Learning Many-body Hamiltonians from Dynamical Data

Frederik Wilde, Augustine Kshetrimayum, Ryan Sweke, Ingo Roth, Jens Eisert

slides at `frederikwil.de/crc2021`

CRC workshop on Machine Learning in Condensed Matter Physics
2021-07-16

Freie Universität Berlin

Hamiltonian Learning

Problem: Given observations about a (closed) quantum system, find its Hamiltonian.

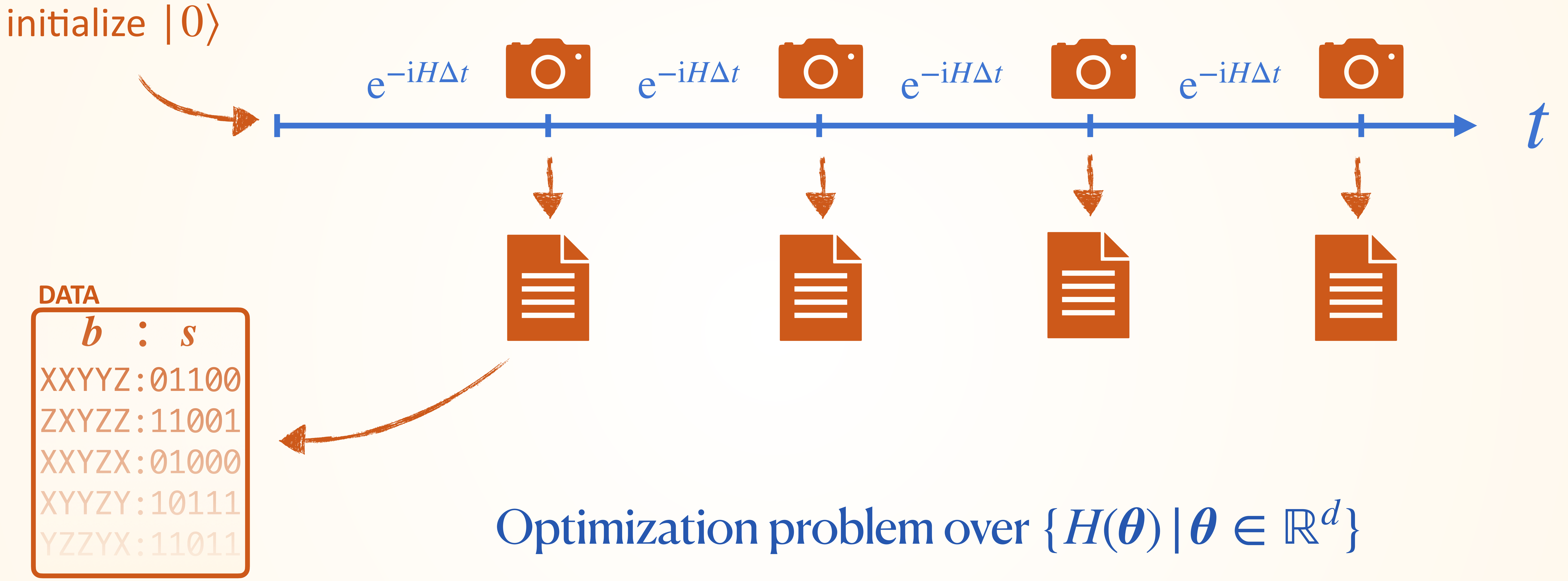
PREVIOUS

- Sequential Monte-Carlo algorithm: $\hat{H} = \mathbb{E}(H\,|\,\text{data}) = \int \mathrm{d}H\ H\,\mathbb{P}(H\,|\,\text{data})$ [Grenade et al.]
- Learning at the pulse level (*time dependent*) [Krastanov et al.]
- Learning from thermal states [Yu et al., Anshu et al.]
- Bayesian learning from steady states [Evans et al.]

RELATED TOPICS

- Learning of properties via *classical shadows* and *neural networks*
- *Randomized benchmarking*
- *State tomography* and *process tomography*

Setting



Optimization problem over $\{H(\boldsymbol{\theta}) \,|\, \boldsymbol{\theta} \in \mathbb{R}^d\}$

LOSS FUNCTION

$$L(\boldsymbol{\theta}) = -\sum_x \log \mathbb{P}(x \,|\, \boldsymbol{\theta}), \quad x = (t, \boldsymbol{b}, \boldsymbol{s})$$

Learning

LOSS FUNCTION

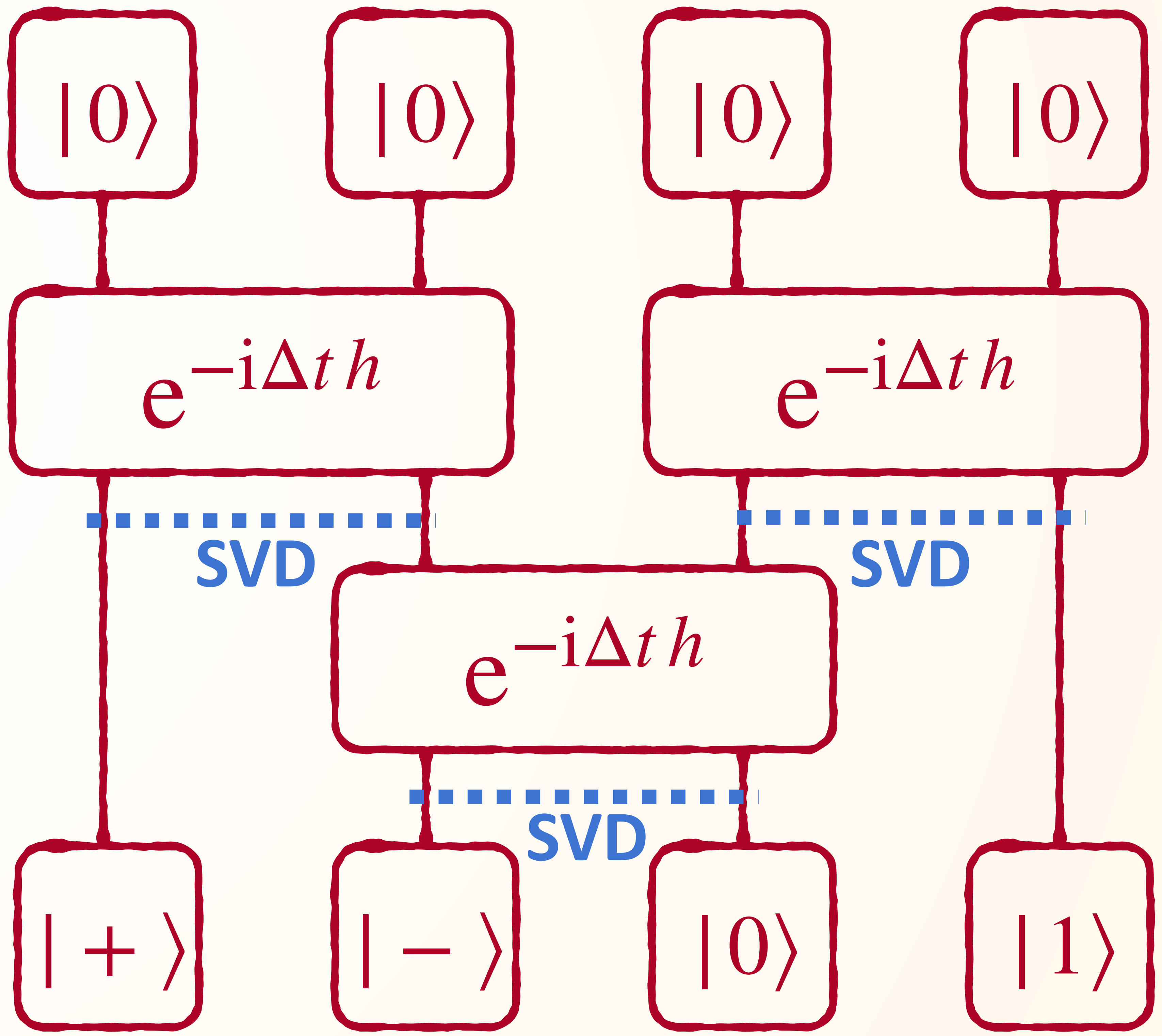
$$L(\boldsymbol{\theta}) = -\sum_x \log \mathbb{P}(x\,|\,\boldsymbol{\theta}), \quad x = (t, \boldsymbol{b}, \boldsymbol{s})$$

$$\mathbb{P}(t, \boldsymbol{b}, \boldsymbol{s}\,|\,\boldsymbol{\theta}) = \left| \langle \boldsymbol{s_b} \,|\, \mathrm{e}^{-\mathrm{i}tH(\boldsymbol{\theta})} \,|\, 0 \rangle \right|^2$$

e.g. $\boldsymbol{b} = (XXZZ)$ and $\boldsymbol{s} = (0101)$:

$|s_b\rangle = |+-0\,1\rangle$

Use Time-Evolving Block Decimation (TEBD) to compute probabilities efficiently



STRATEGY

Minimize L via gradient-descent.

➡ Use *Automatic Differentiation*

∇L also requires the differentiation of the singular value decomposition (SVD)

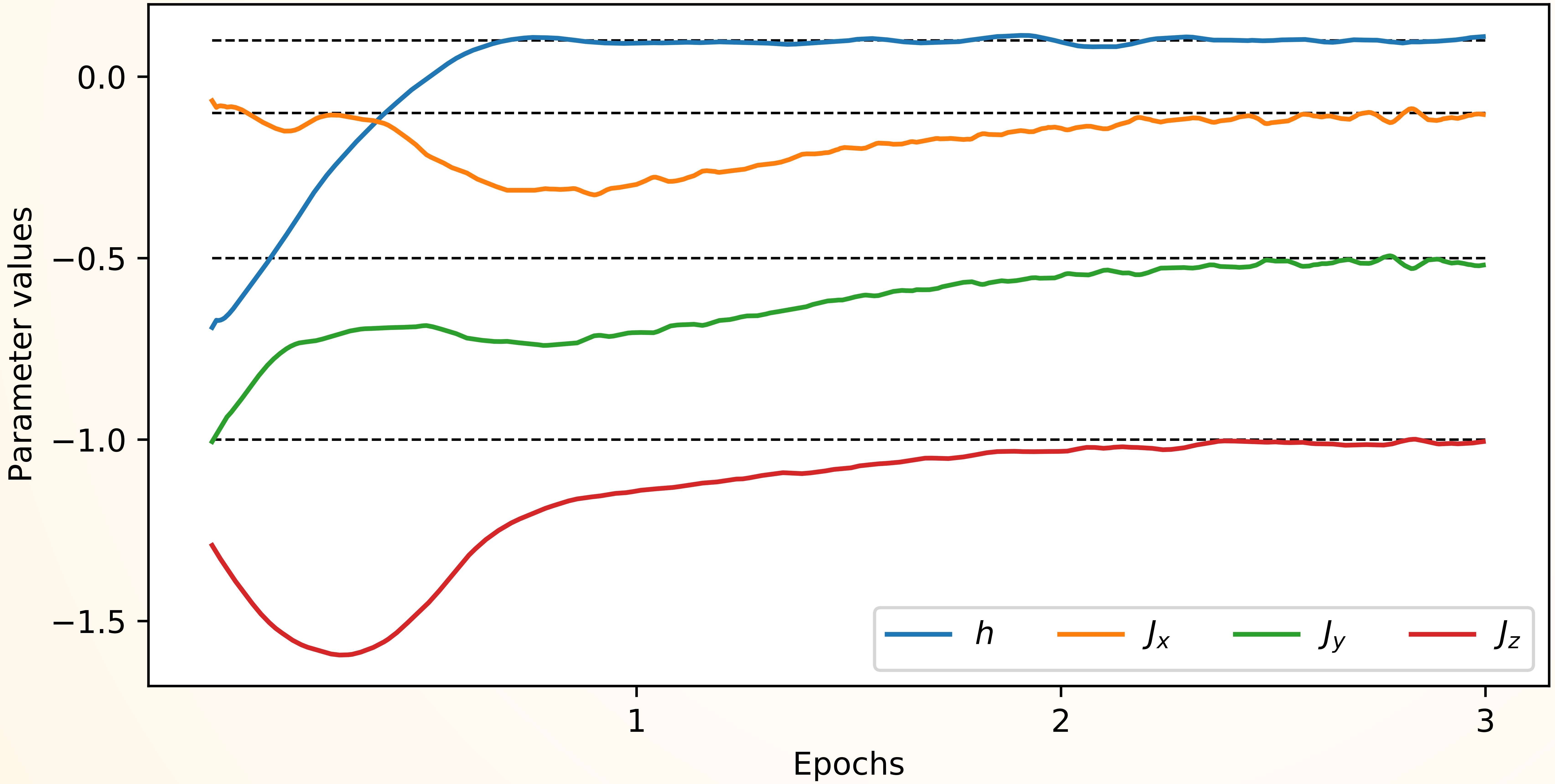
github.com/google/jax

github.com/google/jax/pull/5225

Results

$$H = \sum_i hX_i + J_x X_i X_{i+1} + J_y Y_i Y_{i+1} + J_z Z_i Z_{i+1} \longrightarrow \text{learn } \boldsymbol{\theta} = (h, J_x, J_y, J_z)$$

ADAM optimizer turns out to be most robust



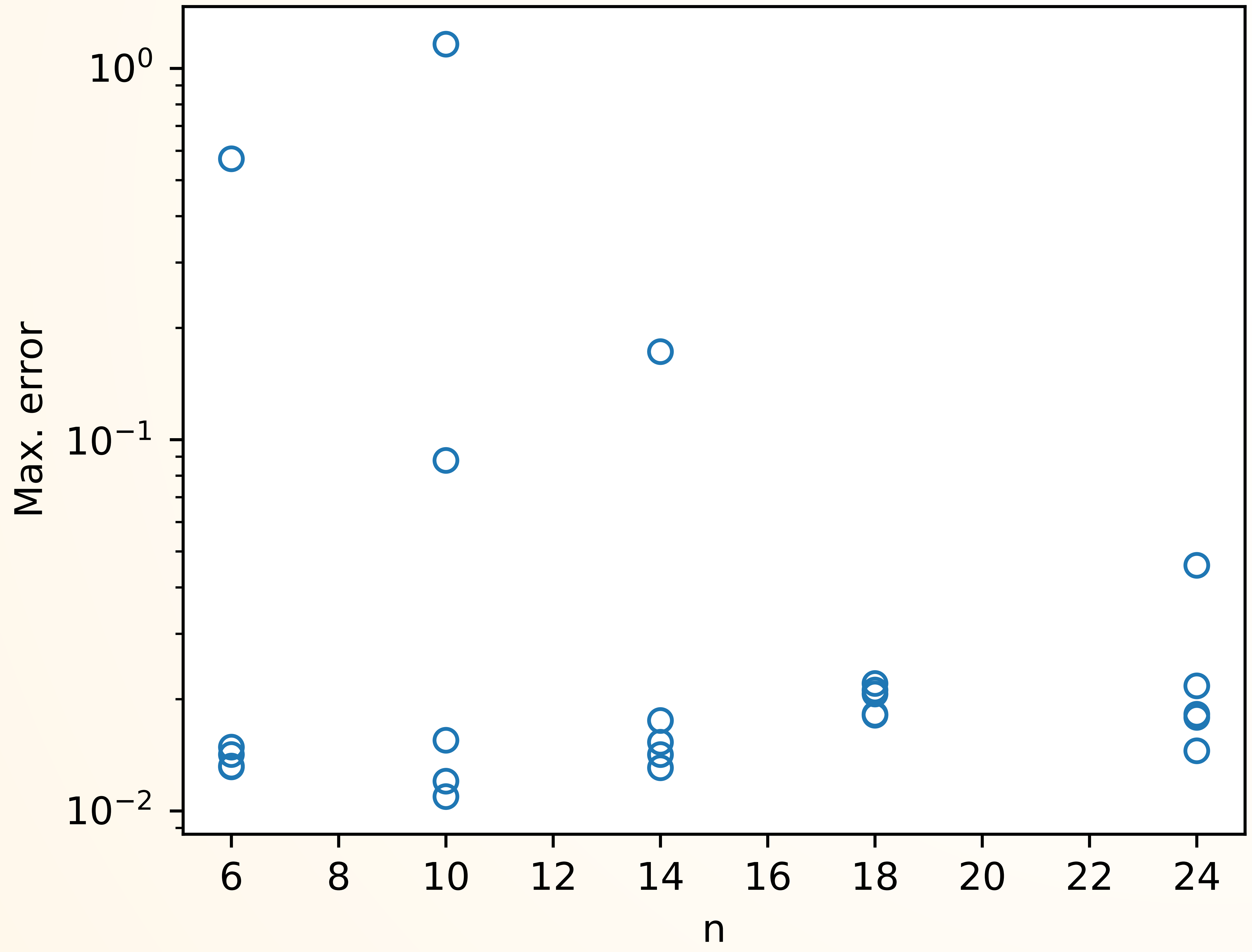
$$n = 10$$
$$\chi = 30$$
$$\Delta t_{\text{Trotter}} = 0.02$$
$$\#(\text{snapshots}) = 10$$
$$\#(\text{samples}) = 10000$$
$$\varepsilon_{\text{trunc}} \approx 0.005$$
$$\varepsilon_{\text{total}} \approx 0.0003$$

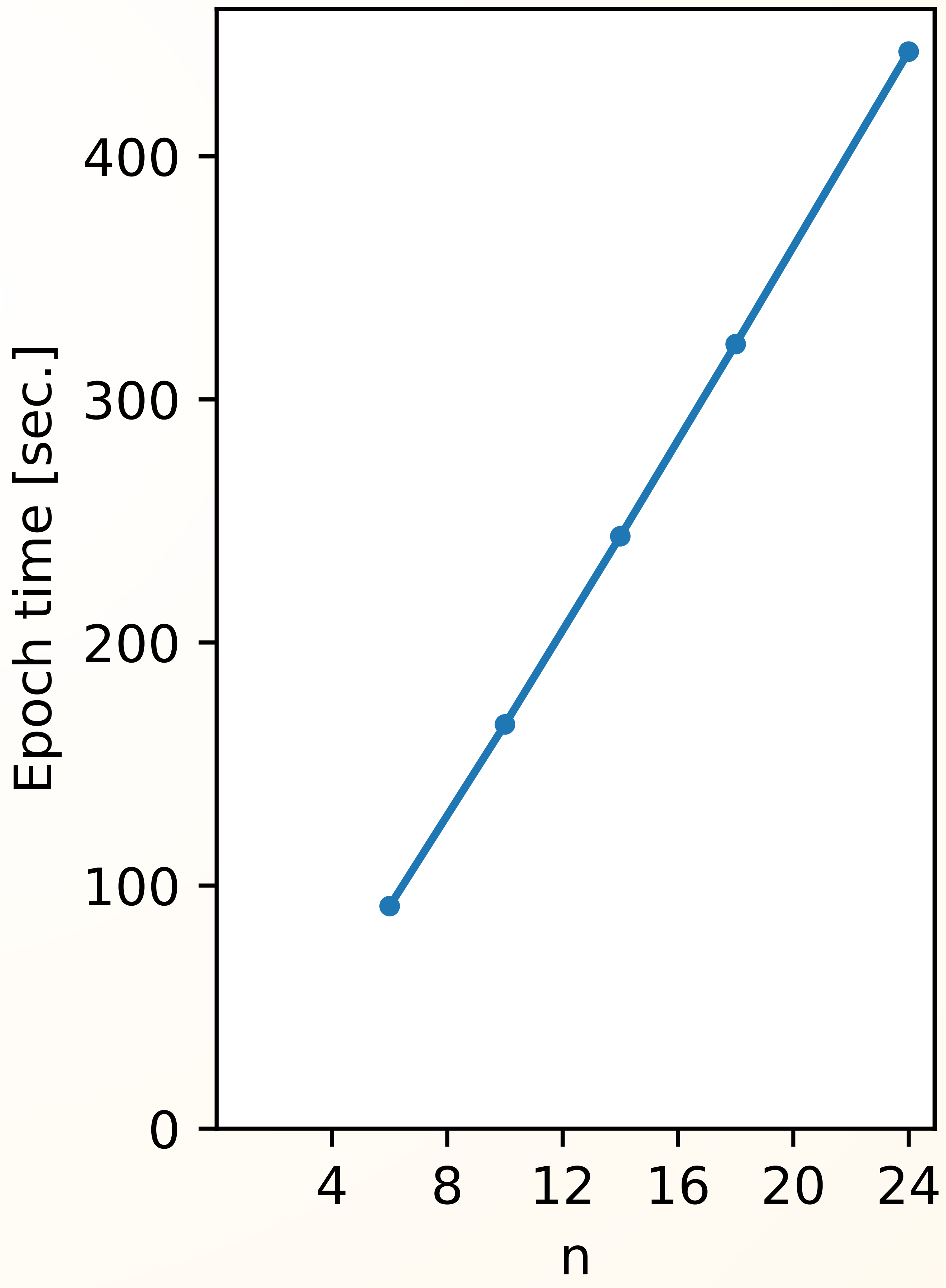
Max. error $:= \left\|\bar{\boldsymbol{\theta}} - \boldsymbol{\theta}_{\text{true}}\right\|_{\infty}$

Scaling

$$H = \sum_i h_i X_i + J_x X_i X_{i+1} + J_y Y_i Y_{i+1} + J_z Z_i Z_{i+1}$$

$\#(\text{parameters}) = n + 3$





$\chi = 30$

$\Delta t_{\text{Trotter}} = 0.02$

$\#(\text{snapshots}) = 5$

$\#(\text{samples}) = 10000$

Thanks

slides at `frederikwil.de/crc2021`