Sampling lattice gauge theory in 3/4D with normalizing flows

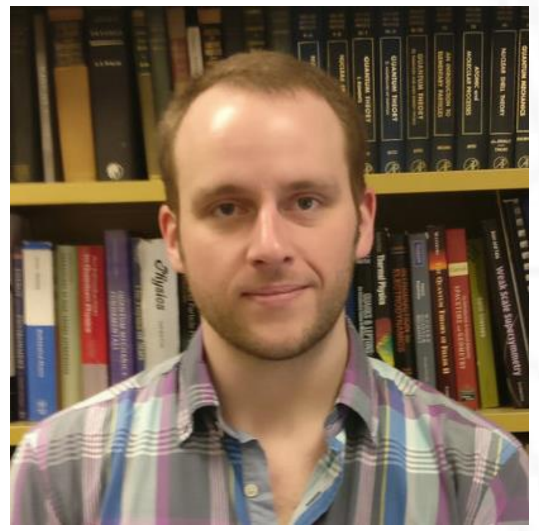
Technical note on boosting performance of normalizing flows

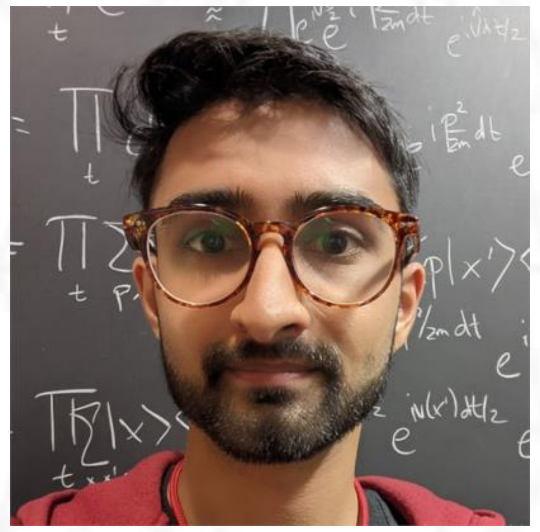
Denis Boyda, ANL
MIT, ANL LCF

Lattice 21, July 26 – 30, zoom/gather@MIT

The team


Phiala Shanahan


Daniel Hackett


Gurtej Kanwar


Denis Boyda

Argonne NATIONAL LABORATORY

Full-time job

DeepMind


Sébastien Racanière


Danilo Rezende

NEW YORK UNIVERSITY


Kyle Cranmer


Michael Albergo

UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386


Julian Urban

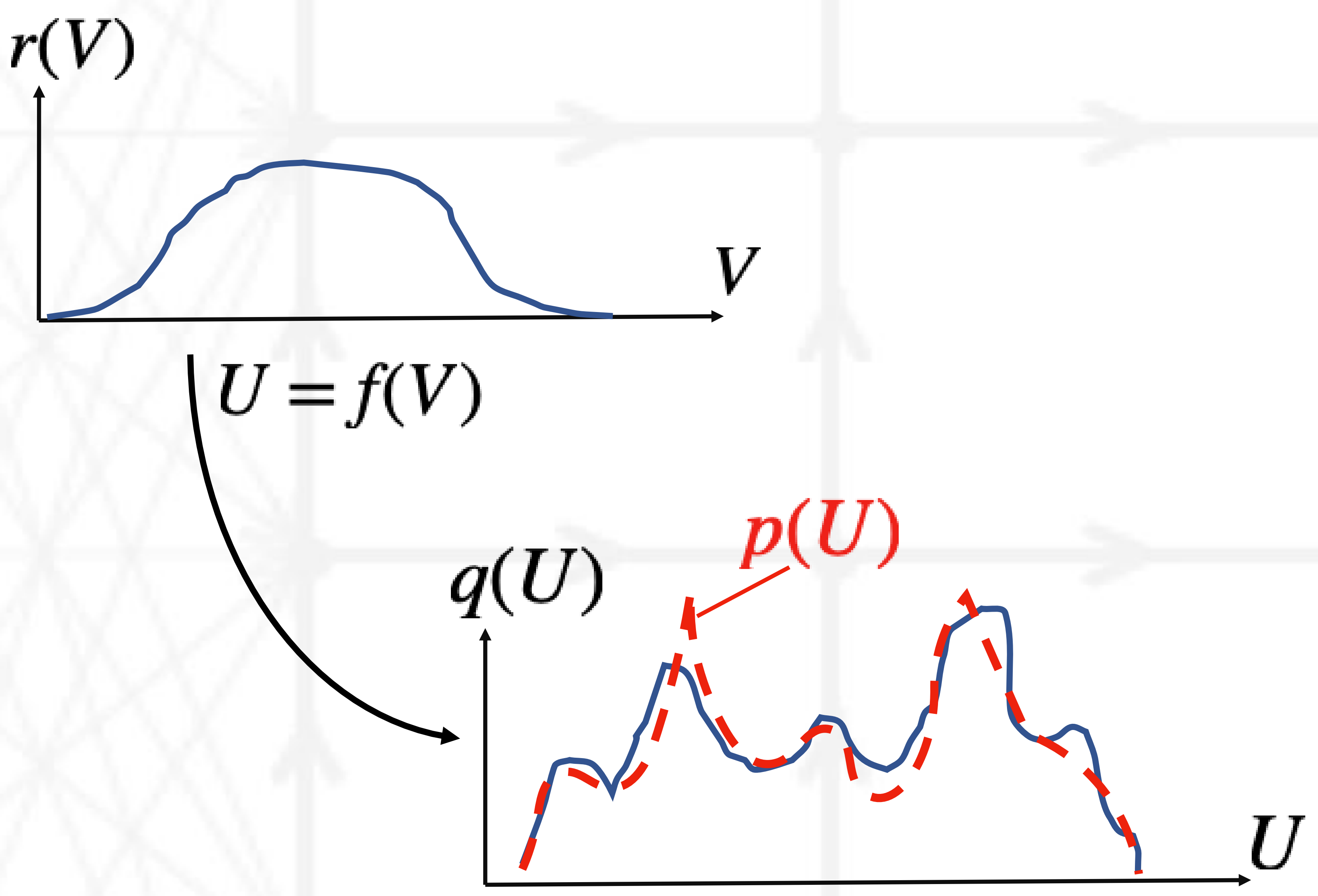
Outline

This talk is about technical optimization of normalizing flows towards improving sampling efficiency in 3/4D

- Intro:
  - Normalizing flows
  - sampling lattice gauge theories
- Masking patterns
- Frozen loops
- ML optimizations

Normalizing flows

Flow-based models learn a change-of-variables that transforms a known distribution to the desired one
[Rezende & Mohamed 1505.05880]



[Image credit: Gurtej Kanwar, Dan Hackett]

- Generate samples V from prior distribution $r(V)$ such that
  - is simple / cheap to draw samples from
  - distribution density is known

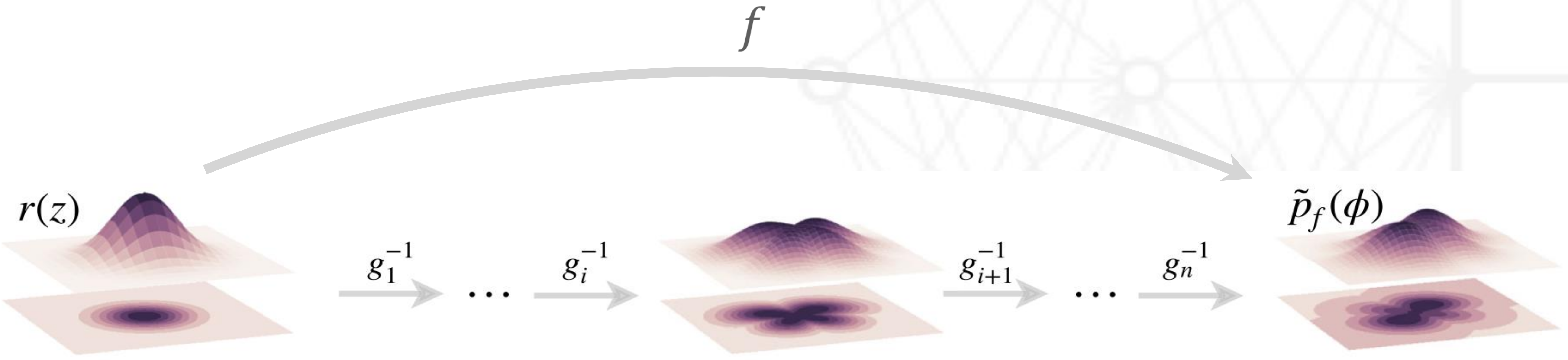
- Transform prior variables with a flow $f_\theta(V)$ which
  - is expressive
  - is invertible
  - has tractable Jacobian

$$\boxed{q(U) = r(V)\left|\det_{ij} \frac{\partial [f(V)]_i}{\partial V_j}\right|^{-1}}$$

- Optimize flow parameters θ to reproduce target distribution $q(U) \approx p(U)$

Coupling-based flows

- Flow is a composition of **coupling layers** $\boldsymbol{g_i}$

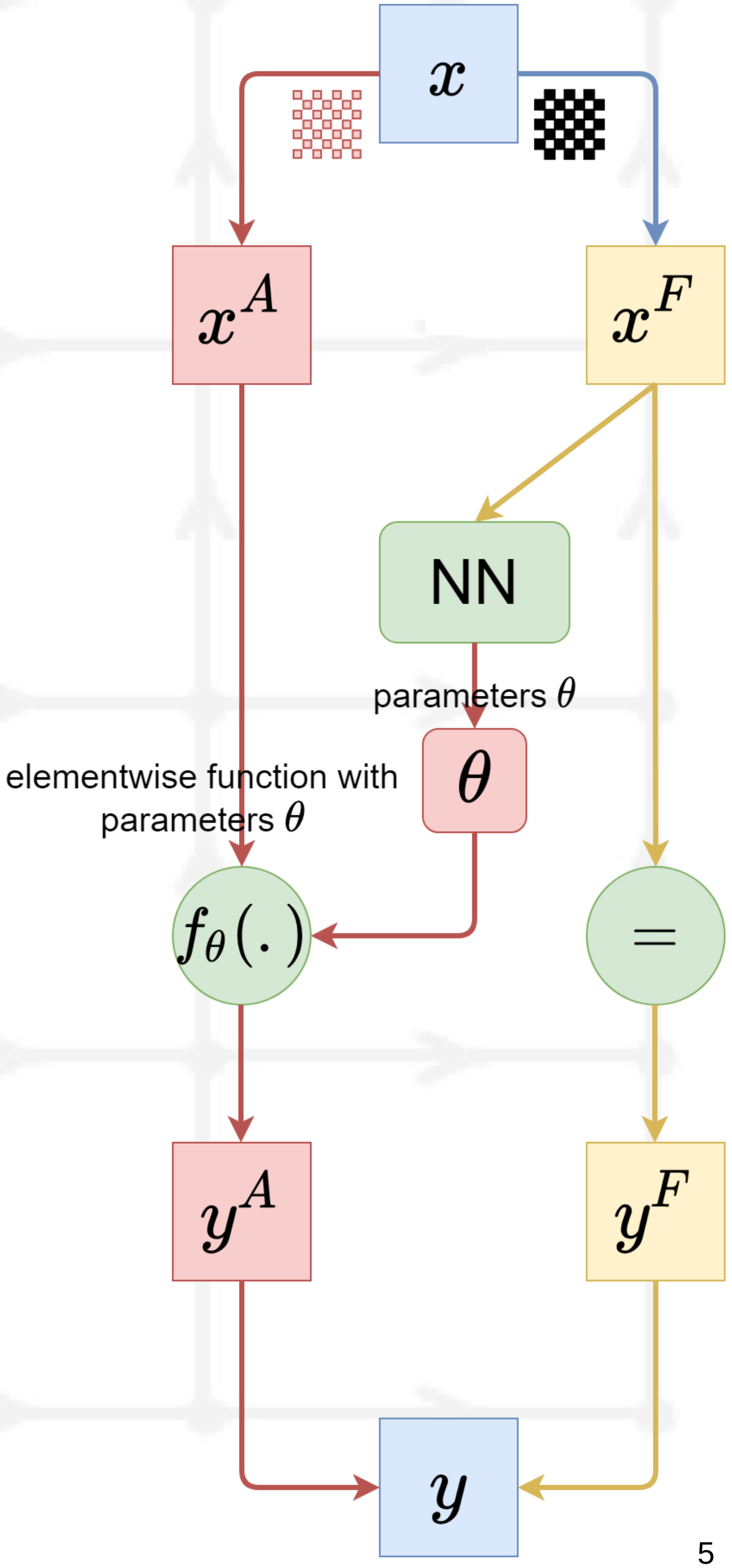


- Introduce **mask** $m_i = \{0,1\}$ such that all elements are split to **frozen** and **active** sets:
  element i is **frozen** when $m_i = 0$
  element i is **active** when $m_i \neq 0$

- Each coupling layer **splits variables** and **transforms** a subset of variables conditioned on a complementary subset

$$\phi = g_i(z) = \begin{cases} y_i = x_i \text{ , when } m_i = 0 \text{ —> } \underline{\textit{frozen}} \\ y_i = f_\theta(x_i) \text{ , when } m_i = 1 \text{ —> } \underline{\textit{active}} \end{cases}$$

parameters of transformations $\boldsymbol{\theta}$ are functions (NN) of **frozen** variables



Gauge-equivariant flows

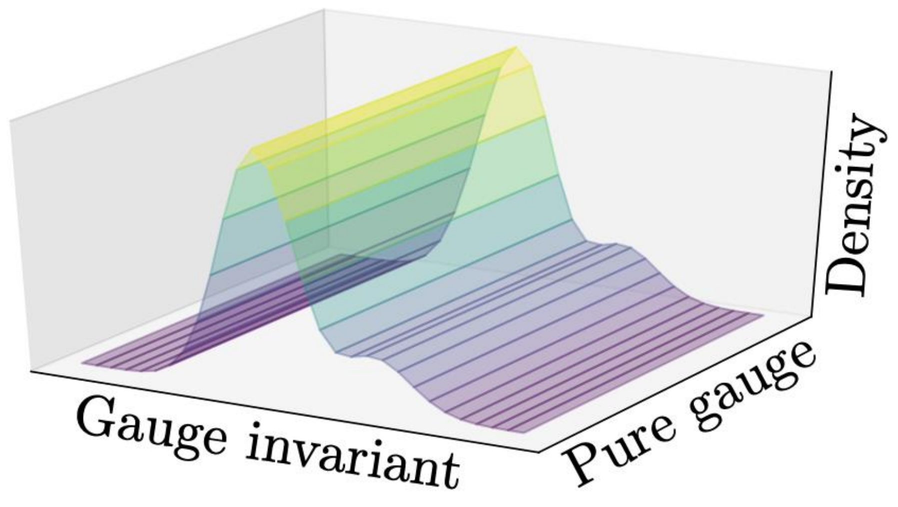
Normalizing flows produce invariant posterior distribution if

- Prior distribution is invariant

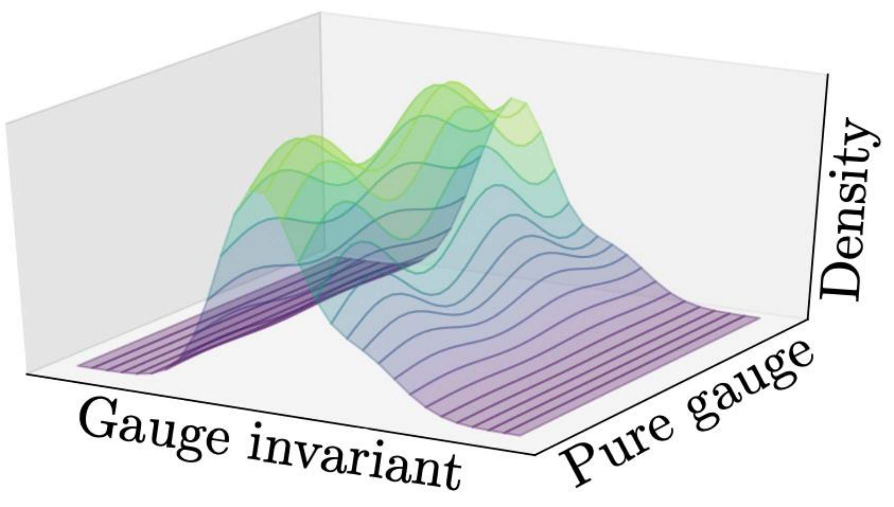
$$r(U) = r(\Omega \circ U)$$

- Flow transformation is equivariant

$$f(U) \to f(\Omega \circ U) = \Omega \circ f(U)$$



exact by model definition



approximately learned

Lattice gauge transformation

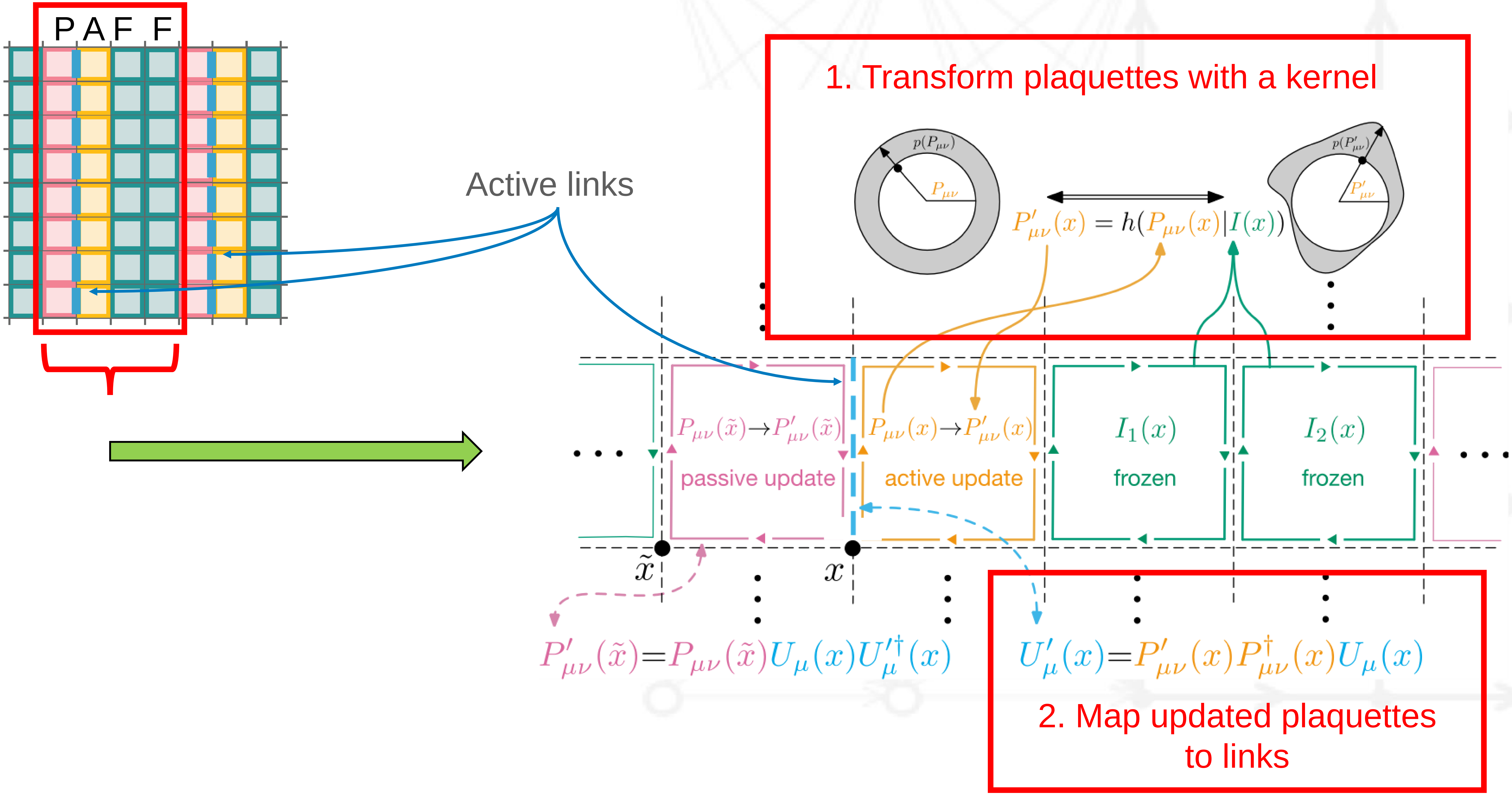
$$U_\mu(x) \to \Omega \circ U_\mu(x) = \Omega(x) U_\mu(x) \Omega(\mathrm{x}+\mu)$$
$$P_{\mu\nu}(x) \to \Omega(x) P_{\mu\nu}(x) \Omega(x)^\dagger$$
$$P_{\mu\nu}(x) U_\mu(x) \to \Omega(x) P_{\mu\nu}(x) U_\mu(x) \Omega(\mathrm{x}+\mu)$$

Idea:

1) transform open plaquettes $P_{\mu\nu} \to P'_{\mu\nu} = g\,(P_{\mu\nu}|I)$
2) and map them to links $U_\mu \to U'_\mu = P'_{\mu\nu} P^\dagger_{\mu\nu} U_\mu$

⇒ One needs to derive **plaquette mask** from **link mask**

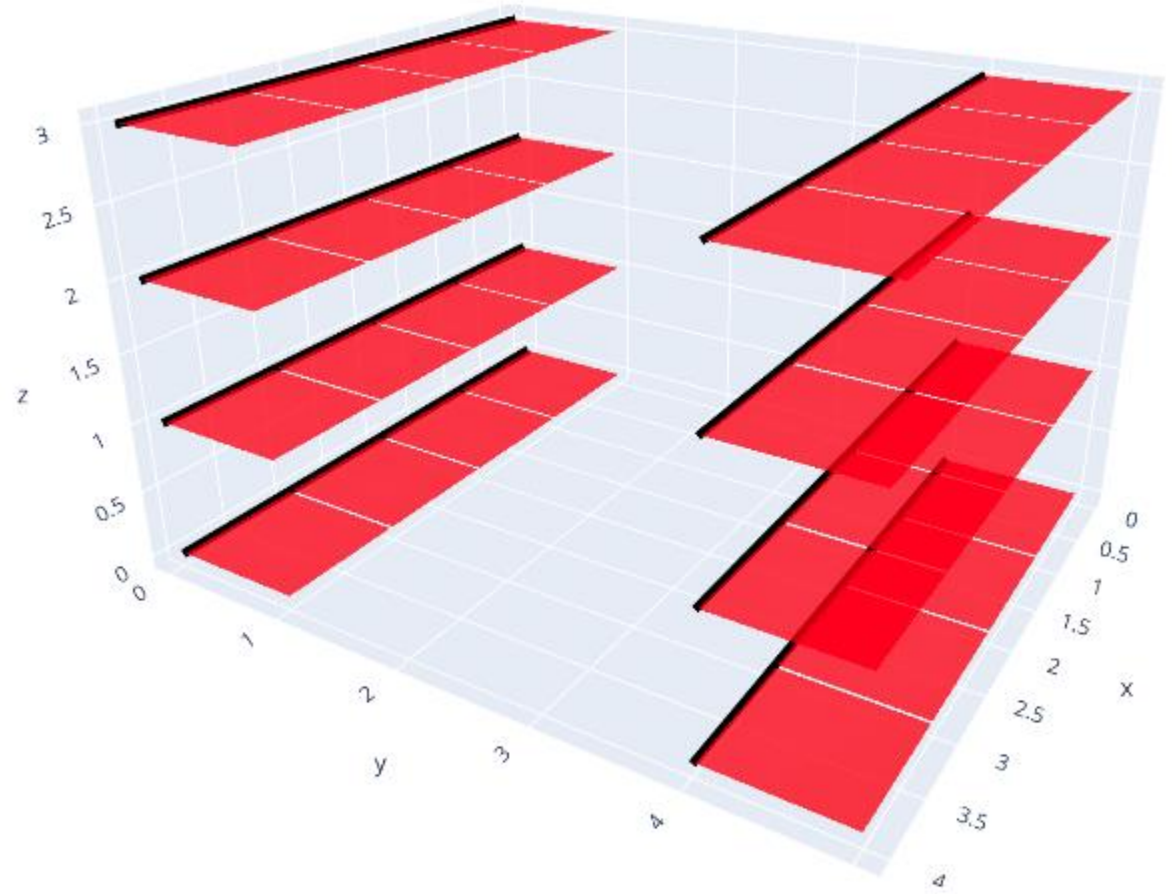
Gauge-equivariant coupling layer

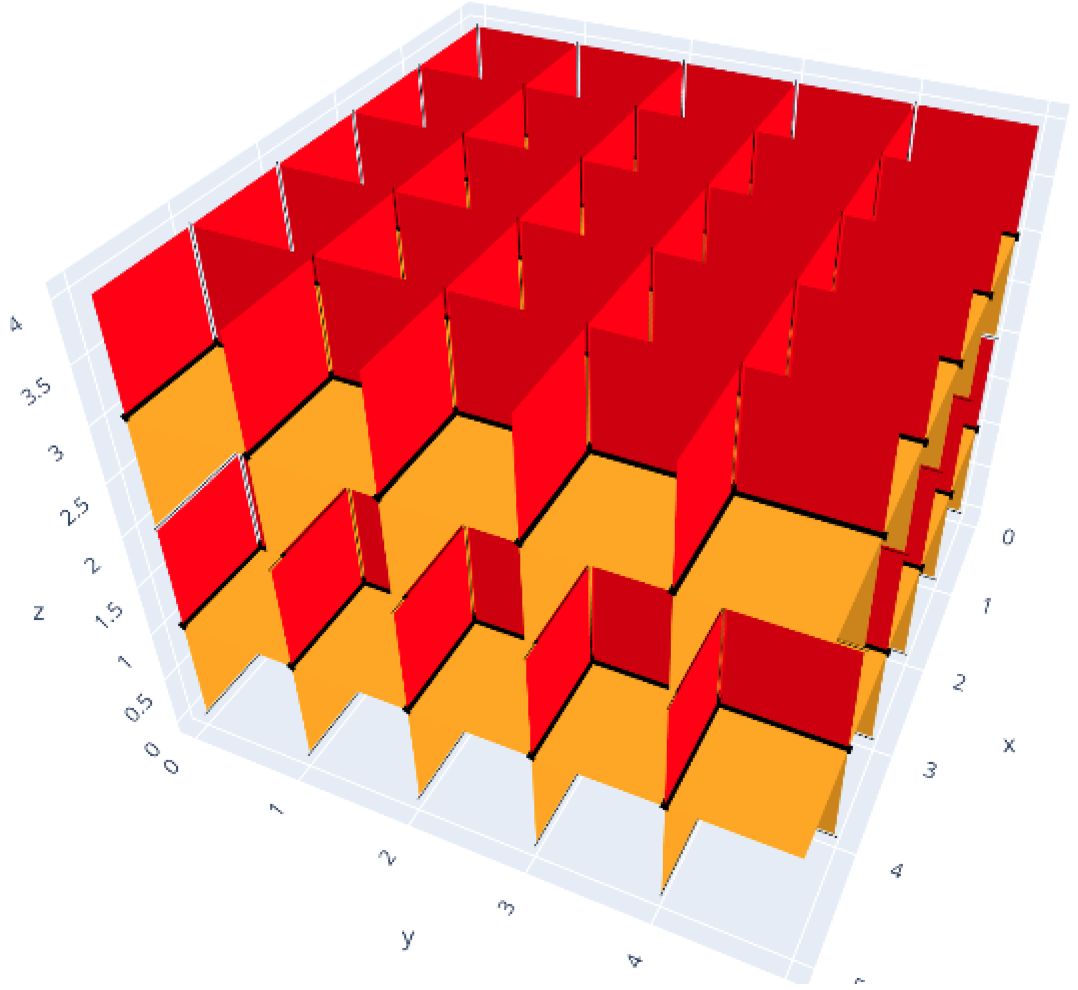


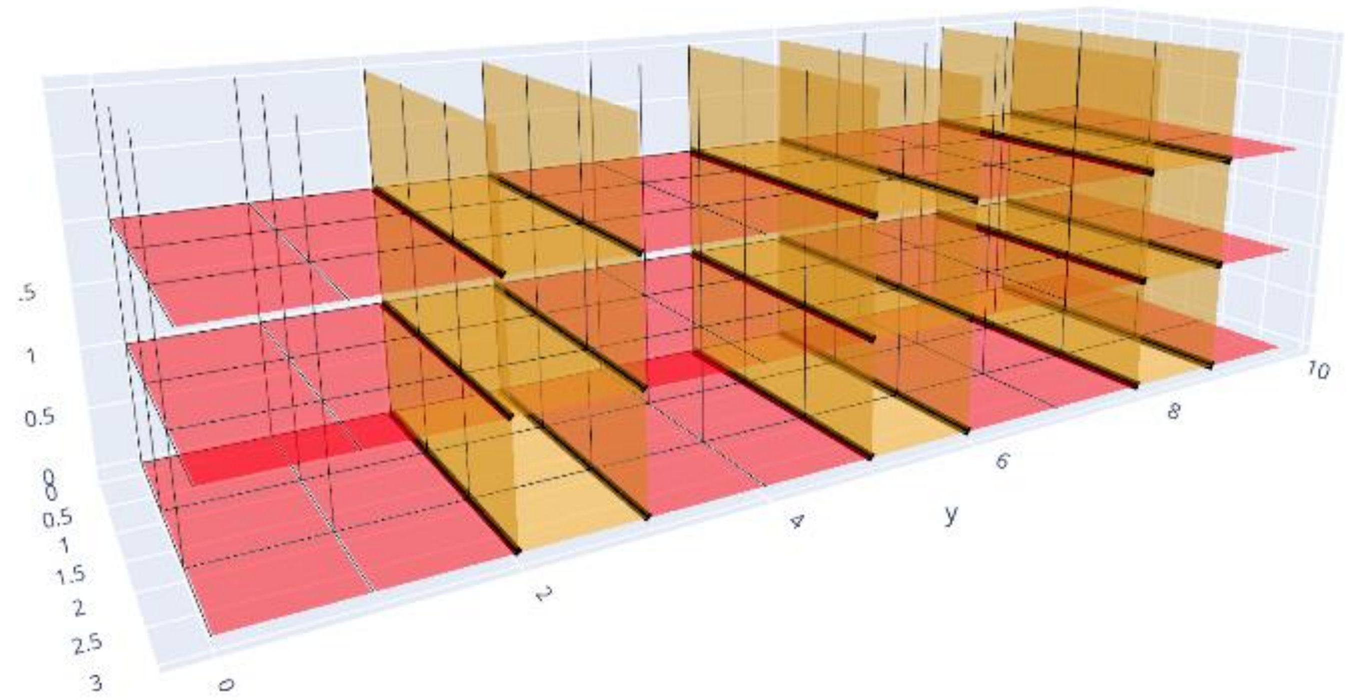
Algorithm for masking pattern

How can we generalize mask for higher dimensions?
Is there better mask?

Active links are shown by **black lines**
Active plaquettes are shown by red color
Frozen plaquettes are either orange or not shown







Requirements for **masking algorithm**:
- generalizable to Nd with few parameters
- allows propagation of information from vicinity of active link to the link
- allows to control sparsity and density

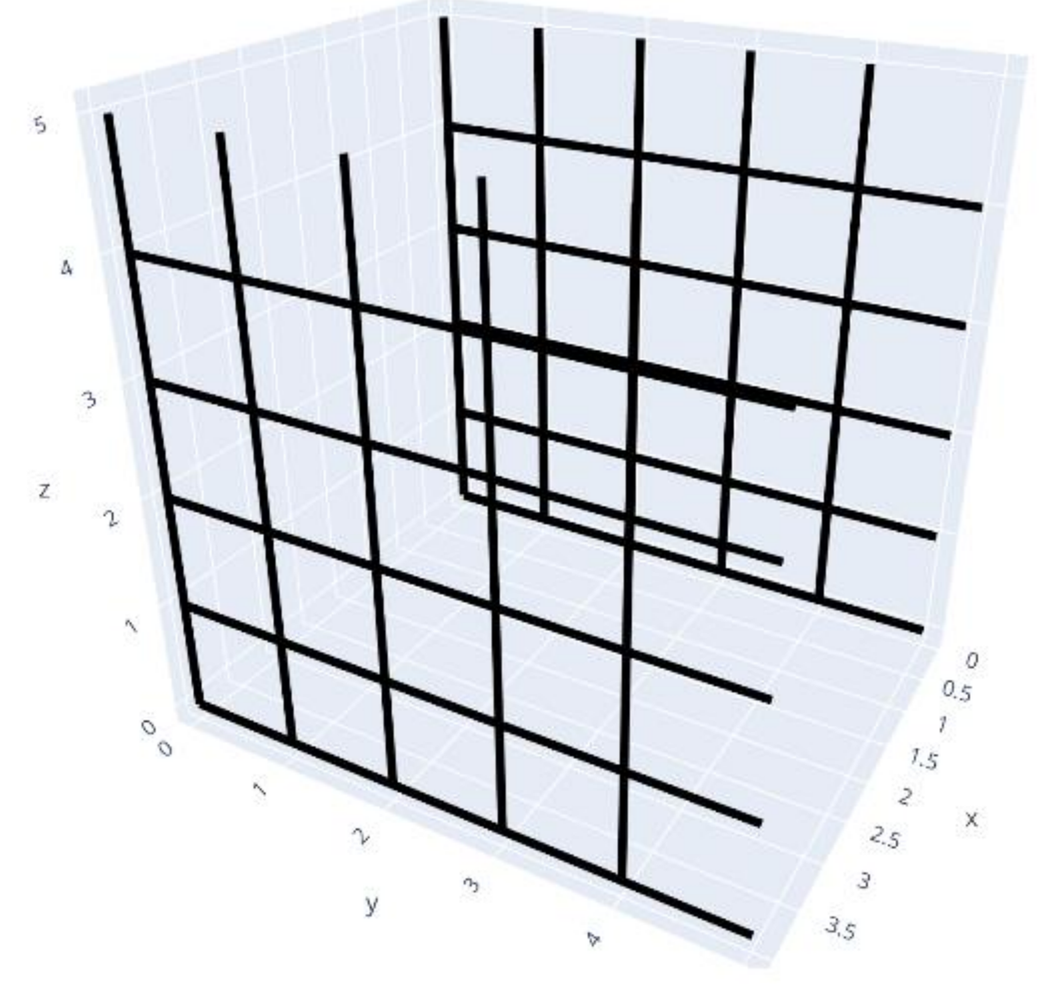
Algorithm for masking pattern

Idea: generalize checkerboard mask
- set *directions* of active links
- every link has a *phase*
- moving in any direction changes phase by *step_mu*
- active links have phase mod width = 0

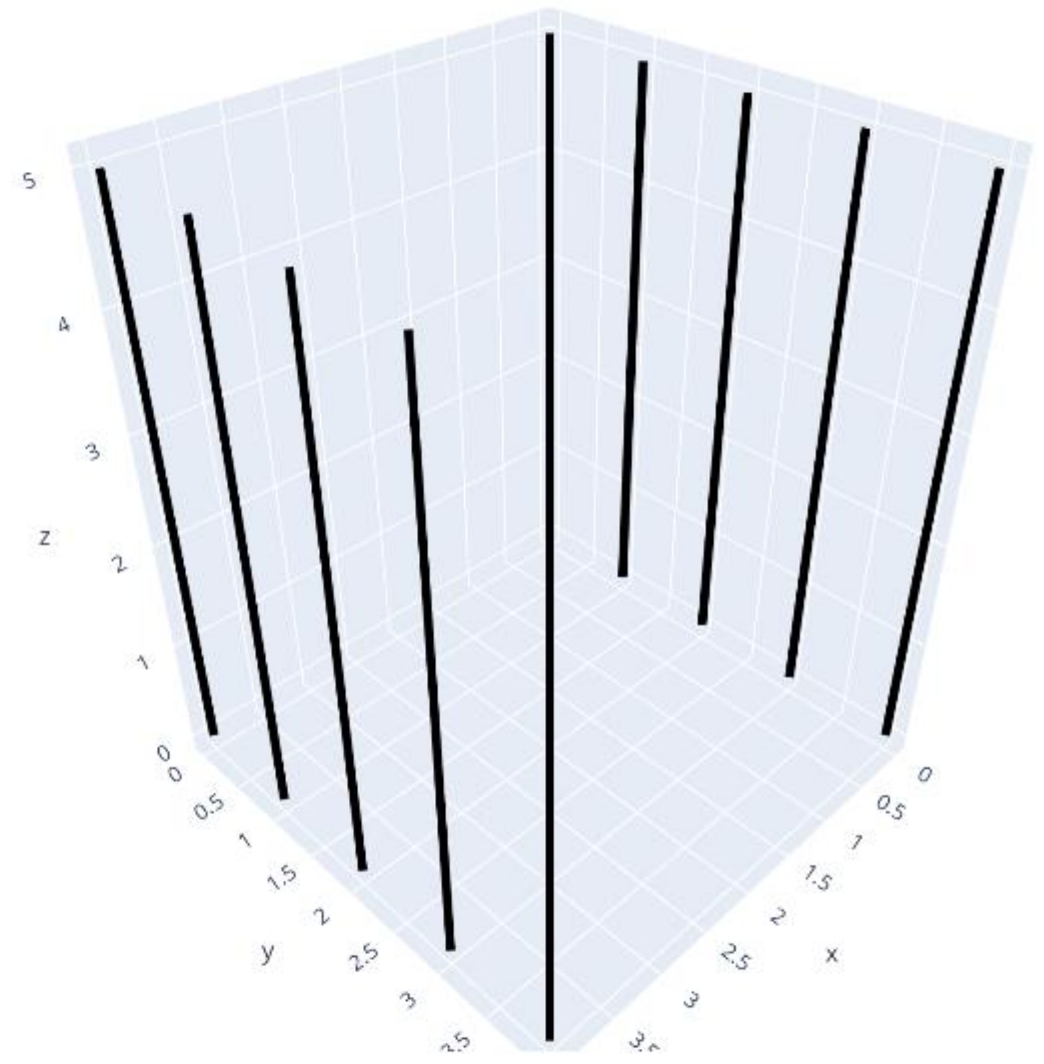
more sparse than checkboard

Active links are shown by **black lines**

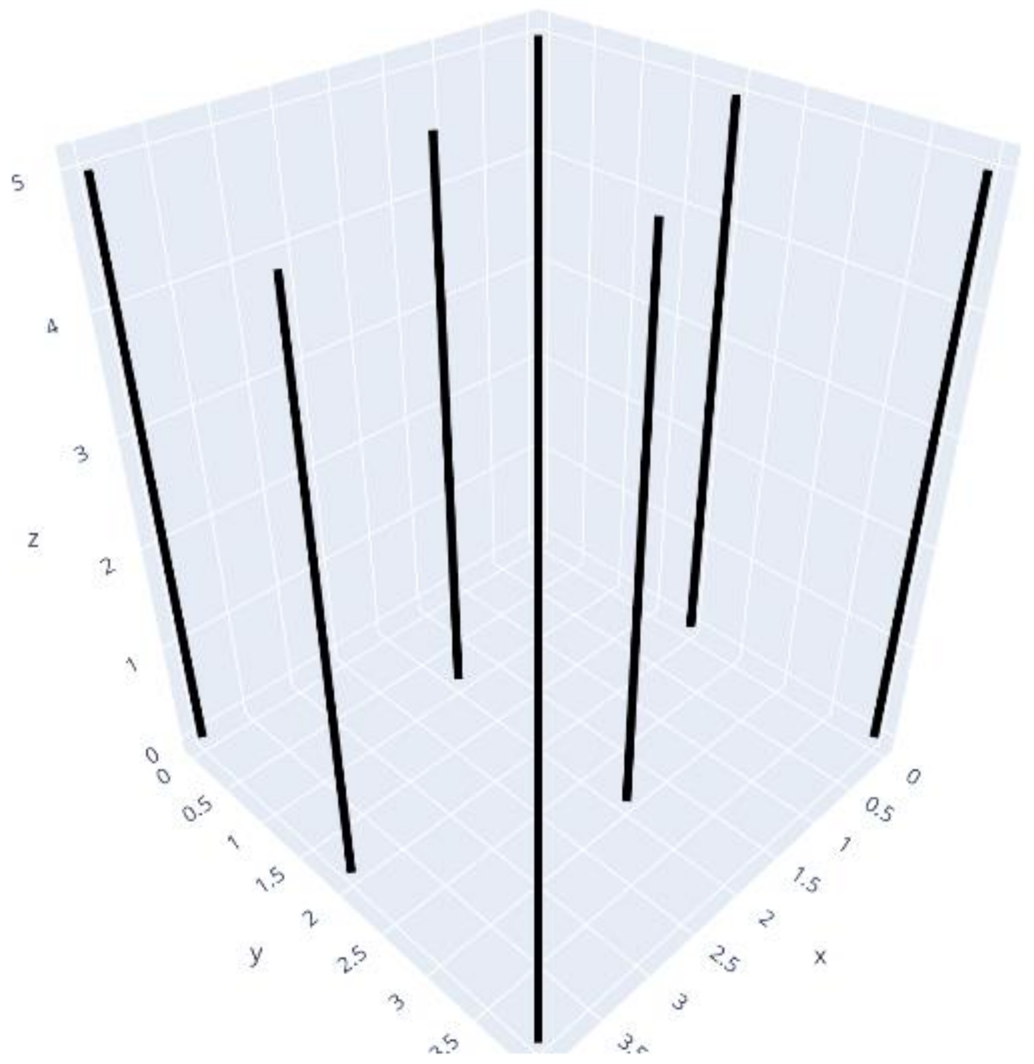
directions= [1, 2], steps = [1, 0, 0]



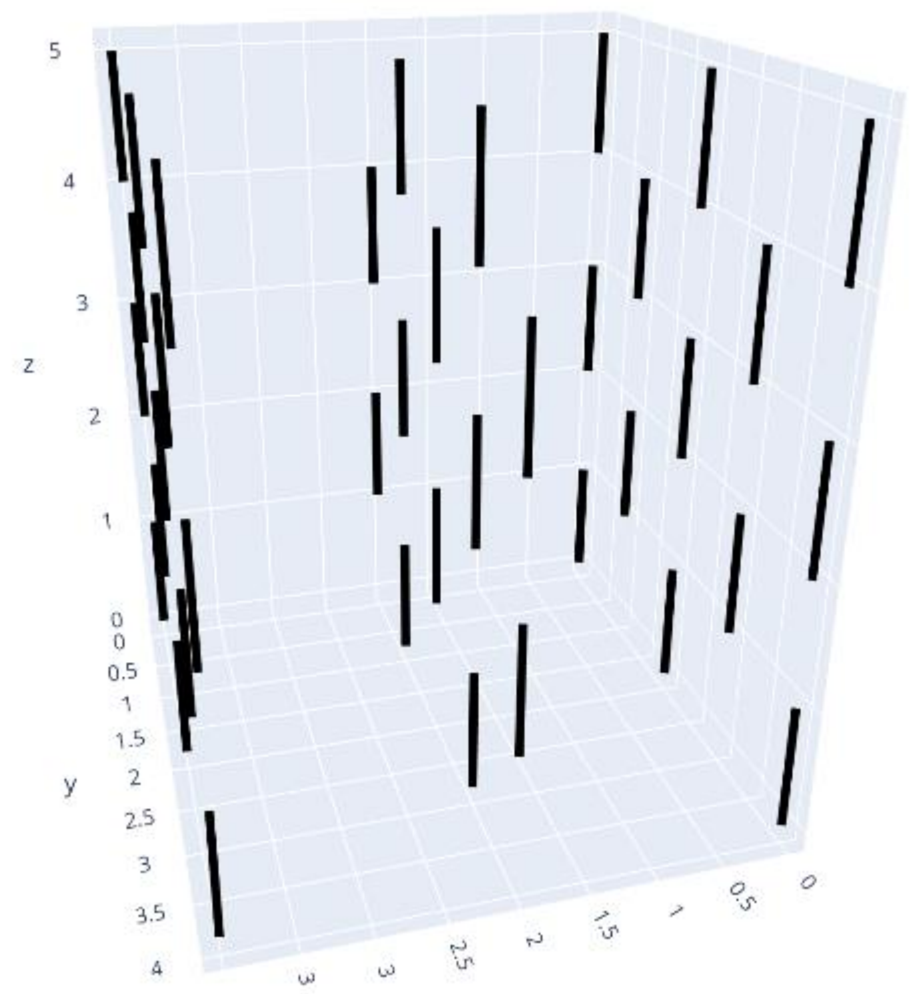
directions= [2], steps = [1, 0, 0]



directions= [2], steps = [1, 2, 0]

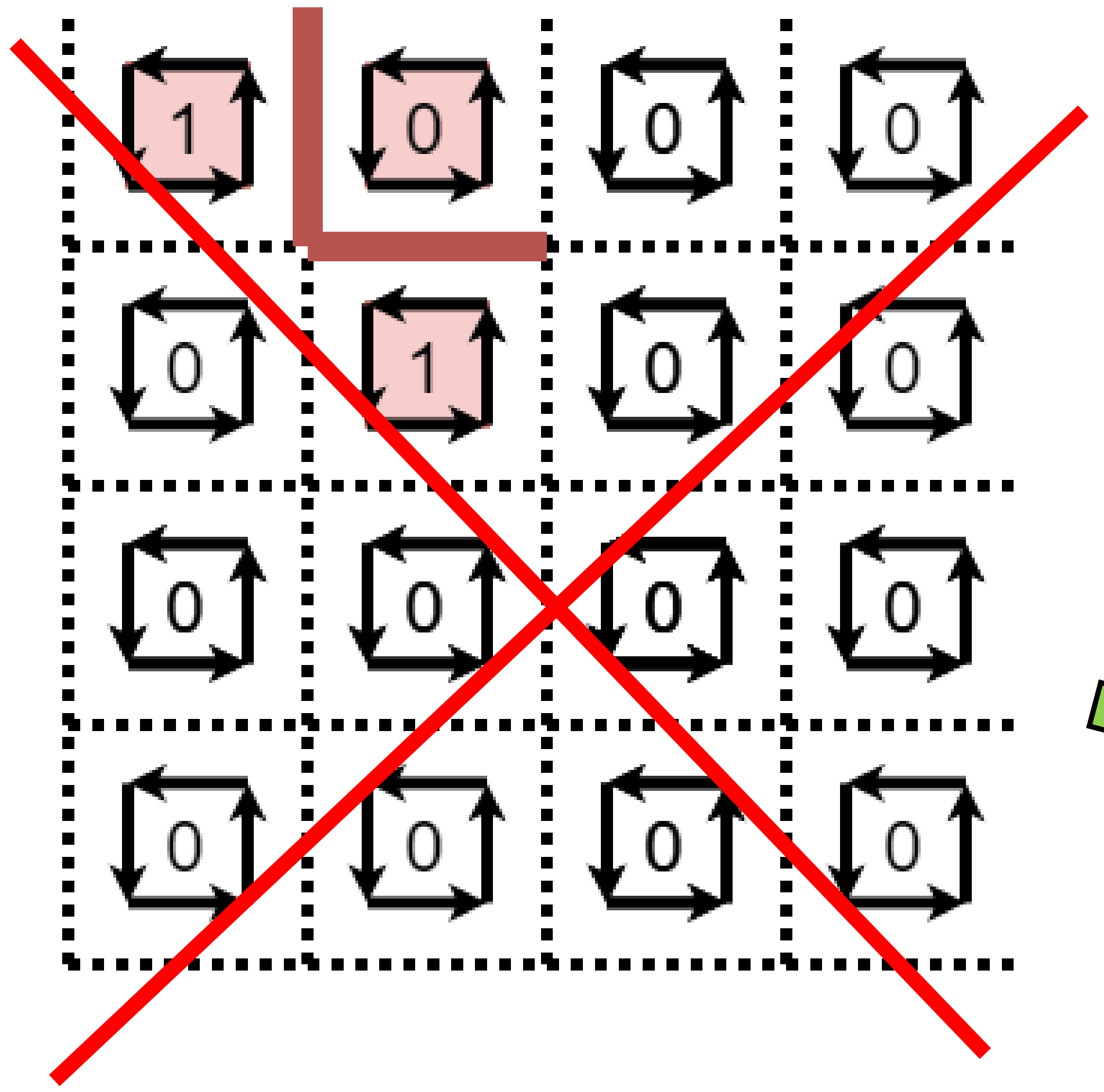


directions= [1], steps = [1, 2, 2]

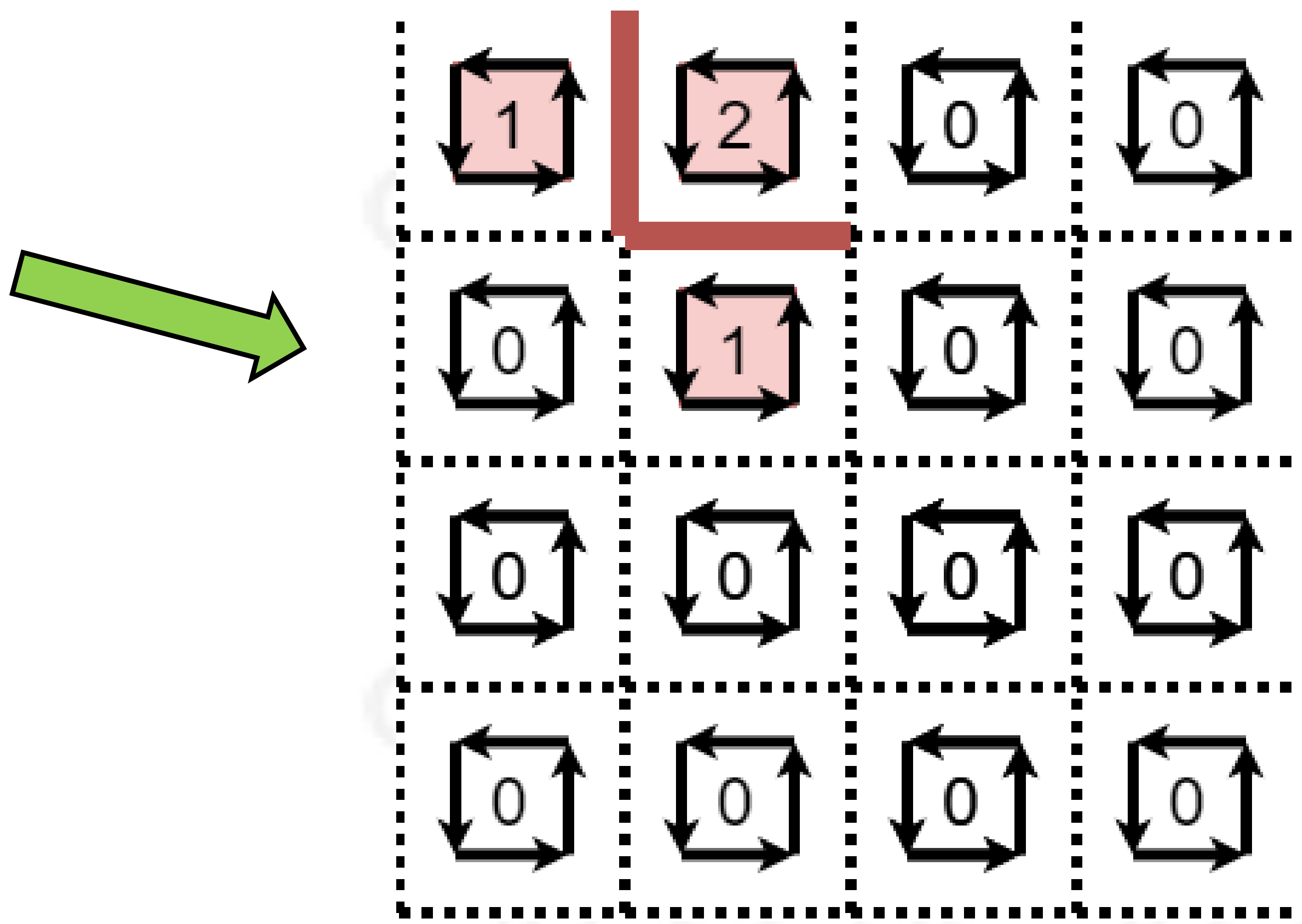


Algorithm for Loop mask

The algorithm must **prevent canceling active links out**



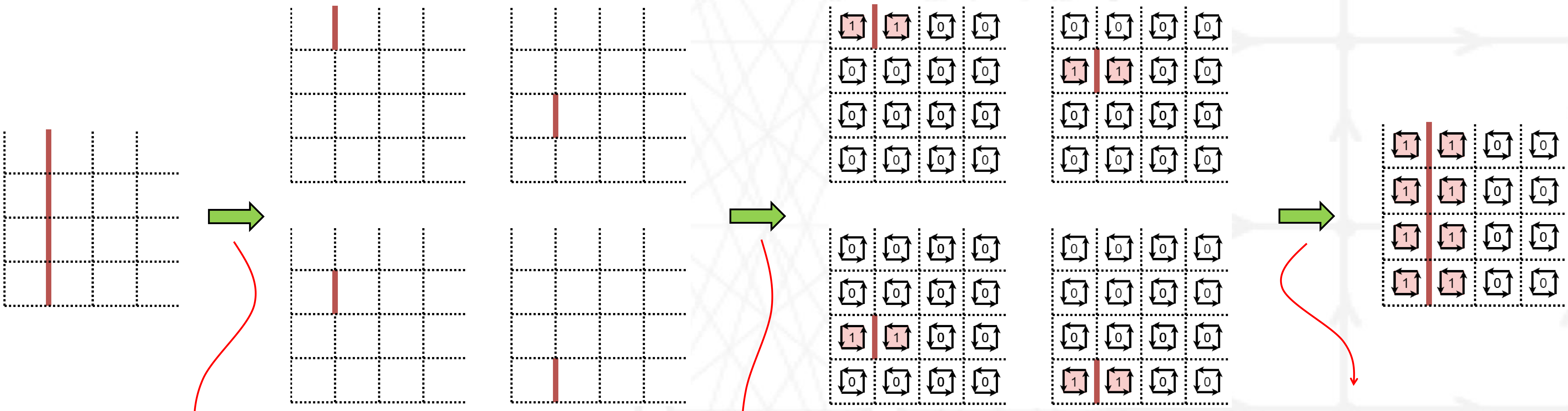
and be **valid for any link mask!**



Idea: any loop containing no active links will be **frozen**.
Use properties of NaNs.

Idea: count how many active links contain every loop

Algorithm for Loop mask



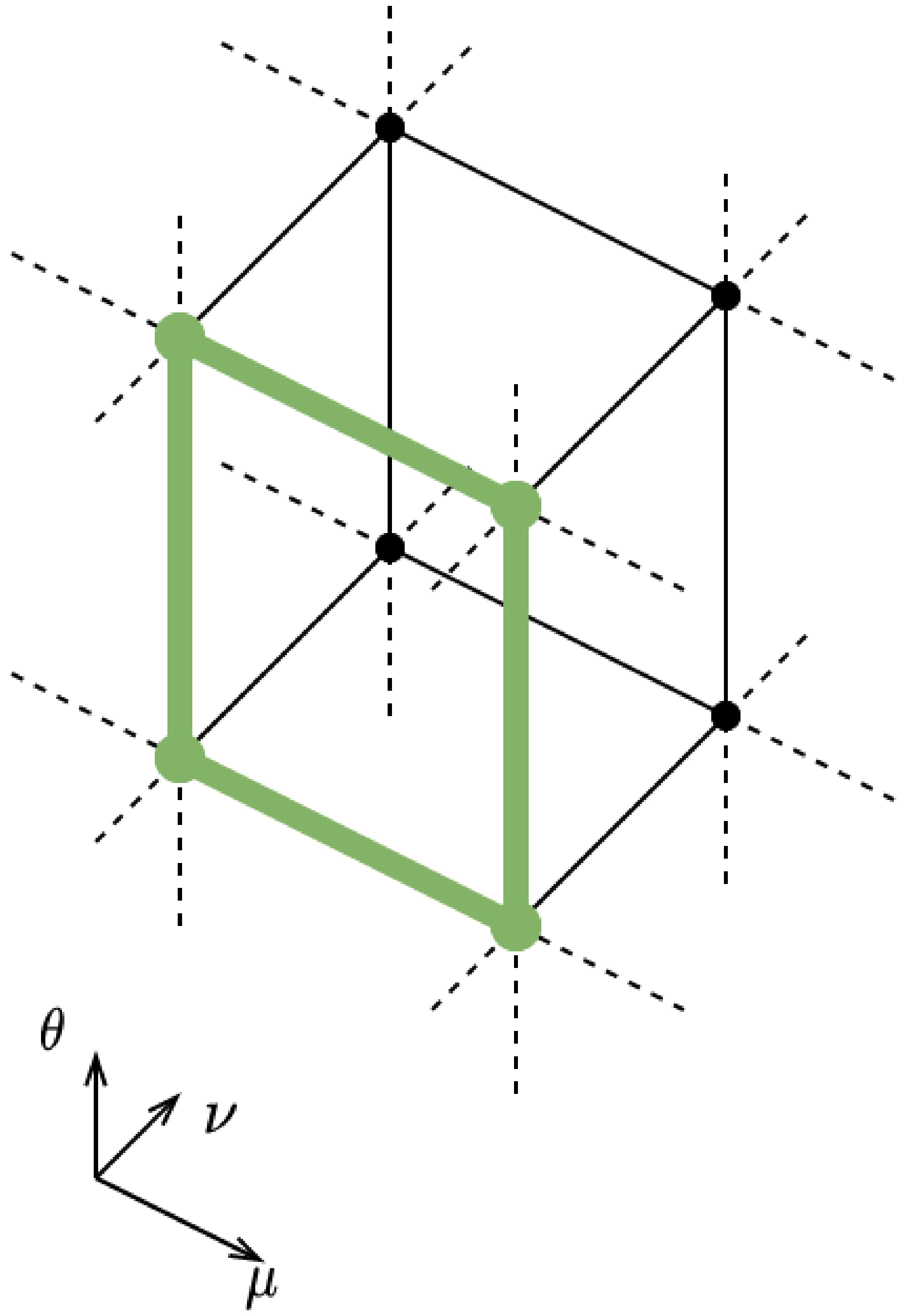
split active link mask to samples such that every sample contains only one active link

compute loops on every sample

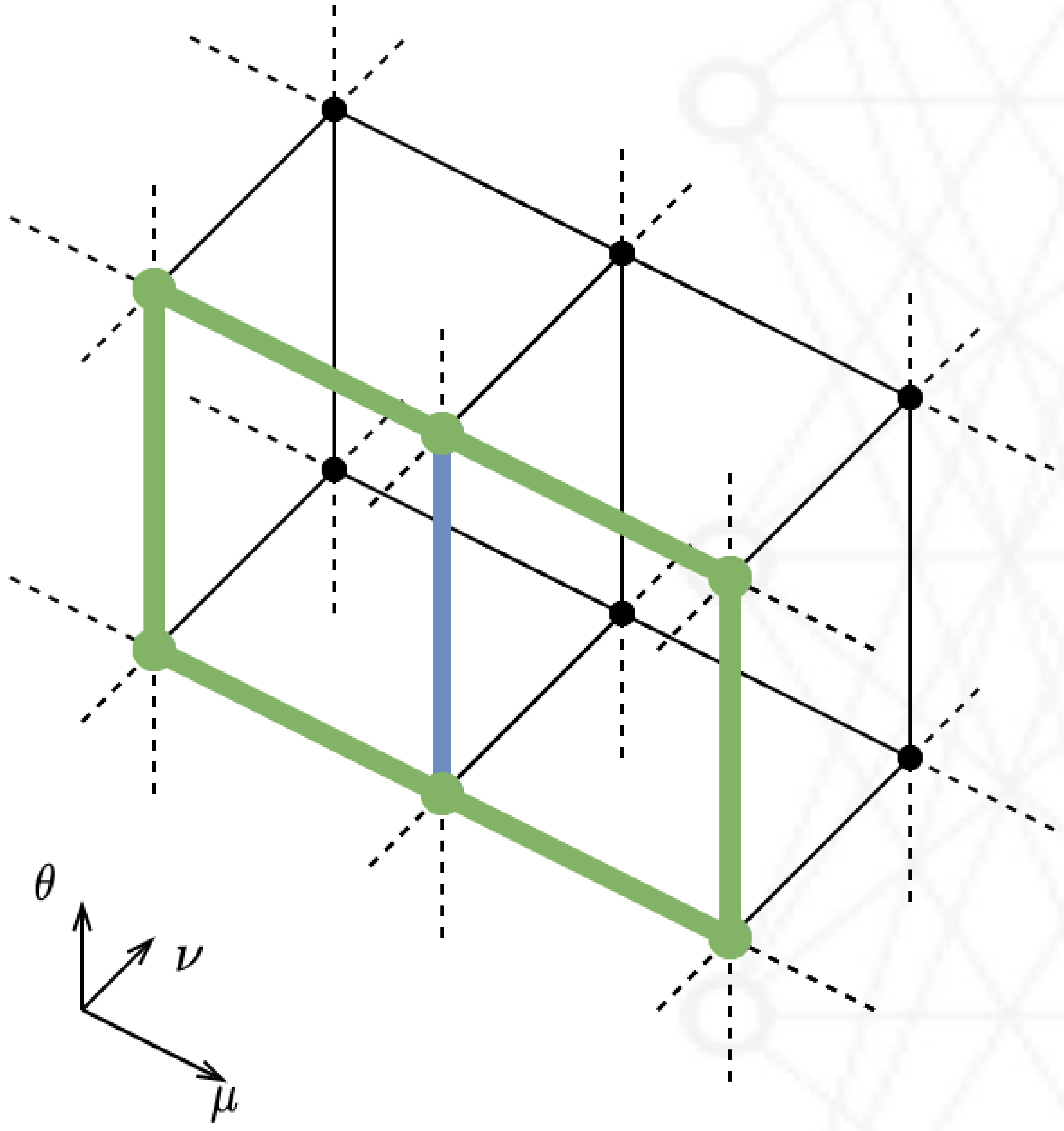
sum samples

Frozen loops

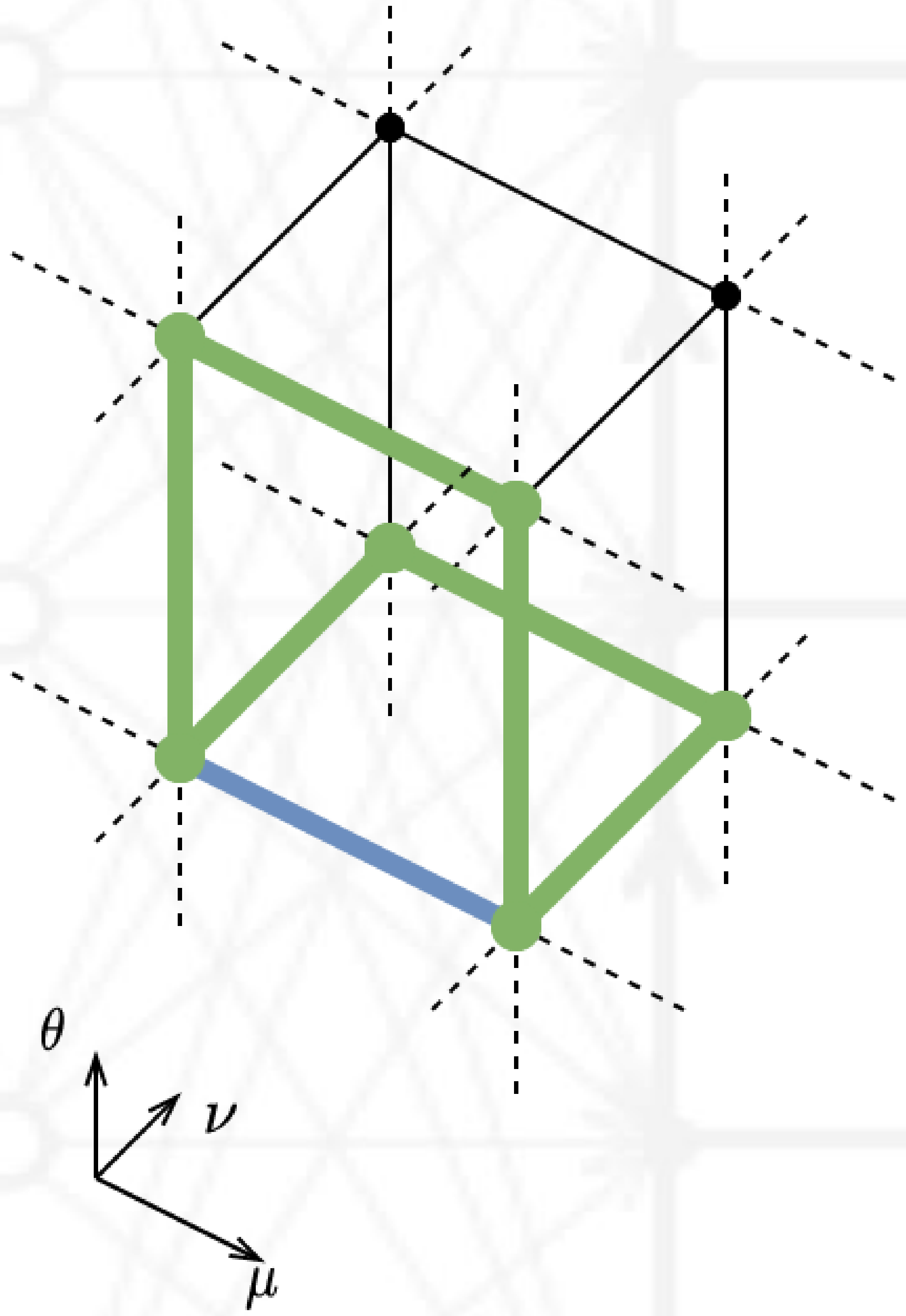
plaquette



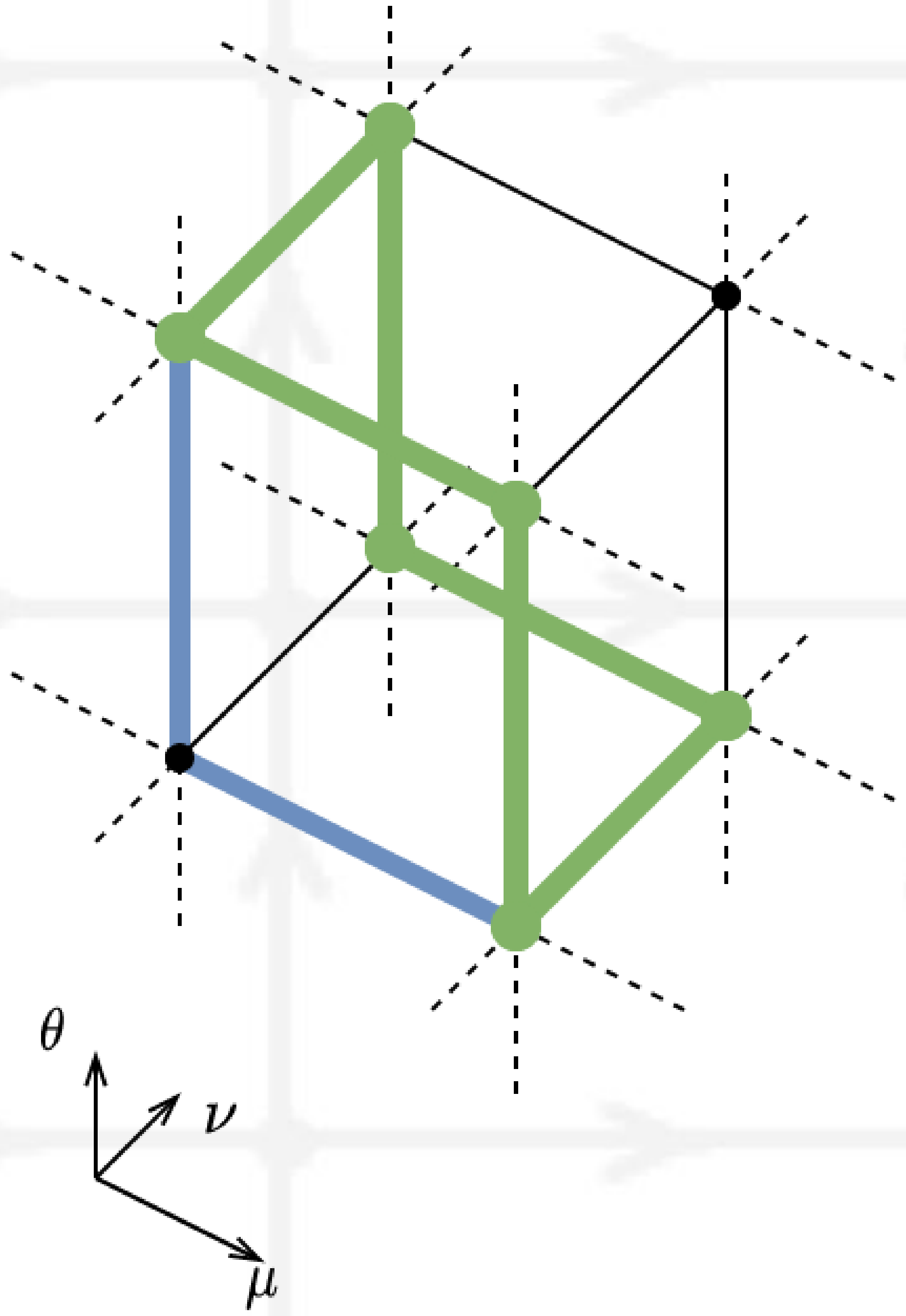
2x1 loop



chair



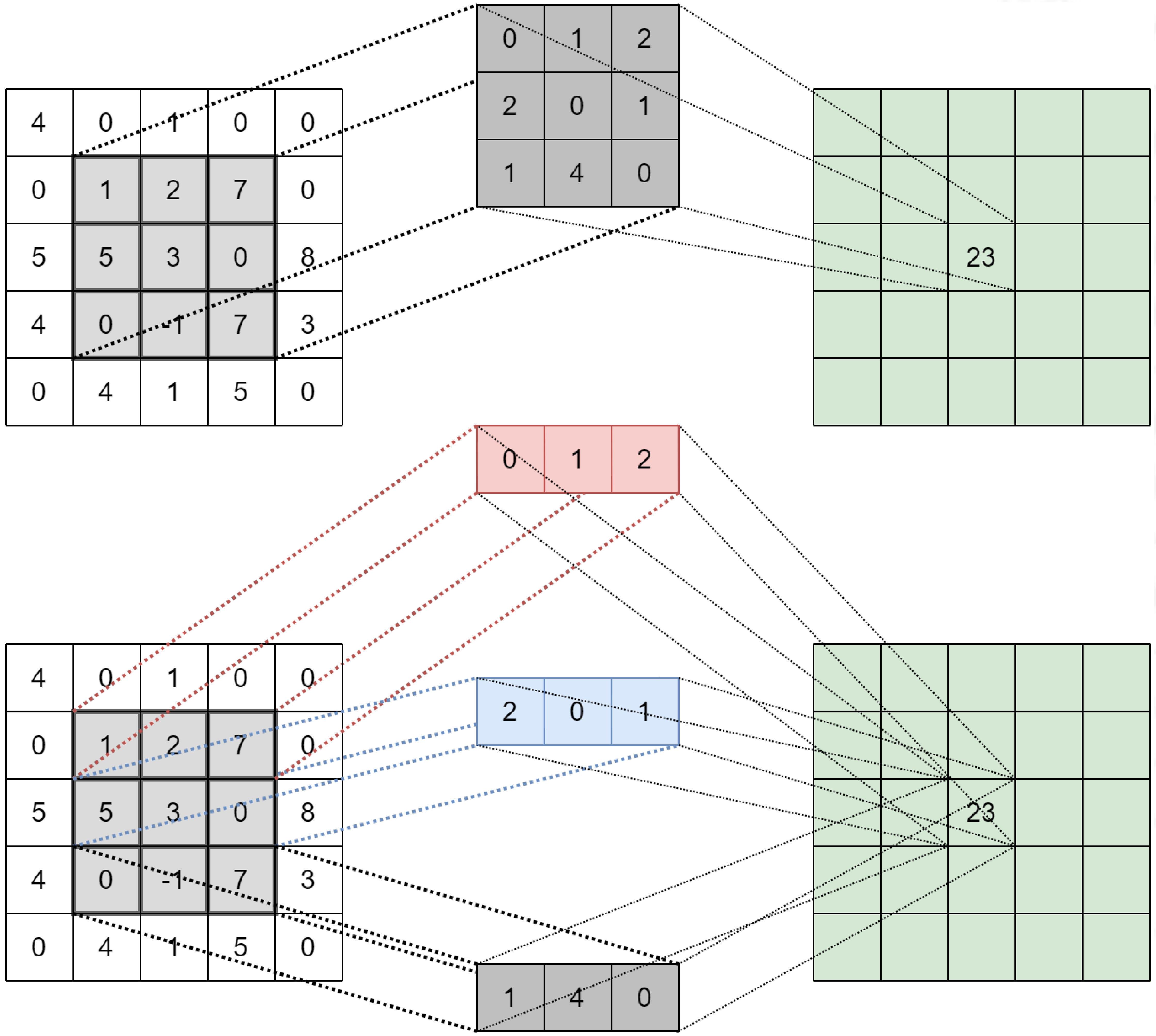
crown



Not all loops can be constructed from plaquettes due to masking!

But, bigger loops can be constructed from plaquettes, 2x1 loops, chairs and crowns!

Convolutional layer in 4D



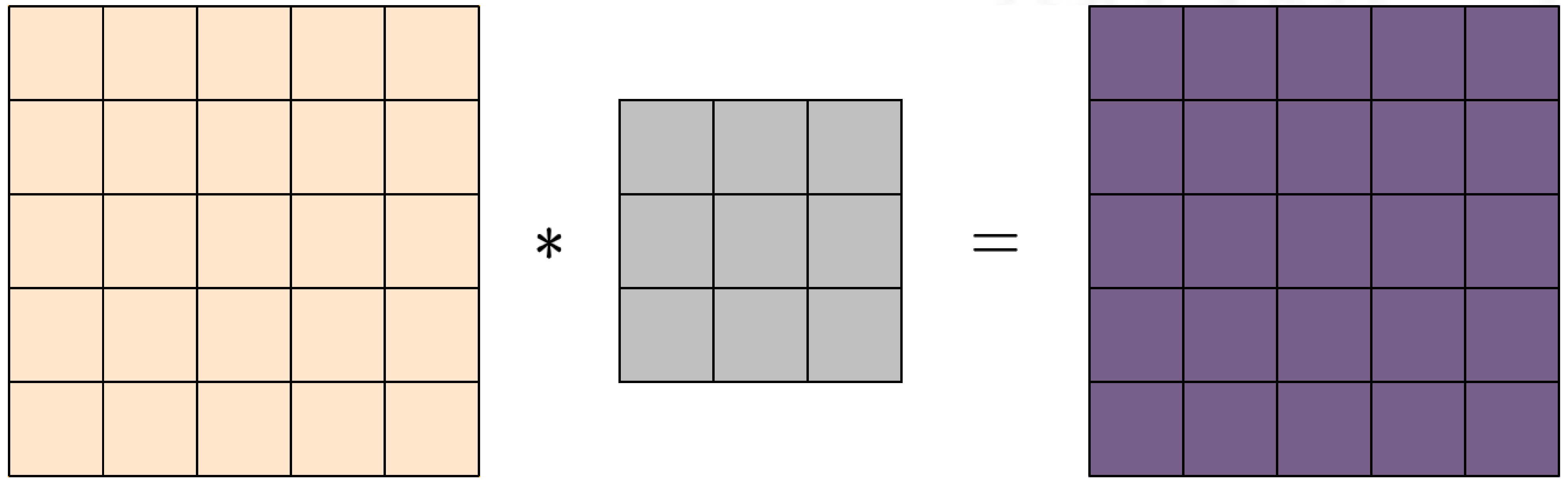
Idea:
Convolution layer in 4D is sum of convolution layers in 3D

Depending on parameters naive implementation in PyTorch shows a slowing down for 20%-50% comparing with PyTorch 3D Convolutions on GPU (Nvidia).
See implementations
- https://github.com/boydad/pytorch_conv4D
- https://github.com/funkey/conv4d
- https://github.com/timothygebhard/pytorch-conv4d
- https://github.com/pvjosue/pytorch_convNd

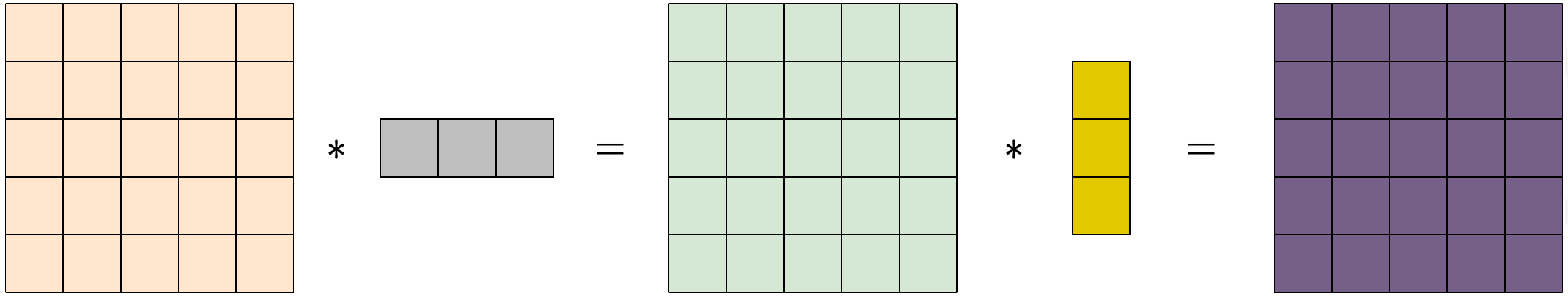
Factorized convolutions

Idea: matrix is outer product of two vectors

$$\begin{pmatrix} 3 & 6 & 9 \\ 4 & 8 & 12 \\ 5 & 10 & 15 \end{pmatrix} = \begin{pmatrix} 3 \\ 4 \\ 5 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 3 \end{pmatrix}$$



Factorized convolutions:
- fewer parameters
- fewer multiplications
- easier to train
- no need for 4D convolution



see CP decomposition and tucker decomposition for details; and review on decompositions https://arxiv.org/pdf/1906.06196

Some other optimizations

Coupling transformation

- Rational quadratic spline with bias

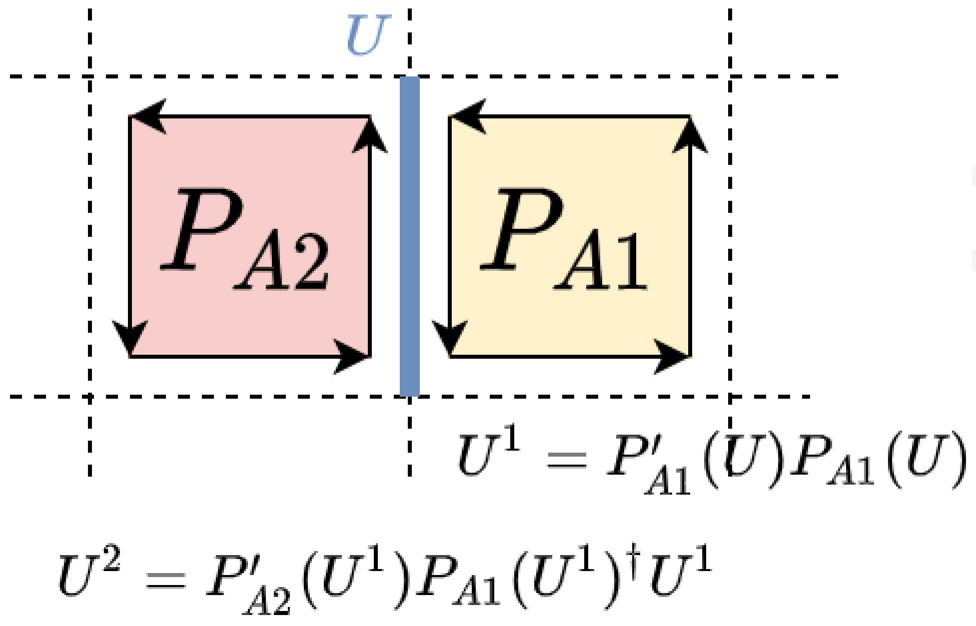
$$y^{active} = RQS\left(x^{active} \middle| NN(x^{frozen})\right) + NN(x^{frozen})$$

Optimizer regularization

- Long training can cause instability due to Adam optimizer (regularize or use amsgrad)

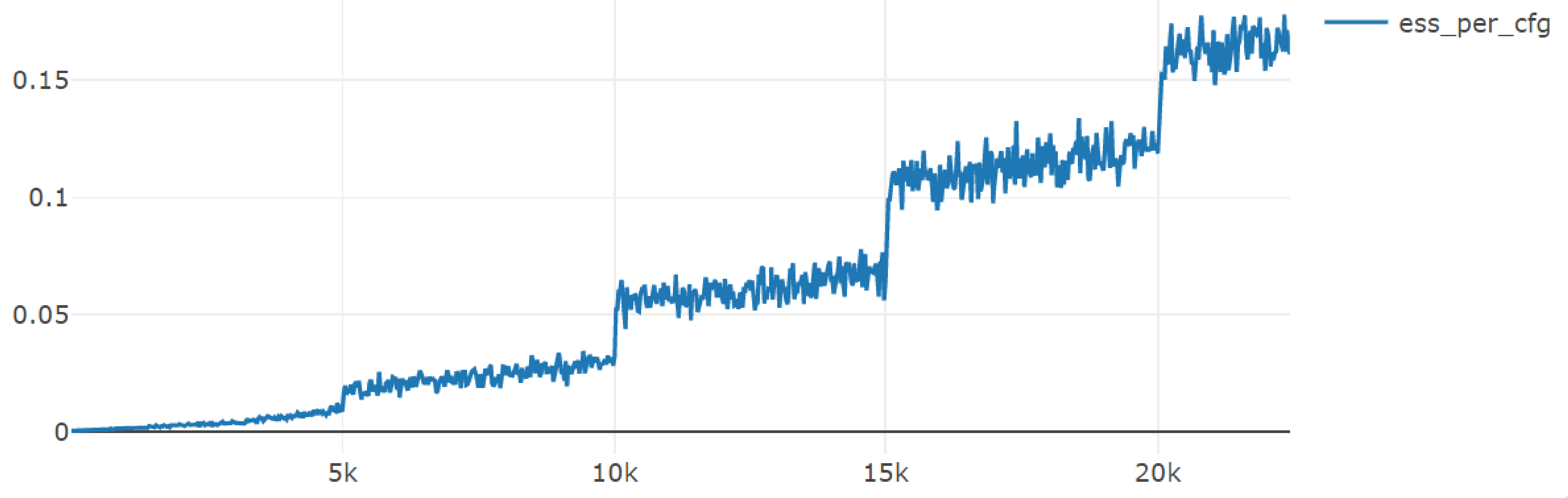
Multi-update

- Update link several times using all active plaquettes



Scheduler policy

- Long training with several decreasing learning rate is efficient



Conclusions

- Development of a flow-based model for lattice gauge theory in 3/4D required a set of optimizations

- All optimizations proposed here may be used for other models

Acknowledgment

This work is associated with an ALCF Aurora Early Science Program project, and used resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DEAC02-06CH11357.