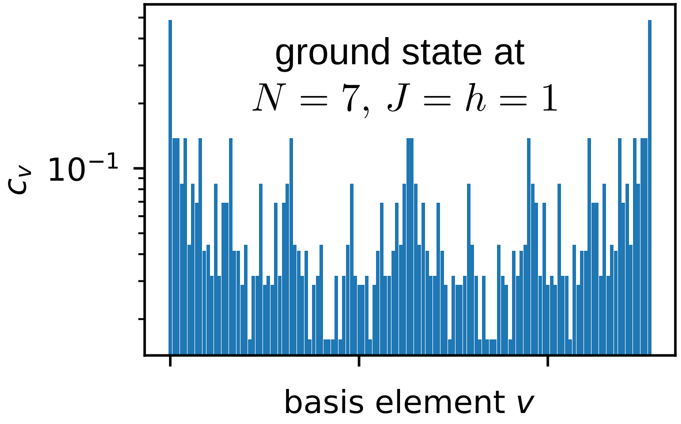
Encoding the quantum states

$$H_{\mathrm{TFIM}} = -J \sum_{\langle i,j \rangle}^{N} \sigma_z^i \sigma_z^j - h \sum_{i}^{N} \sigma_x^i$$

1D, periodic boundary conditions

stoquastic H → $c_v \in \mathbb{R}_{\geq 0}$

ansatz: $c_v = \sqrt{p_\theta(v)}$



Learning algorithm

(1) Calibrate chip, initialize θ

(2) Sample $v, h \sim p_\theta$

} BSS2

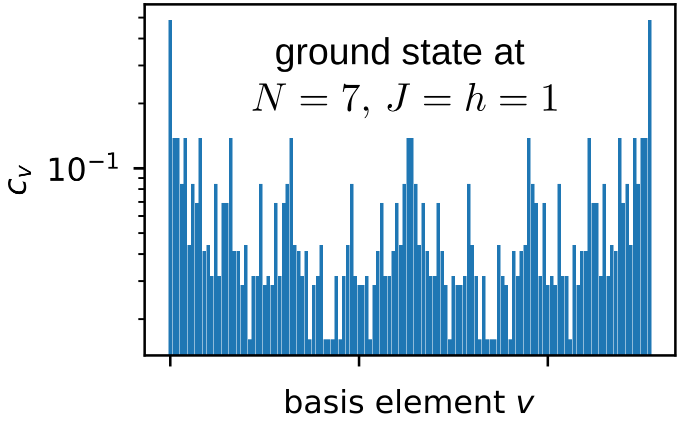
Encoding the quantum states

$$H_{\text{TFIM}} = -J \sum_{\langle i,j \rangle}^{N} \sigma_z^i \sigma_z^j - h \sum_{i}^{N} \sigma_x^i$$

1D, periodic boundary conditions

stoquastic H → $c_v \in \mathbb{R}_{\geq 0}$

ansatz: $c_v = \sqrt{p_\theta(v)}$



Learning algorithm

(1) Calibrate chip, initialize θ
(2) Sample $v, h \sim p_\theta$
} BSS2

(3) Calculate $\Delta\theta = \partial_\theta E_\theta$

(4) Update $\theta \leftarrow \theta - \eta \Delta\theta$

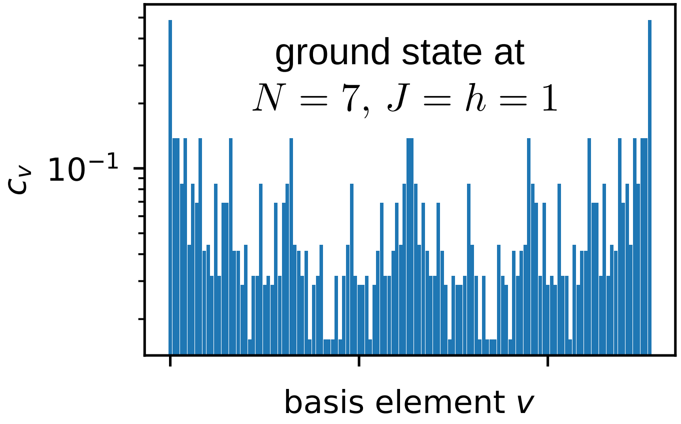
Encoding the quantum states

$$H_{\mathrm{TFIM}} = -J \sum_{\langle i,j \rangle}^{N} \sigma_z^i \sigma_z^j - h \sum_{i}^{N} \sigma_x^i$$

1D, periodic boundary conditions

stoquastic H $\rightarrow$ $c_v \in \mathbb{R}_{\geq 0}$

ansatz: $c_v = \sqrt{p_\theta(v)}$



Learning algorithm

(1) Calibrate chip, initialize θ
(2) Sample $v, h \sim p_\theta$
BSS2

(3) Calculate $\Delta\theta = \partial_\theta E_\theta$
(4) Update $\theta \leftarrow \theta - \eta\Delta\theta$
Host

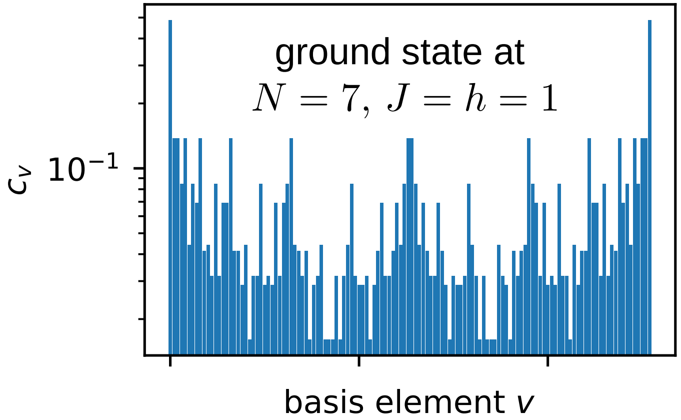
Encoding the quantum states

$$H_{\mathrm{TFIM}} = -J \sum_{\langle i,j \rangle}^{N} \sigma_z^i \sigma_z^j - h \sum_{i}^{N} \sigma_x^i$$

1D, periodic boundary conditions

stoquastic H $\rightarrow$ $c_v \in \mathbb{R}_{\geq 0}$

ansatz: $c_v = \sqrt{p_\theta(v)}$



Learning algorithm

(1) Calibrate chip, initialize θ

(2) Sample $v, h \sim p_\theta$

BSS2

(3) Calculate $\Delta\theta = \partial_\theta E_\theta$

(4) Update $\theta \leftarrow \theta - \eta \Delta\theta$

Host

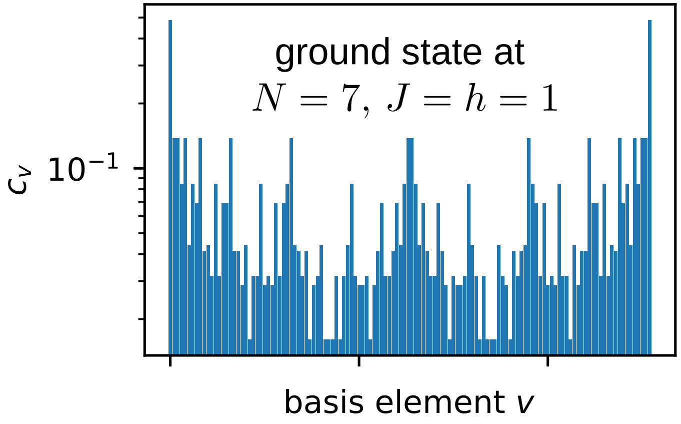
Encoding the quantum states

$$H_{\text{TFIM}} = -J \sum_{\langle i,j \rangle}^{N} \sigma_z^i \sigma_z^j - h \sum_{i}^{N} \sigma_x^i$$

1D, periodic boundary conditions

stoquastic H $\rightarrow$ $c_v \in \mathbb{R}_{\geq 0}$

ansatz: $c_v = \sqrt{p_\theta(v)}$



Learning algorithm

(1) Calibrate chip, initialize θ
(2) Sample $v, h \sim p_\theta$
BSS2

(3) Calculate $\Delta\theta = \partial_\theta E_\theta$
(4) Update $\theta \leftarrow \theta - \eta \Delta\theta$
Host

variational energy

$$E_\theta = \langle \psi_\theta | H | \psi_\theta \rangle = \sum_{v,v'} H_{v,v'} \sqrt{p_\theta(v) p_\theta(v')}$$

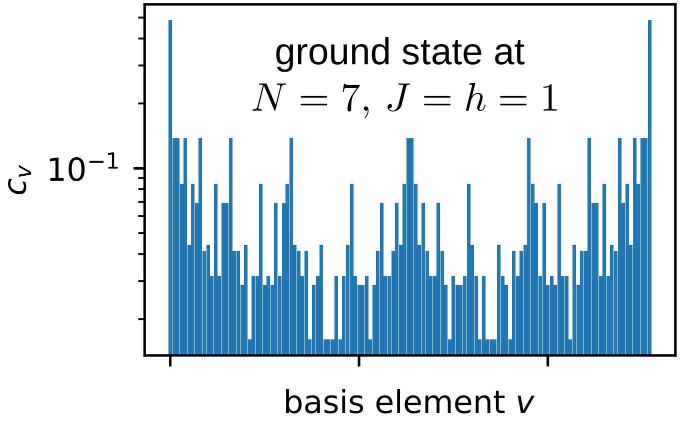
Encoding the quantum states

$$H_{\mathrm{TFIM}} = -J \sum_{\langle i,j \rangle}^{N} \sigma_z^i \sigma_z^j - h \sum_{i}^{N} \sigma_x^i$$

1D, periodic boundary conditions

stoquastic H → $c_v \in \mathbb{R}_{\geq 0}$

ansatz: $c_v = \sqrt{p_\theta(v)}$



Learning algorithm

(1) Calibrate chip, initialize θ
(2) Sample $v, h \sim p_\theta$
} BSS2

(3) Calculate $\Delta\theta = \partial_\theta E_\theta$
(4) Update $\theta \leftarrow \theta - \eta \Delta\theta$
} Host

variational energy

$$E_\theta = \langle \psi_\theta | H | \psi_\theta \rangle = \sum_{v,v'} H_{v,v'} \sqrt{p_\theta(v) p_\theta(v')}$$

gradient from sample average

$$\partial_{W_{ij}} E_\theta = \langle E_v^{\mathrm{loc}} v_i h_j \rangle_{p_\theta} - E_\theta \langle v_i h_j \rangle_{p_\theta}$$

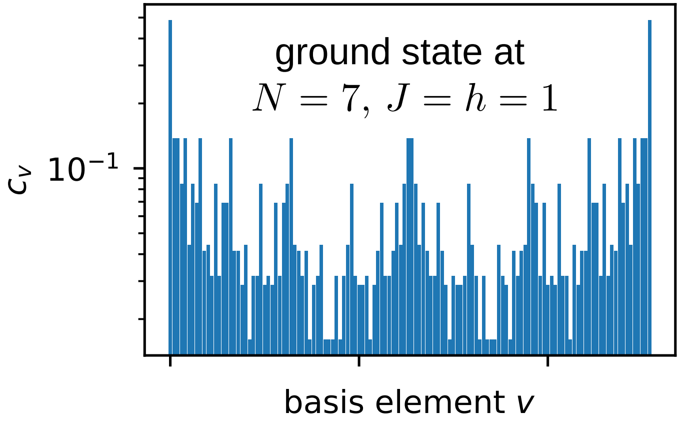
Encoding the quantum states

$$H_{\mathrm{TFIM}} = -J \sum_{\langle i,j \rangle}^{N} \sigma_z^i \sigma_z^j - h \sum_{i}^{N} \sigma_x^i$$

1D, periodic boundary conditions

stoquastic H → $c_v \in \mathbb{R}_{\geq 0}$

ansatz: $c_v = \sqrt{p_\theta(v)}$



Learning algorithm

(1) Calibrate chip, initialize θ
(2) Sample $v, h \sim p_\theta$
BSS2
(3) Calculate $\Delta\theta = \partial_\theta E_\theta$
(4) Update $\theta \leftarrow \theta - \eta\Delta\theta$
Host

variational energy

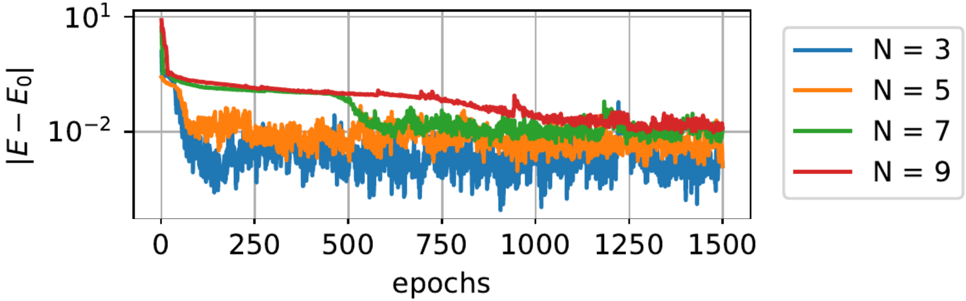
$$E_\theta = \langle \psi_\theta | H | \psi_\theta \rangle = \sum_{v,v'} H_{v,v'} \sqrt{p_\theta(v) p_\theta(v')}$$

gradient from sample average

$$\partial_{W_{ij}} E_\theta = \langle E_v^{\mathrm{loc}} v_i h_j \rangle_{p_\theta} - E_\theta \langle v_i h_j \rangle_{p_\theta}$$

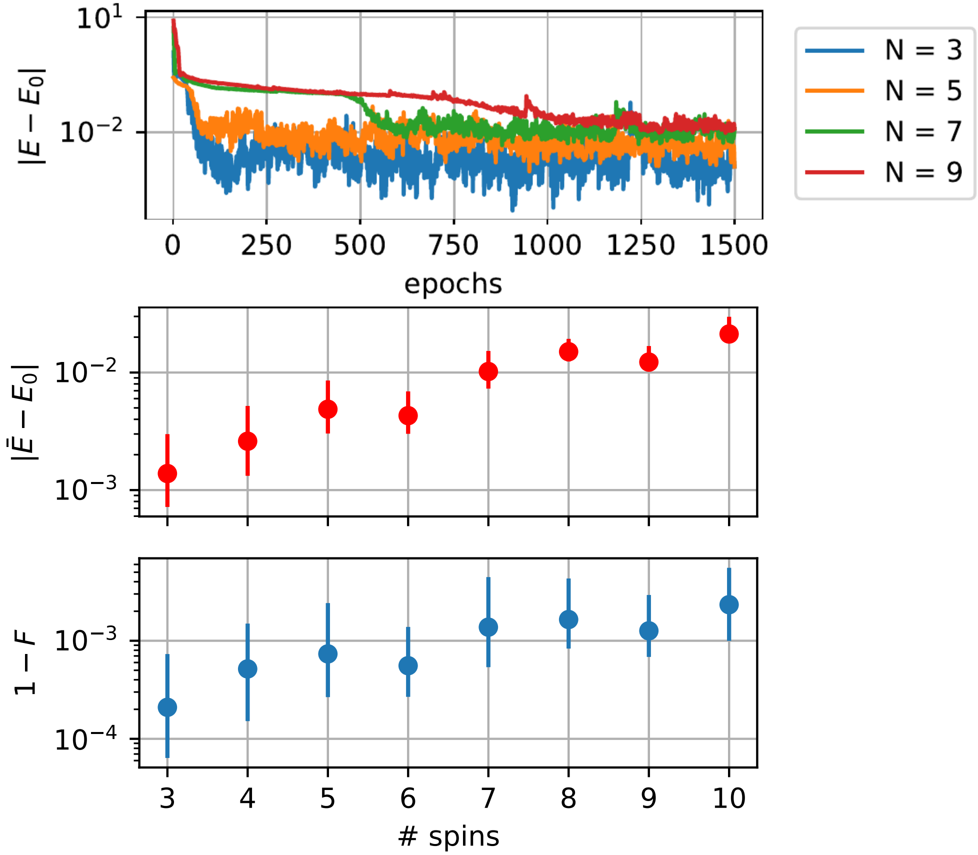
$$\mathrm{E}_v^{\mathrm{loc}} = \frac{\sum_v H_{v,v'} \sqrt{p_\theta(v')}}{\sqrt{p_\theta(v)}}$$

Results



all figures at critical point $J = h = 1$

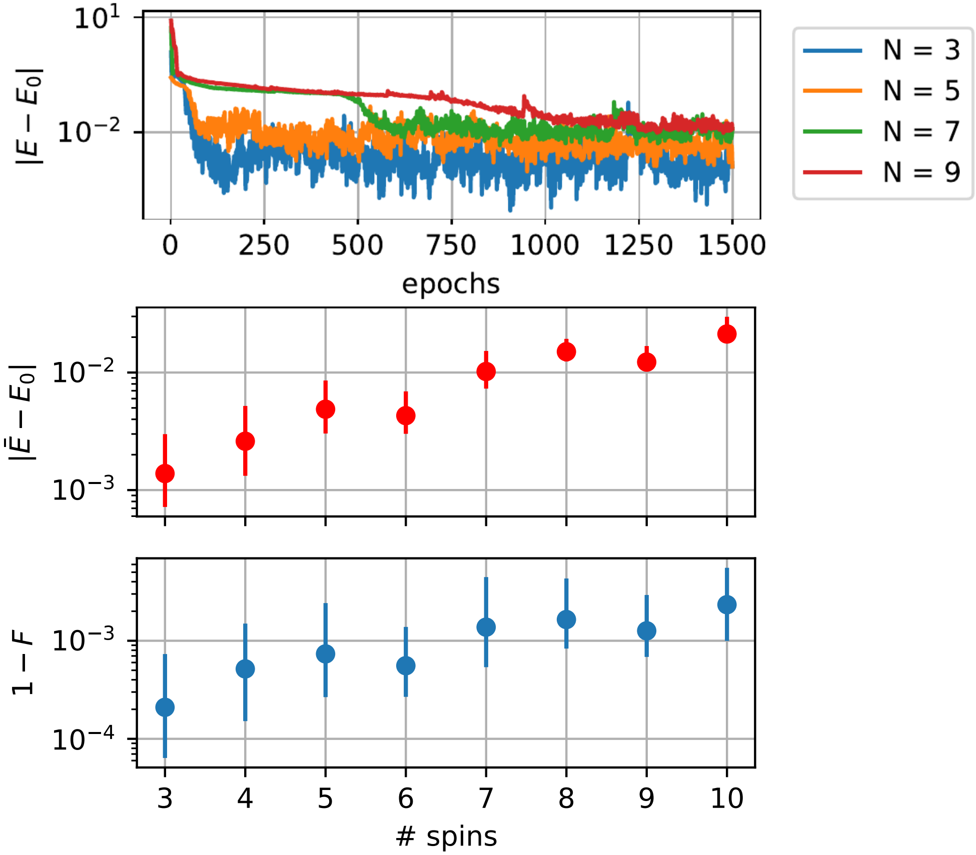
Results



all figures at critical point $J = h = 1$

fidelity $F = \sqrt{|\langle \psi_0 | \psi_\theta \rangle|}$

Results



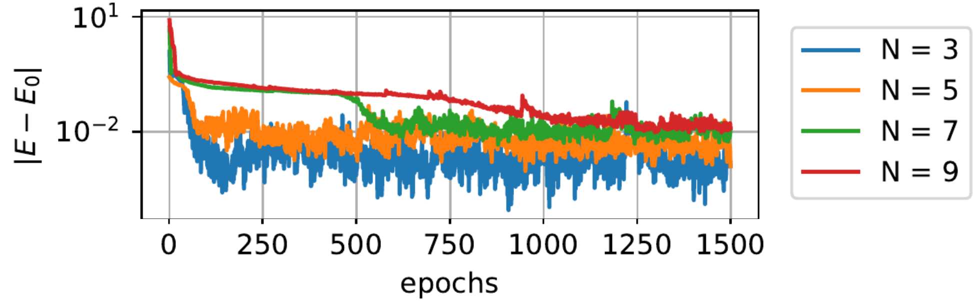
all figures at critical point $J = h = 1$

fidelity $F = \sqrt{|\langle \psi_0 | \psi_\theta \rangle|}$

$\{3, ..., 8\}$: $N_h = 40$, $N_{\text{sample}} = 2 \cdot 10^5$

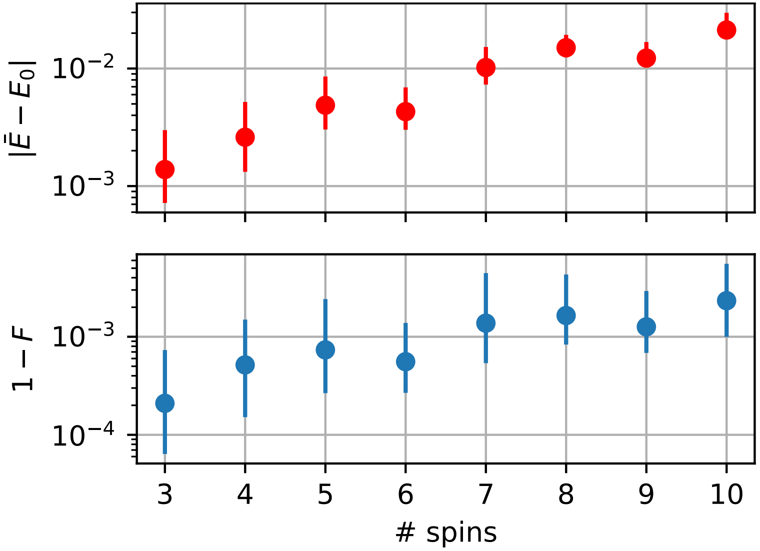
$\{9, 10\}$: $N_h = 50$, $N_{\text{sample}} = 4 \cdot 10^5$

Results

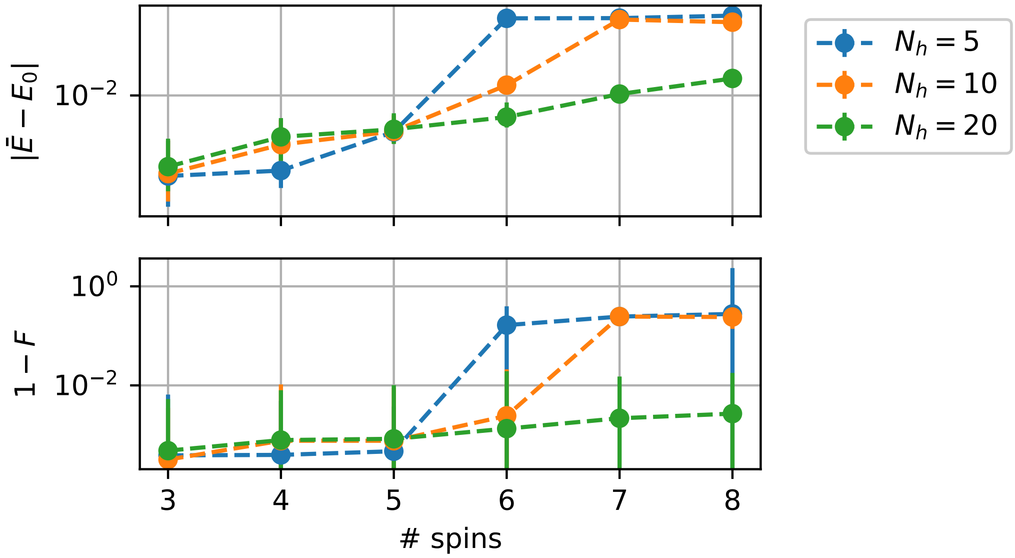


all figures at critical point $J = h = 1$

fidelity $F = \sqrt{|\langle \psi_0 | \psi_\theta \rangle|}$



$\{3, ..., 8\}$: $N_h = 40,\ N_{\text{sample}} = 2 \cdot 10^5$
$\{9, 10\}$: $N_h = 50,\ N_{\text{sample}} = 4 \cdot 10^5$



$\text{N}_{\text{sample}} = 2 \cdot 10^5$

Summary

Summary

- Neural sampling with spiking neural networks

Summary

- Neural sampling with spiking neural networks
- Gradient-based host-in-the-loop learning algorithm

Summary

- Neural sampling with spiking neural networks
- Gradient-based host-in-the-loop learning algorithm
- Variational energy minimization for 1D TFIM

Summary

Future directions

- Neural sampling with spiking neural networks
- Gradient-based host-in-the-loop learning algorithm
- Variational energy minimization for 1D TFIM

Summary

- Neural sampling with spiking neural networks
- Gradient-based host-in-the-loop learning algorithm
- Variational energy minimization for 1D TFIM

Future directions

- Steady state search with POVMs on BSS2

Summary

- Neural sampling with spiking neural networks
- Gradient-based host-in-the-loop learning algorithm
- Variational energy minimization for 1D TFIM

Future directions

- Steady state search with POVMs on BSS2
- Utilize on-chip learning

Summary

- Neural sampling with spiking neural networks
- Gradient-based host-in-the-loop learning algorithm
- Variational energy minimization for 1D TFIM

Future directions

- Steady state search with POVMs on BSS2
- Utilize on-chip learning
- Explore alternative mappings of the spiking dynamics

Summary

- Neural sampling with spiking neural networks
- Gradient-based host-in-the-loop learning algorithm
- Variational energy minimization for 1D TFIM

Thank you!

Future directions

- Steady state search with POVMs on BSS2
- Utilize on-chip learning
- Explore alternative mappings of the spiking dynamics

Twitter: @GaerttnerGroup