Ensemble generation for lattice QFT using machine learning

Gurtej Kanwar, MIT

JLab Theory Seminar

November 30, 2020

Denis Boyda
Ketandu Chiedu
Dan Hackett
Gurtej Kanwar
Yoshi Saito
Phiala Shanahan

Michael Albergo
Kyle Cranmer

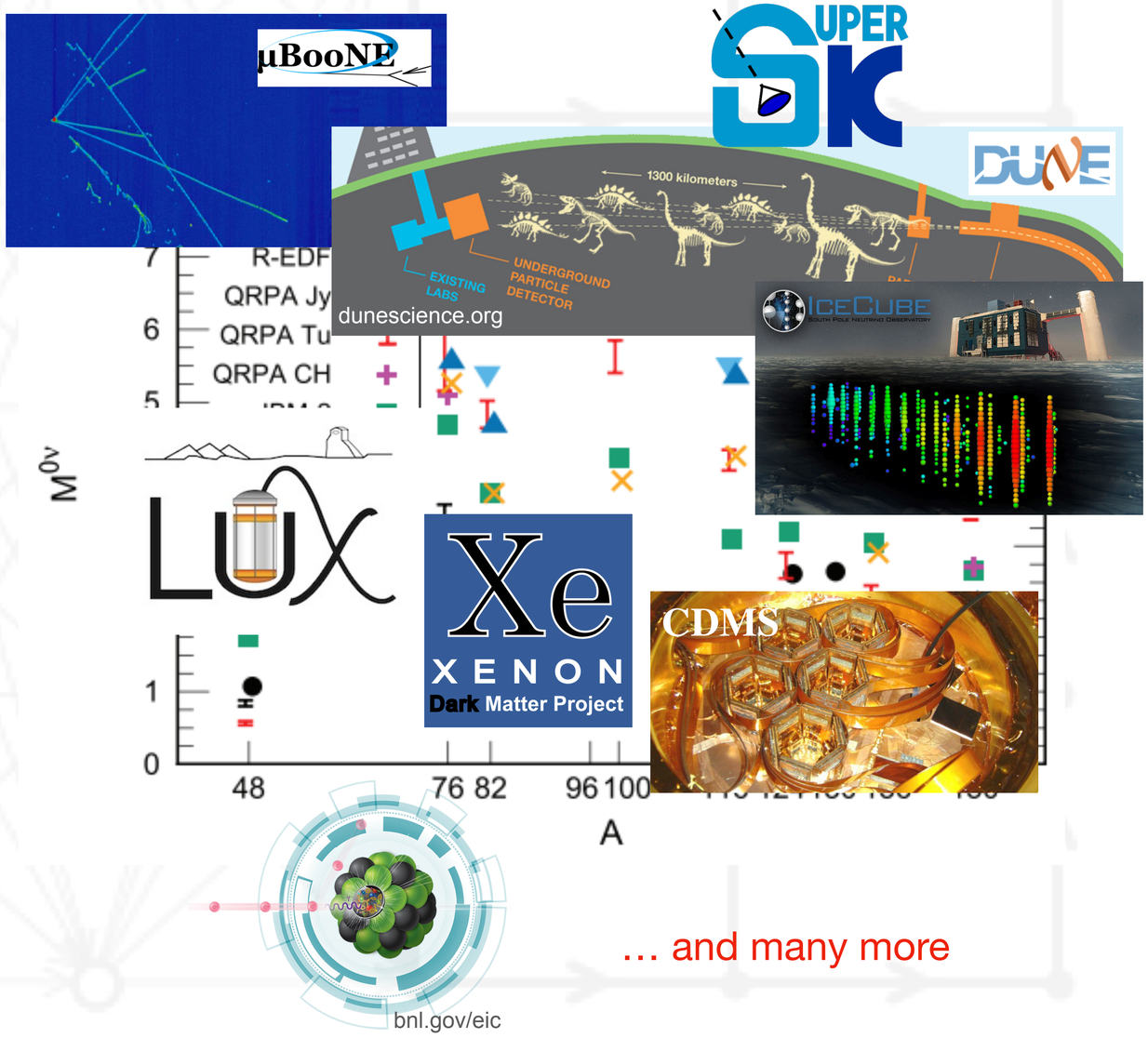
Jiunn-Wei Chen
Javier Hsieh

Alex Matthews
Sébastien Racanière
Danilo Rezende
George Papamakarios
Peter Wirnsberger

Julian Urban

(Just one) Motivation for an ab initio approach

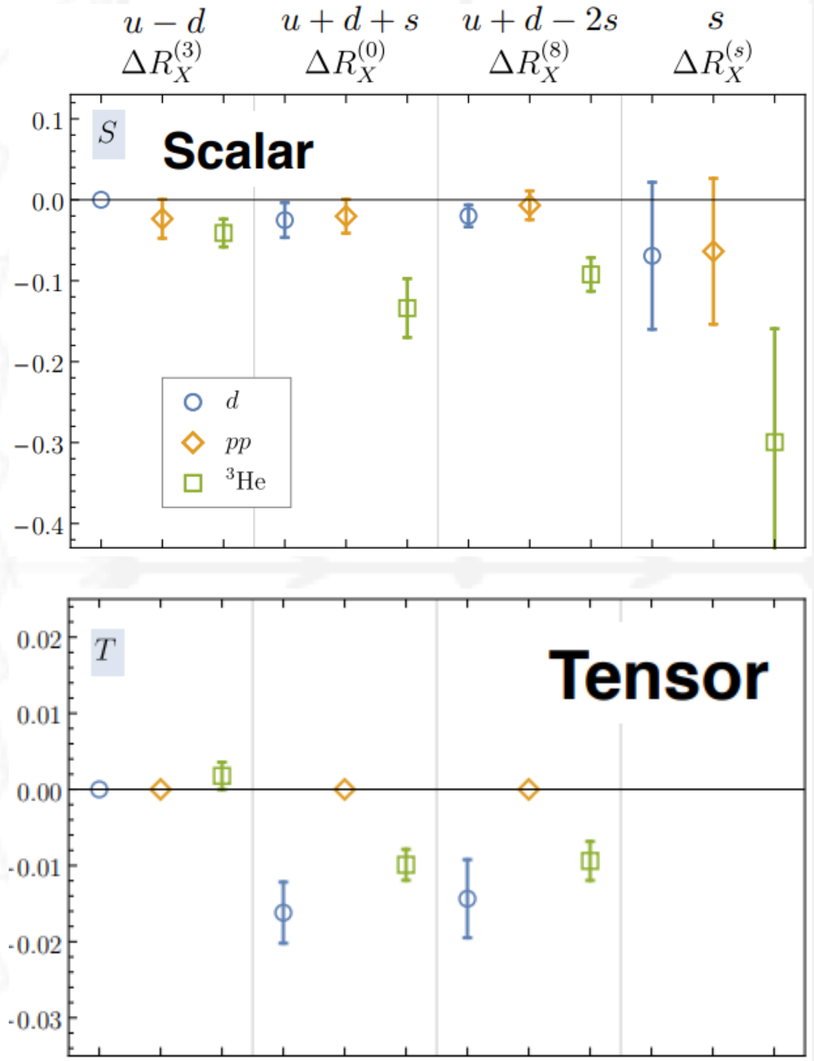
- Many experiments for new physics rely on nuclear targets / samples
- Need to know SM predictions for nuclear matrix elements, structure functions
- Models disagree: **ab initio is key!**



… and many more

Ab initio nuclear physics

- **Lattice QCD** gives theoretical input in non-perturbative regime
- Nuclear matrix elements from theory → LECs for EFT methods
- Complementary to experiment



[Chang et al, PRL120 (2018) 152002]
[Shanahan, INT-18-70]

Outline

- Background:
  - Lattice gauge theories
  - Efficient ensemble generation
- Normalizing flows:
  - "Flow-based" MCMC sampler
  - Imposing symmetries (e.g. gauge, translational, …)
- Applications:
  - Scalar theory and U(1) + SU(N) gauge theory in 2D

[Albergo, **GK**, Shanahan PRD100 (2019) 034515]

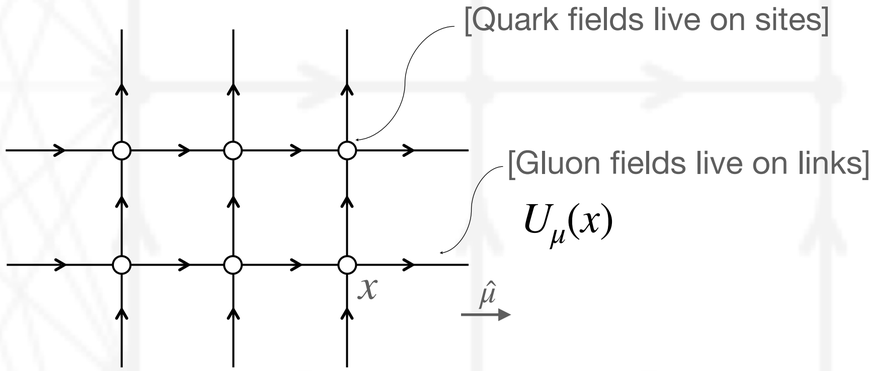
[**GK**, Albergo, Boyda, Cranmer, Hackett, Racanière, Rezende, Shanahan PRL125 (2020) 121601]
[Boyda, **GK**, Racanière, Rezende, Albergo, Cranmer, Hackett, Shanahan 2008.05456, submitted to PRD]

Background

Lattice gauge theory

- Access strongly-coupled limits of gauge theories
  - Low-energy limit of QCD
  - Strongly-coupled composite dark matter [Kribs, Neil 1604.04627]
- Discretized (Euclidean) spacetime
  - Lattice spacing cuts off momenta
  - Exact gauge invariance
- Non-perturbatively compute observables from path integral

$$\langle \mathcal{O} \rangle = \frac{1}{Z} \int \mathcal{D}U \, \mathcal{O}[U] e^{-S[U]}$$



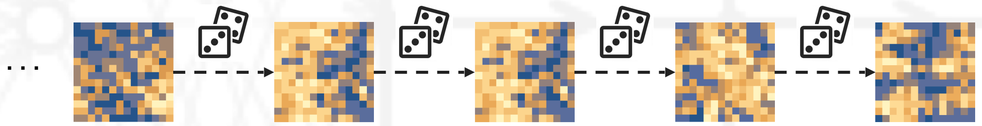
Discretization of QCD on a lattice
[Image credit: Dan Hackett]

Importance sampling

- Sampling histories in path integral enables efficient estimation of (many) observables

$$\langle \mathcal{O} \rangle \approx \frac{1}{n} \sum_{i=1}^{n} \mathcal{O}[U_i] \qquad U_i \sim p(U) = e^{-S(U)}/Z$$

- Markov chain Monte Carlo (MCMC)



Example: MCMC to generate ensembles for scalar field theory

- Performance limitations of local MCMC
  - Information transfer limited by local updates
  - Rare to update entire field coherently

⚠️ **critical slowing down** ⚠️

⚠️ **topological freezing** ⚠️

Better importance sampling?

Ideal world: independently draw samples directly from your distribution

- E.g. sampling Gaussian variables via the Box-Muller transform

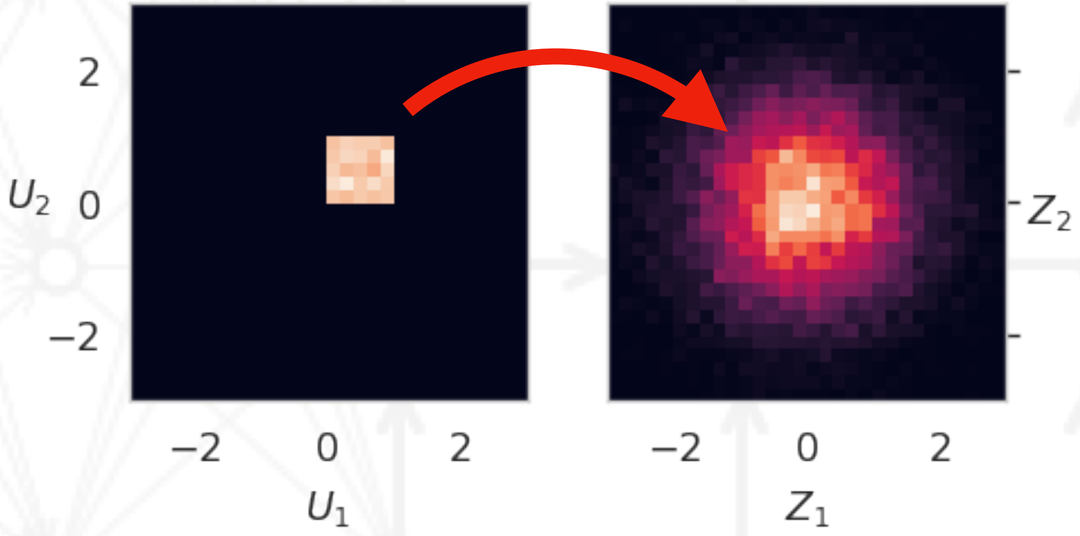
1. Draw samples $U_1, U_2 \in [0,1]^2$ from (uncorrelated) uniform distribution r

2. Change variables

$$Z_1 = \sqrt{-2\log U_1}\cos 2\pi U_2$$
$$Z_2 = \sqrt{-2\log U_1}\sin 2\pi U_2$$

3. Know $r(U_1, U_2) = 1$

$$p(Z_1, Z_2) = r(U_1, U_2)\left|\det_{kl}\frac{\partial Z_k}{\partial U_l}\right|^{-1}$$
$$= \frac{1}{2\pi}e^{-(Z_1^2+Z_2^2)/2}$$



Density is affected by the change of measure!

Better importance sampling?

~~Ideal world~~: independently draw samples directly from your distribution

- Do not know how to exactly sample lattice gauge theory distributions

- We can use an approximate independent sampler (without introducing bias)
  - … if we know the **prob. density** $q(U)$ **being sampled**
  - Reweighting or Independence Metropolis MCMC give unbiased estimates

$$\langle \mathcal{O} \rangle = \frac{\int \mathcal{D}U\, q(U) \left[\mathcal{O}(U) \frac{p(U)}{q(U)} \right]}{\int \mathcal{D}U\, q(U) \left[\frac{p(U)}{q(U)} \right]}$$

Sample $q(U)$ as proposals in a Metropolis-Hasting Markov chain (more later)

Normalizing flows for sampling

1. **Sample** $U \sim q(U)$
2. **Measure** $q(U)$ given U

We already saw a normalizing flow: Box-Muller transform to draw Gaussian vars.

Normalizing flows

ML method to construct samplers for complicated probability distributions

1. Start with an easily-sampled prior distribution r

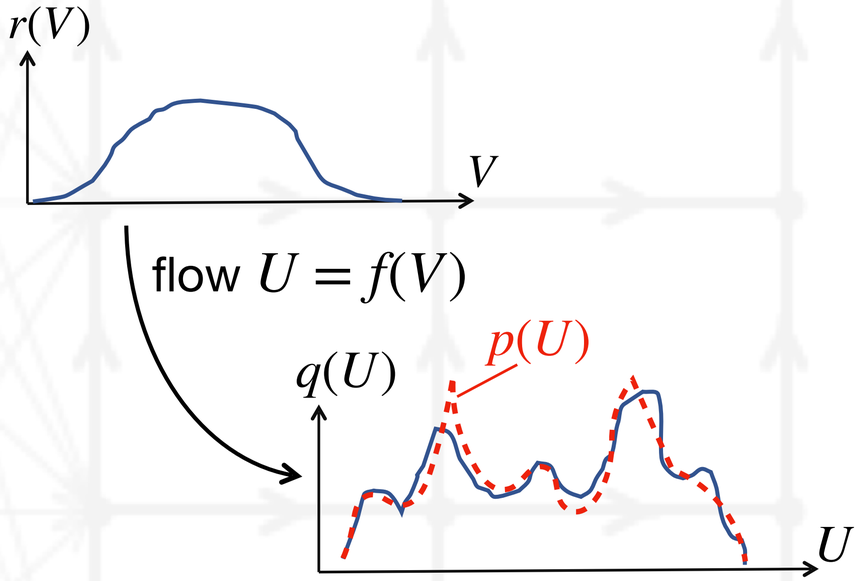
   Ex: uncorrelated uniform, Gaussian, …

2. "Flow" to distribution q using a parametrized change of vars f

   f must be invertible, calculable log-det-Jacobian

Approach: Construct flow as a variational ansatz for p, optimize so that $q \approx p$

Faces generated via "real NVP" flow from uncorrelated noise
[Dinh, Sohl-Dickstein, Bengio 1605.08803]



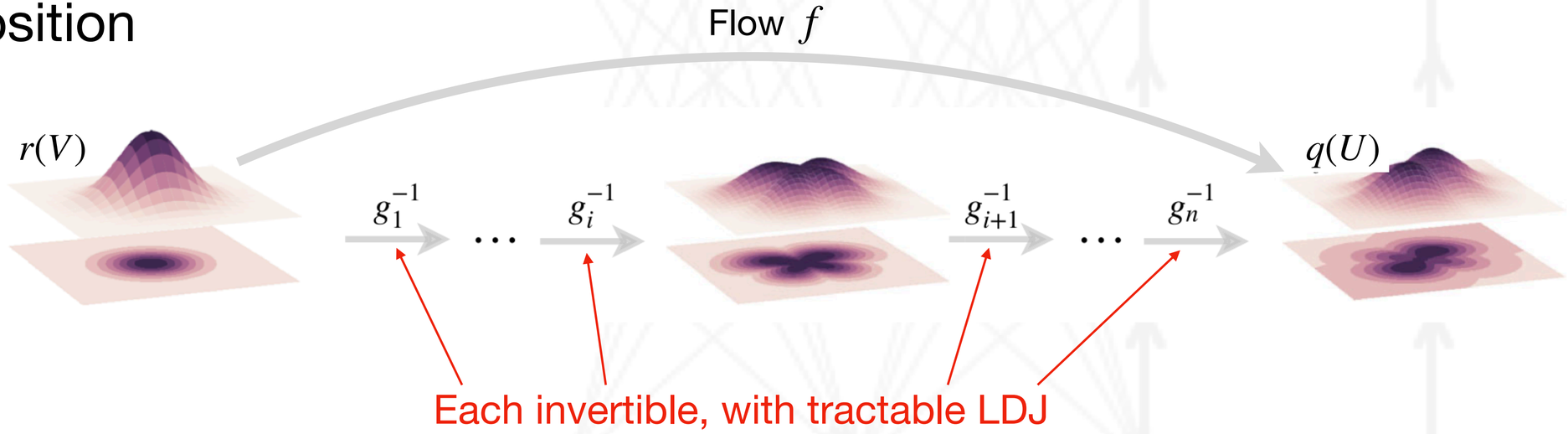
$$q(U) = r(V) \left| \det_{ij} \frac{\partial [f(V)]_i}{\partial V_j} \right|^{-1}$$

[Slide credit: Dan Hackett]

Defining the flow function

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

- The "flow" f must be **invertible** and have **tractable log-det-Jacobian (LDJ)**
  - In Box-Muller transform, f is precisely constructed to produce the Gaussian dist
  - For LQFT, don't know what f needs to be; instead, construct parametrized ansatz and optimize it
- Composition



Coupling layers

Idea: Construct each g to act on a **subset** of components, conditioned only on the complimentary subset.

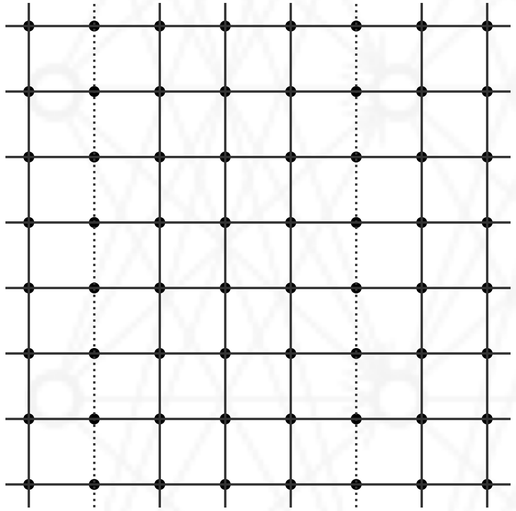
→ Jacobian is explicitly upper-triangular (get LDJ from diag elts)

$$\frac{\partial [g(V)]_i}{\partial V_j} = \left(\begin{array}{cccc|cccc} \frac{\partial [g(V)]_1}{\partial V_1} & & & & & & & \\ & \frac{\partial [g(V)]_2}{\partial V_2} & & & & \text{(nonzero)} & & \\ & & \ddots & & & & & \\ \hline & & & & 1 & & & \\ & 0 & & & & 1 & & \\ & & & & & & \ddots & \end{array}\right)$$

→ Invertible if each diag component invertible, $\dfrac{\partial [g(V)]_i}{\partial V_i} \neq 0.$

Ex: coupling layer for gauge theory

- **Masking pattern:** define which links to freeze and condition on
  - Idea: leave enough frozen context so transform can build correlations between DOFs
  - E.g. freeze all but specific columns (or rows) of links



- Ex: left-multiplication by group-valued function on unfrozen links

$$U'_\mu(x) = \boxed{\exp\left(iW_\xi(\text{frozen neighbors}) \cdot \lambda\right)} U_\mu(x)$$

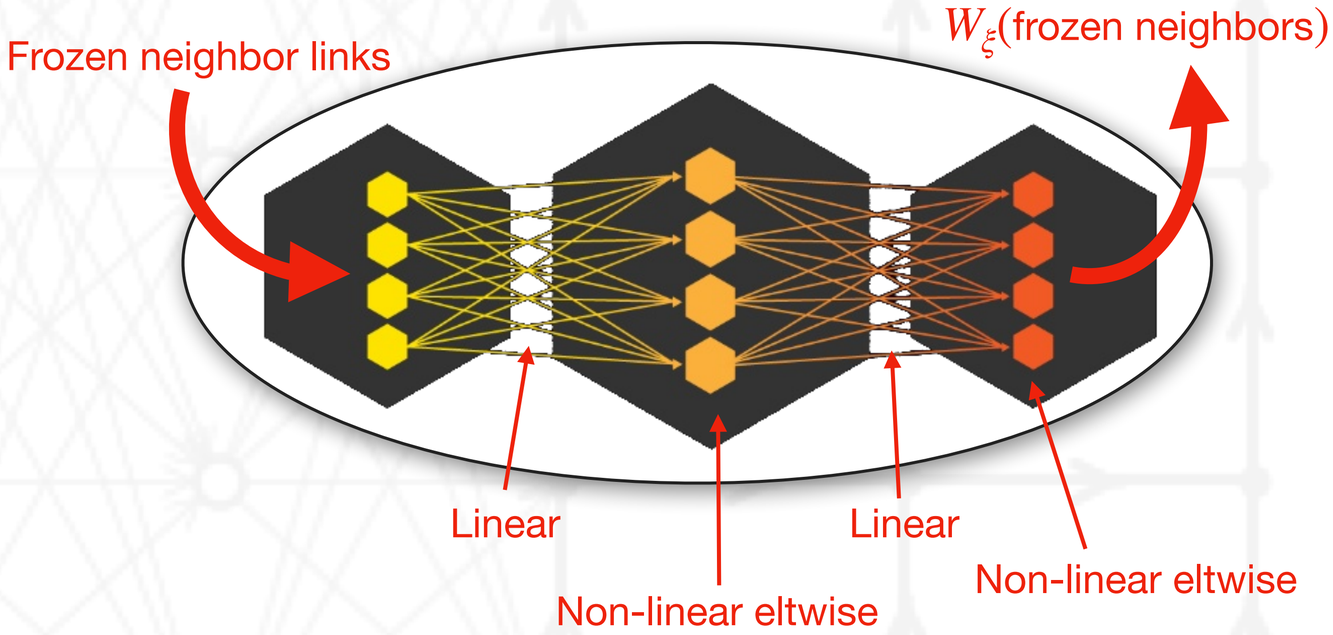
Context Functions and NNs | "What is W_ξ?"

Any parametrized function accepting a collection of $SU(N)$ variables as input.

Our terminology: "*context function*".

Neural networks: compose parametrized linear transforms with non-linear elementwise functions.

→ Universal function approximators



Matrices of weights define linear transforms. Altogether, these weights compose the model parameters ξ.

Optimizing via “self-training”

Optimization by comparing model likelihood $q(U)$ vs true likelihood $p(U)$ on sample configurations U.

- Must not require a large number of samples from real distribution to optimize!
- **Self-training:** take samples from the model, not true distribution
- Kullback-Leibler (KL) divergence between q and p given samples

Flow-based MCMC

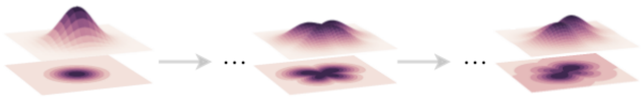
Markov chain constructed using Independence Metropolis accept/reject on model proposals.

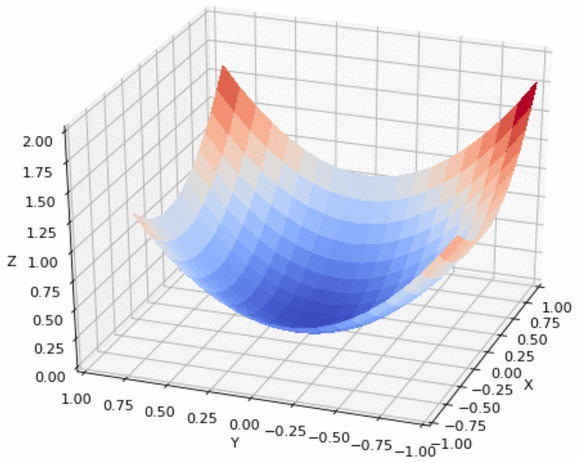
- **Independent** proposals U' from model distribution q
- **Accept** proposal U', making it next elt of Markov chain, with probability

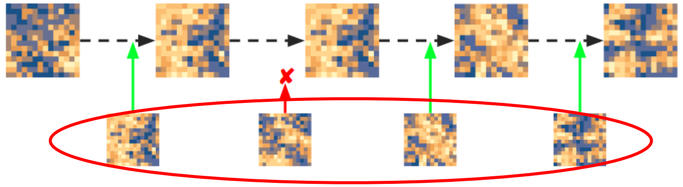
$$p_{\text{acc}}(U \to U') = \min\left(1, \frac{p(U')}{q(U')}\frac{q(U)}{p(U)}\right).$$

- If **rejected**, duplicate previous elt of Markov chain
  - Only need to compute observables on duplicated elts once!

Birds-eye view







generating samples is
"embarrassingly parallel"

Parameterize flow using coupling layers

Each layer contains arbitrary neural nets

Training step

Draw samples from model

Compute loss function

Gradient descent

Desired accuracy?

Save trained model

Markov chain using samples from model

Symmetries

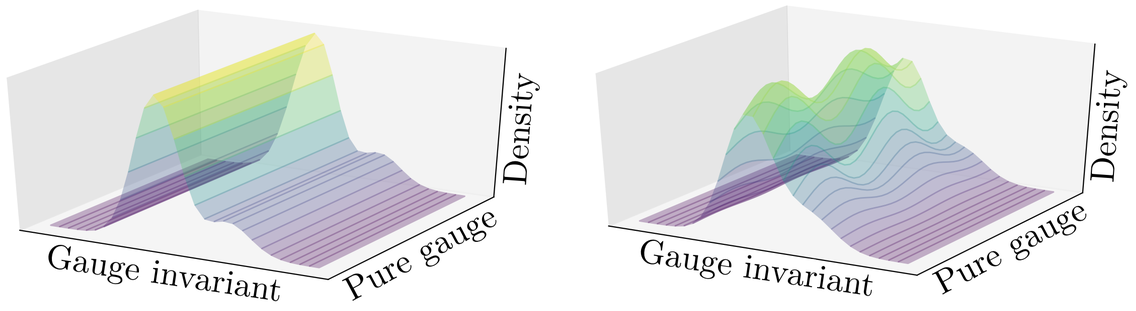
Typical lattice gauge theories are symmetric under

1. (Discrete) translational symmetry
2. Hypercubic symmetry
3. Gauge symmetry

   Elements defined by group-valued fields $\Omega(x)$ that transform the gauge field as

$$(\Omega \cdot U)_\mu(x) = \Omega(x) U_\mu(x) \Omega^\dagger(x + \hat{\mu})$$

Symmetries **factor** distribution into uniform component along symmetry direction, and non-uniform component along invariant direction. Ex for gauge symmetry (schematically):



Learning symmetries

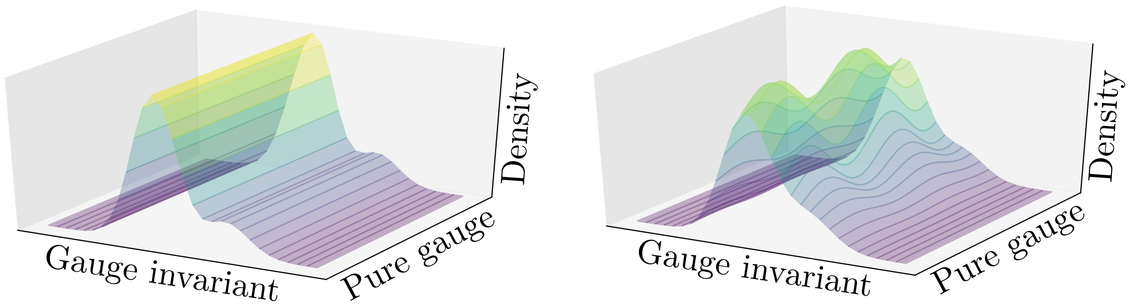
Models will learn any symmetries of the action **approximately**.

- Always made exact after reweighting / flow-based MCMC

Symmetries **factor** distribution into uniform component along symmetry direction, and non-uniform component along invariant direction. Ex for gauge symmetry (schematically):

Some symmetry groups quite large. We can do better by **encoding them explicitly** in model structure!

- Variational ansatz is restricted to only explore distributions like the left one



Symmetries in normalizing flows

Can be imposed on the model by ensuring

1. Prior is **invariant** under the symmetry

Uniform prior distribution is invariant under all symmetries of interest in lattice gauge theory.

2. Flow f is **equivariant** under the symmetry

[Cohen, Welling 1602.07576]

Must construct flow to satisfy equivariance!

Equivariance: symmetry operations commute with application of f

- Translational equivariance achieved by using (1) Convolutional Neural Networks in context functions and (2) symmetric masking patterns.
- Our recent contribution: gauge equivariance

Applications: 1+1D lattice theories

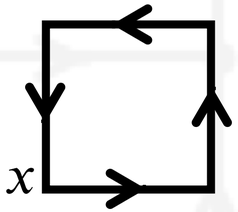
Proof-of-principle at low computational cost,
no theoretical obstacle to higher dims

Lattice gauge theory in 2D

- Wilson gauge action, $SU(N)$:

$$S(U) = -\frac{\beta}{N}\sum_x \mathrm{Re\,tr}\left[P_{01}(x)\right]$$

$$P_{\mu\nu}(x) = U_\mu(x)U_\nu(x+\hat{\mu})U_\mu^\dagger(x+\hat{\nu})U_\nu^\dagger(x)$$



untraced!

- Gauge transforms are a symmetry of the action:

$$(\Omega\cdot U)_\mu(x) = \Omega(x)U_\mu(x)\Omega^\dagger(x+\hat{\mu})$$

- Confinement, ultralocal dynamics

$$p(P_{01}(x)) \approx \exp\left(\frac{\beta}{N}\mathrm{Re\,tr}P_{01}(x)\right)/Z$$

  - Each plaq has independent statistics, up to correlations that are a finite volume effect

Gauge equivariance

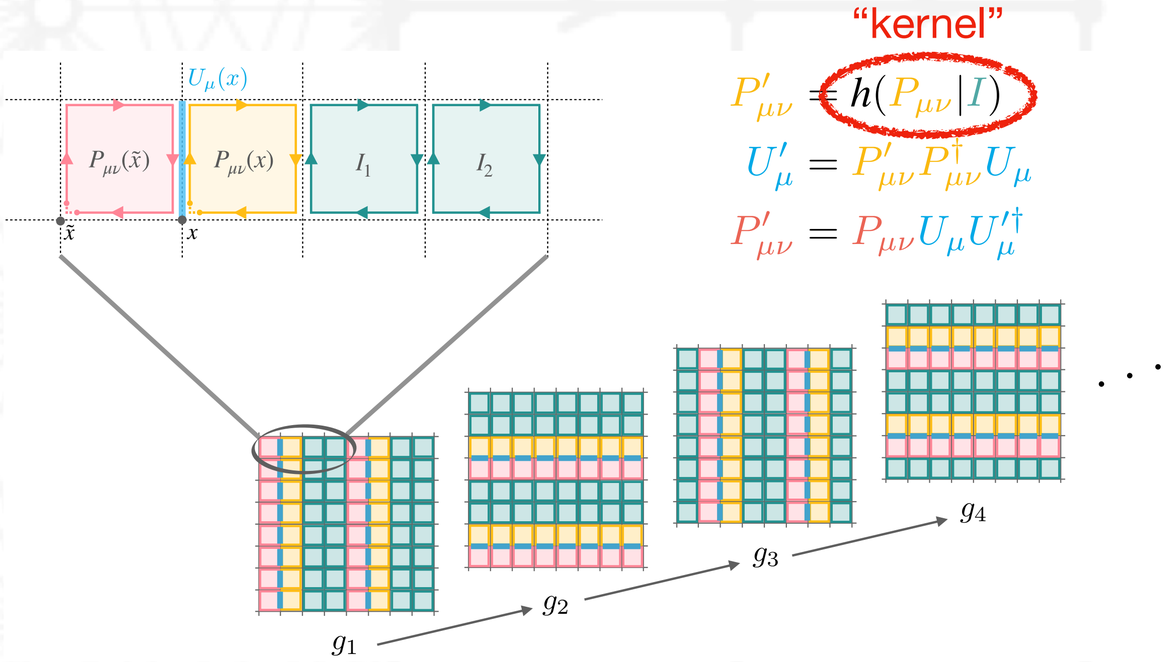
E.g. traced plaquettes, traced Wilson loops

Intuition: act on gauge-invariant quantities only

- Factorized action of coupling layer
- Issue: must remain invertible
- Issue: preserve trans. symmetry

× Gauge fixing with **explicit factorization** (e.g. maximal tree) does not preserve trans. symmetry!

× Gauge fixing with **implicit factorization** (e.g. Landau gauge) hard to preserve across coupling layer!



"kernel"

$$P'_{\mu\nu} = h(P_{\mu\nu}|I)$$
$$U'_\mu = P'_{\mu\nu} P^\dagger_{\mu\nu} U_\mu$$
$$P'_{\mu\nu} = P_{\mu\nu} U_\mu U'^\dagger_\mu$$

Solution: transform group-valued untraced Wilson loops, "absorb" update using link representation

“What is a kernel?”

The core of a gauge-equivariant coupling layer; acts on a selection of untraced Wilson loops (e.g. subset of untraced plaquettes)

Must satisfy:

1. Invertible and tractable log-det-Jacobian
2. Equivariant under matrix conjugation

$$h(XPX^{-1}) = X\,h(P)\,X^{-1}$$

3. Only conditioned on gauge-invariant frozen quantities

Guarantees **gauge equiv**: gauge transforms act on untraced loops via matrix conjugation

“Condition on”: pass as input to context functions used in defining h

[Image credit: Dan Hackett]

Kernel for $U(1)$ gauge theory

- Variables are 1×1 matrices (i.e. scalars):

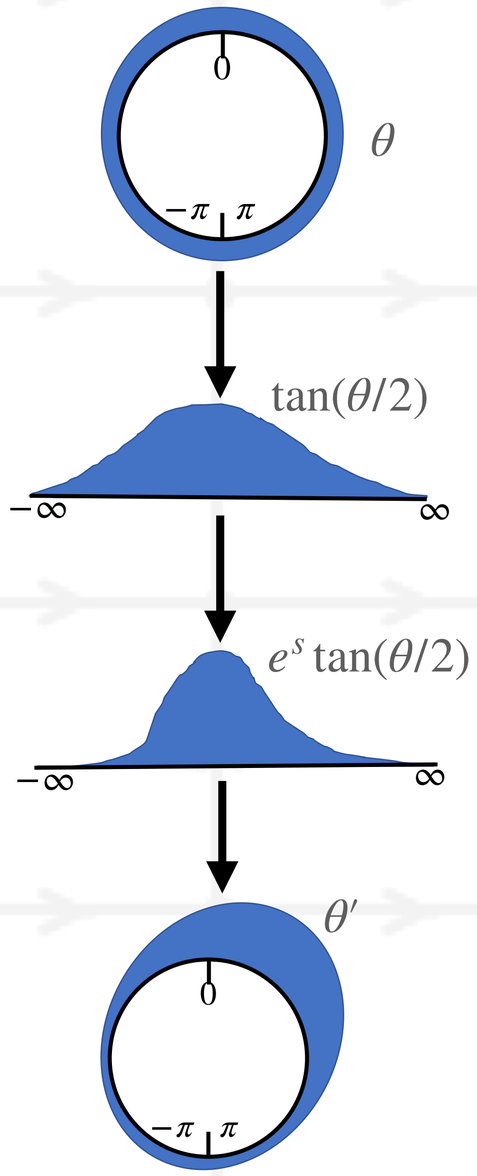
$$X P X^{-1} = P$$

- Required: invertible function suitable for $U(1)$ vars
  - Developed flows for compact variables (tori and spheres) in [†]
  - We choose a "non-compact projection" transform:

s, t = convolutional NNs

$$\theta' = 2 \arctan\left(e^{s_\xi(I)} \tan(\theta/2)\right) + t_\xi(I)$$

ξ = model parameters $\quad$ I = nearby frozen plaquettes



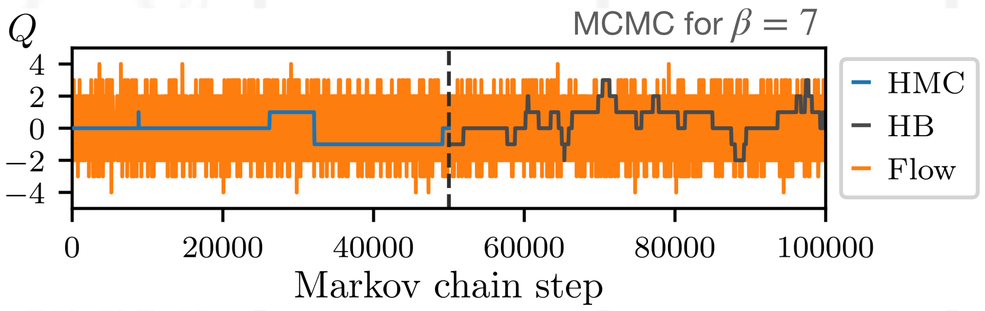
"Non-compact projection" for $U(1)$

[†] [Rezende, Papamakarios, Racanière, Albergo, **GK**, Shanahan, Cranmer ICML(2020) 2002.02428]

Results for $U(1)$ gauge theory

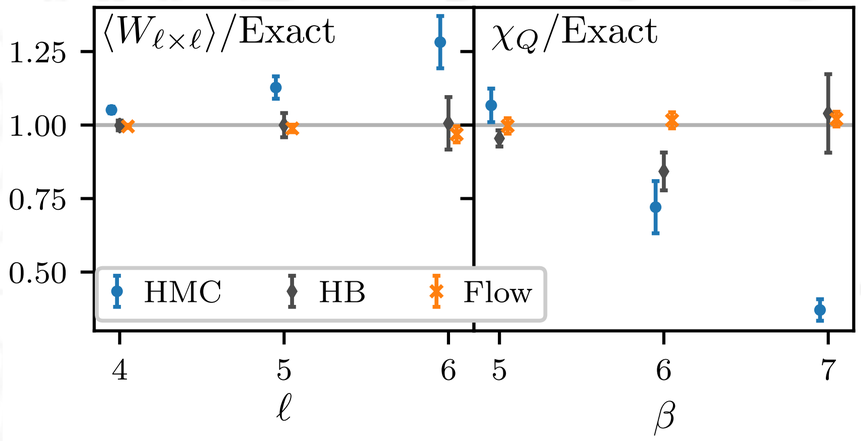
There is exact lattice topology in 2D.

$$Q = \frac{1}{2\pi} \sum_x \arg(P_{01}(x))$$



Compared flow, analytical, HMC, and heat bath on 16×16 lattices for $\beta = \{1,\ldots,7\}$

- Topo freezing in HMC and heat bath
- Flow-based MCMC observables agree with analytical



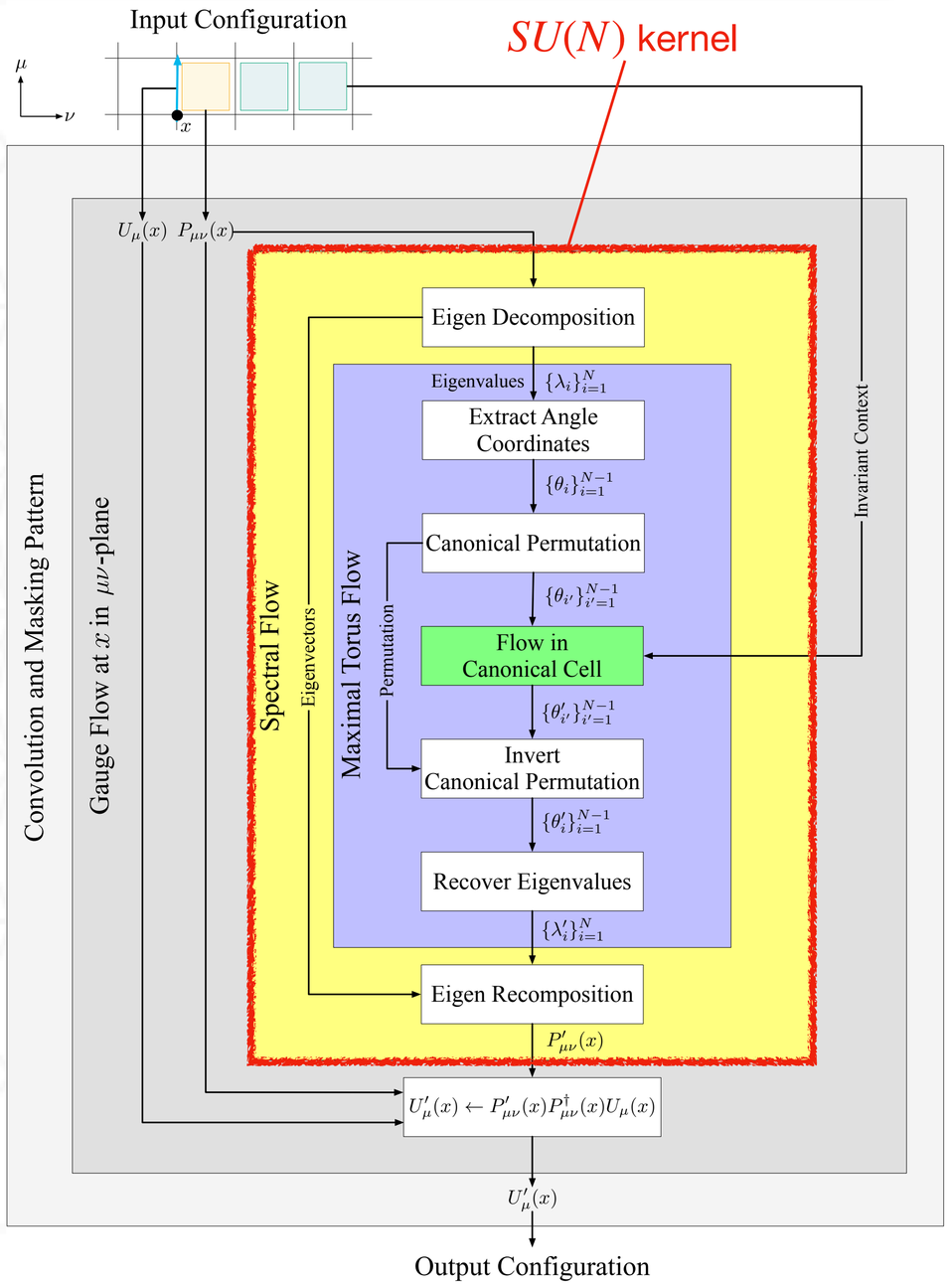
$$W_{\ell\times\ell} = \prod_{x\in\ell\times\ell} P_{01}(x) \qquad \chi_Q = \langle Q^2/V\rangle$$

Kernel for $SU(N)$ theories

Intuition: should move points between conjugacy classes, without moving around within CCs

Conjugacy classes for $SU(N)$ described by **spectrum** of the matrix: unordered set of eigenvalues. Kernel should transform spectrum!

- Act on list of eigenvalues
- Equivariant under permutations



Learning $SU(N)$ plaquette distributions

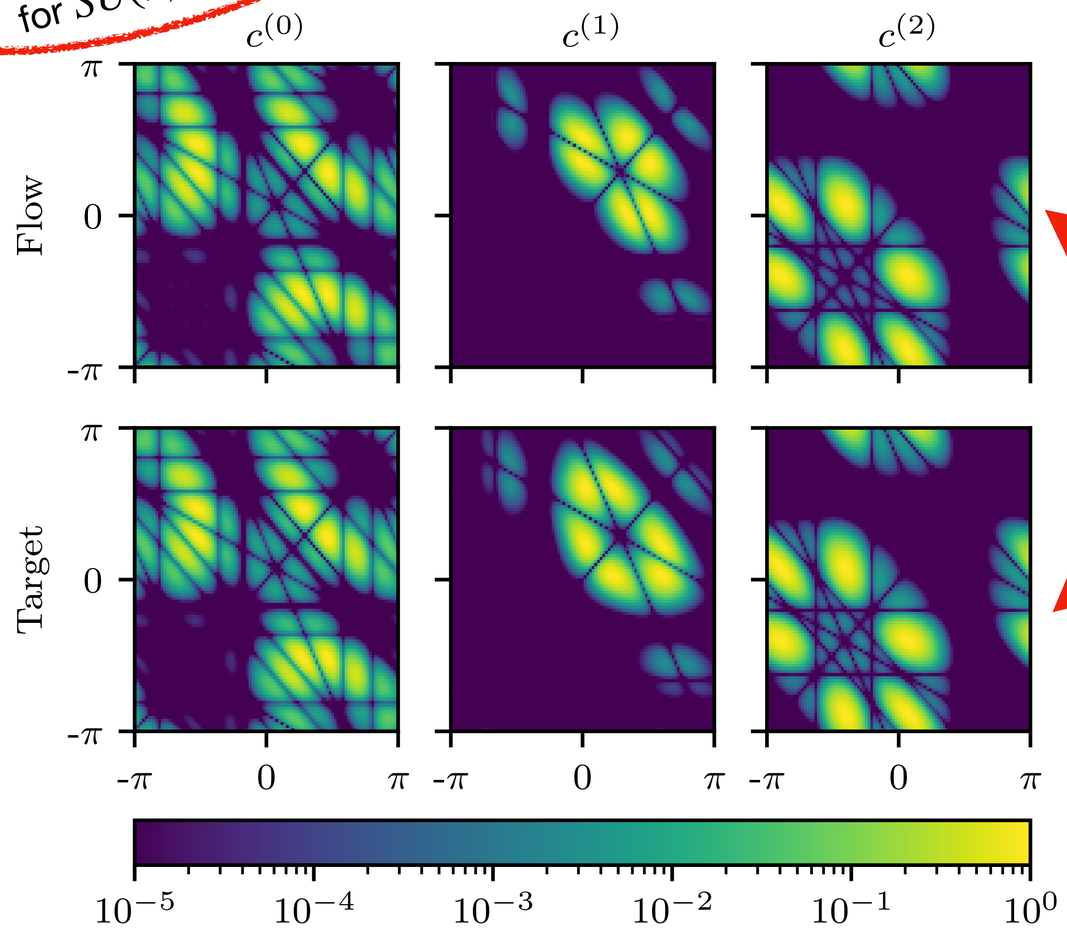
Tested kernel in isolation by learning distributions on a single $SU(N)$ variable

- Representative of marginal distribution on untraced plaquette
- Conjugation-invariant action with several choices of coefficients

$$S_i(U) := -\frac{\beta}{N}\mathrm{Re}\,\mathrm{tr}\left[\sum_n c_n^{(i)} U^n\right]$$

- Tests for $SU(2)$ to $SU(100)$

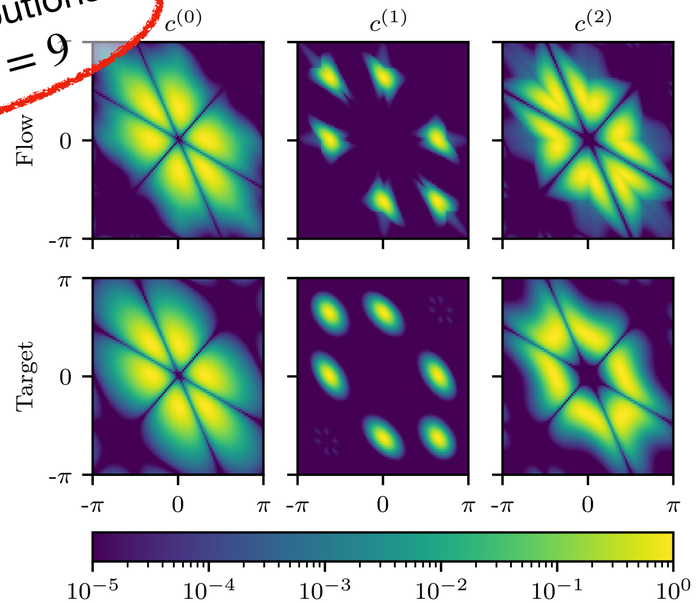
Plaquette distributions for $SU(9)$, $\beta = 9$



Agree!

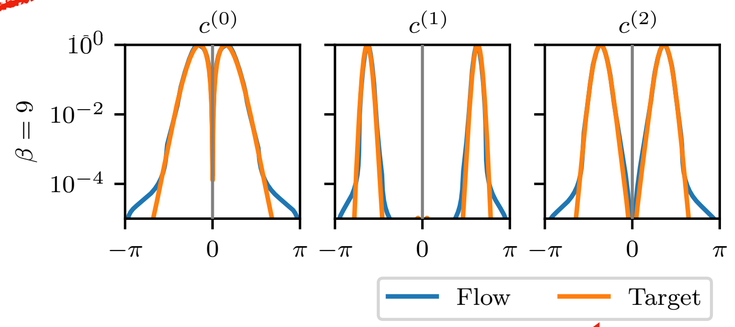
Density has zeros on vertical, horizontal, and diagonal lines where the slice crosses walls of cells

Plaquette distributions for $SU(3)$, $\beta = 9$



Agree!

Plaquette distributions for $SU(2)$, $\beta = 9$



Agree!

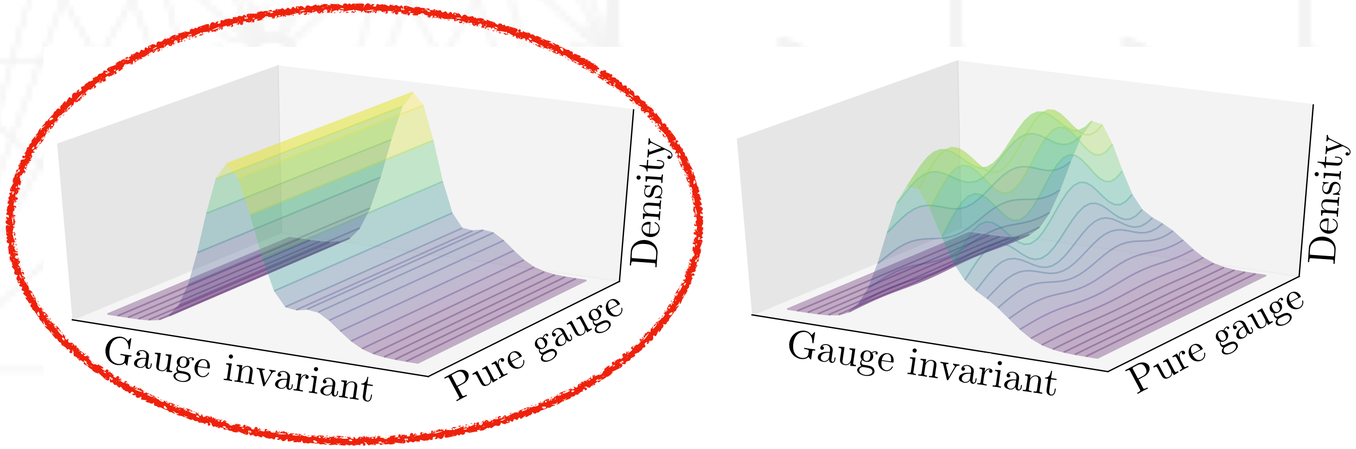
Learning $SU(2)$ and $SU(3)$ gauge theory

Normalizing flows trained for 2D lattice gauge theory on 16×16 lattices.

- Approx equal 't Hooft couplings:
  $\beta = \{1.8, 2.2, 2.7\}$ for $SU(2)$ and
  $\beta = \{4.0, 5.0, 6.0\}$ for $SU(3)$

- 48 coupling layers, update all links 6 times

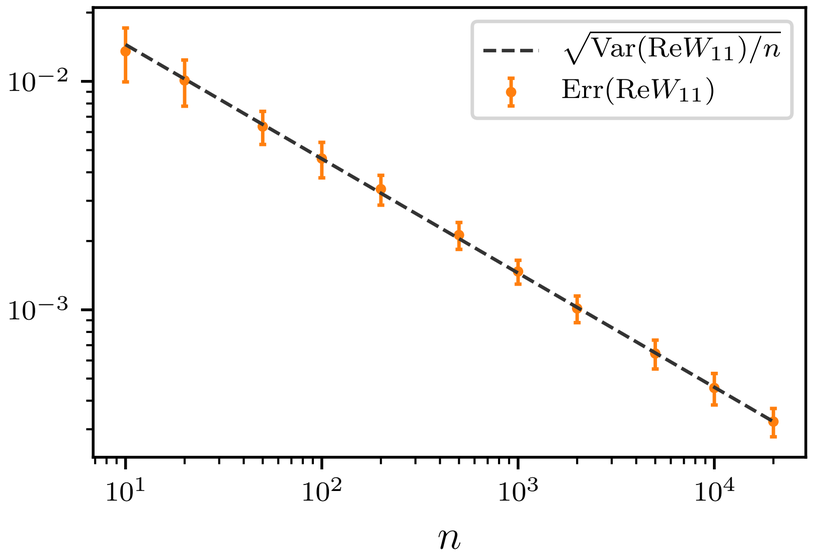
- Kernels suitable for $SU(N)$ for **exact gauge invariance, exact subgroup of translational invariance**

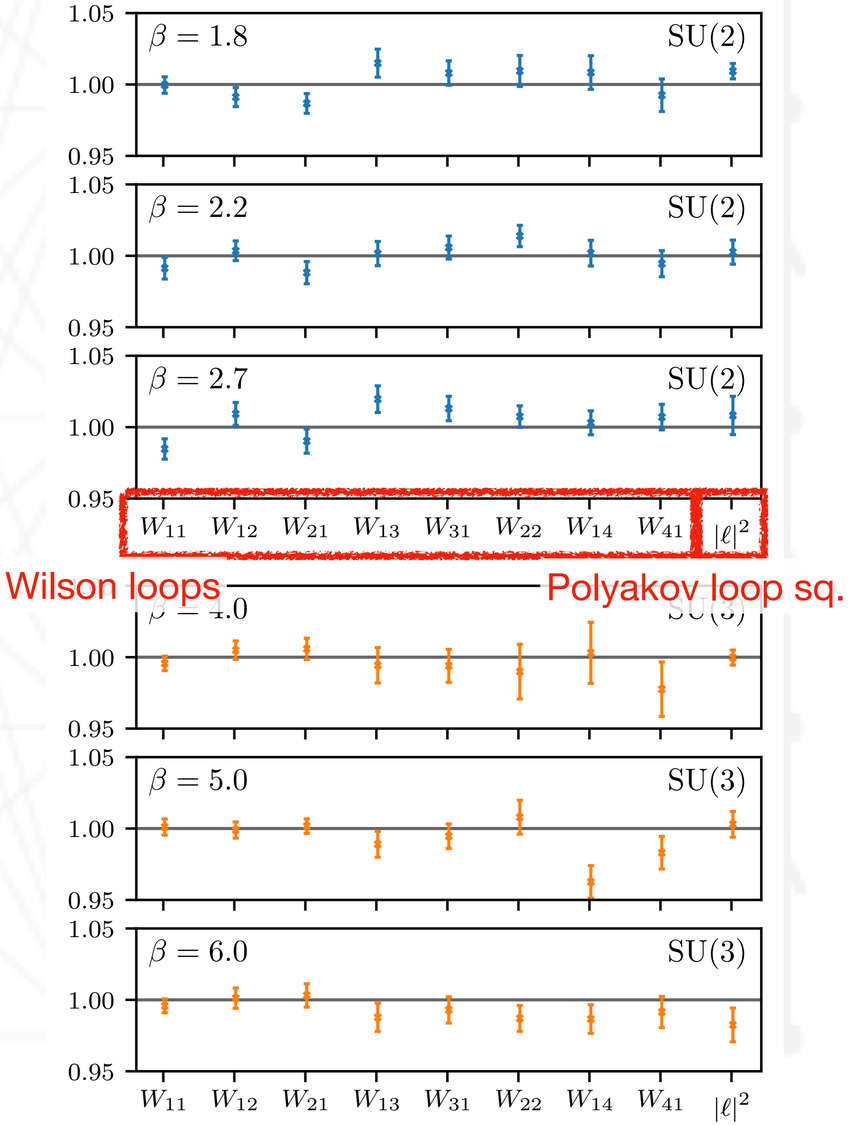


Results for $SU(2)$ and $SU(3)$ gauge theory

- Flow-based MCMC observables agree with analytical
- High-quality model: autocorrelation time in flow-based Markov chain τ_{int} = 1—4
- $1/\sqrt{n}$ scaling with number of samples n

thinned by τ_{int}!





Summary

- Flow-based MCMC gives exact results from approximate model proposals
  - Inaccuracies in the model = increased autocorrelation time
- Gauge symmetry can be incorporated without breaking (most of) translational symmetry
  - Gauge equivariant coupling layers
  - Kernels for $U(1)$ and $SU(N)$

And a simple alternative for $U(N)$ in our paper!

- High-quality models produced for
  - Lattice scalar theory in 1+1D
  - $U(1)$, $SU(2)$, and $SU(3)$ lattice gauge theory in 1+1D

Outlook

Several directions for future work:

1. Choices of untraced loops to transform, gauge-inv loops as input
2. Performance on multimodal distributions?
3. Training hyperparameter tuning, different model arch for inner flows
4. Scaling of required model complexity in taking continuum limit?
5. Incorporation of dynamical fermions

Outlook

If the method can be scaled to state-of-the-art calculations, ensemble generation could look like…

easily parallelized

… a single up-front cost to train a model

… more efficient parameter sweeps (retrain a model from nearby params)

easily parallelized

… cheap gauge field generation

… reduced storage costs (store/transfer the model, not configs!)

… or, no storage costs? (generate configs on the fly for measurements?)

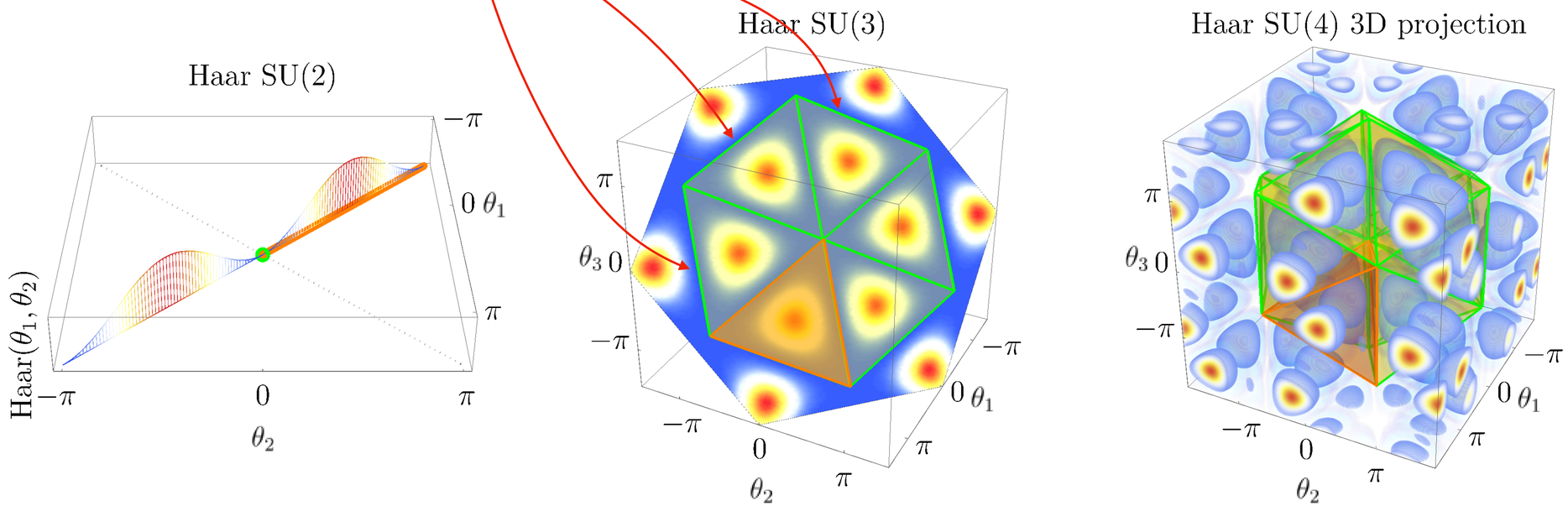
In the upcoming exascale era, exploiting massively parallel resources will be key!

Backup slides

Permutations on $SU(N)$ eigenvalues

Eigenvalues satisfy $\prod_k \lambda_k = 1 \rightarrow \boxed{\sum_k \theta_k = 0 \pmod{2\pi}}$ **maximal torus** of $SU(N)$

Permutations exchange **cells** in the space of $\{\theta_k\}$



Approach: map input to a “canonical” cell, transform within cell, undo canonical map

$SU(N)$ test plaquette distributions

Tested kernel in isolation by learning distributions on a single $SU(N)$ variable

- Representative of marginal distribution on untraced plaquette
- Conjugation-invariant action with several choices of coefficients

$$S_i(U) := -\frac{\beta}{N}\mathrm{Re\,tr}\left[\sum_n c_n^{(i)} U^n\right]$$

set i	$c_1^{(i)}$	$c_2^{(i)}$	$c_3^{(i)}$
0	1	0	0
1	0.17	-0.65	1.22
2	0.98	-0.63	-0.21

Exact marginal distribution on plaquettes

Randomly sampled coeffs with 1 mode

- Special case implementations tested for $SU(2)$ and $SU(3)$
- Unoptimized generic implementation learns $c^{(0)}$ for $SU(4), \ldots, SU(100)$

Symmetries in $SU(2)$ and $SU(3)$ models

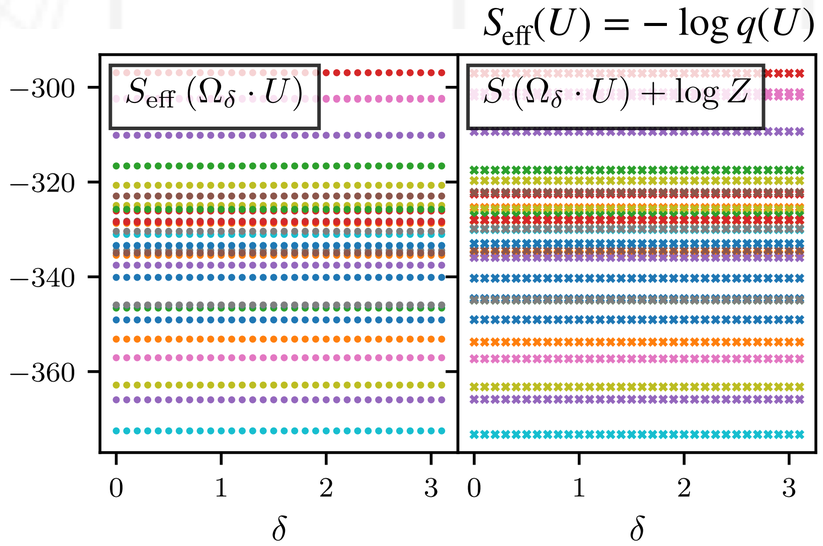
Exact **gauge symmetry** by construction.

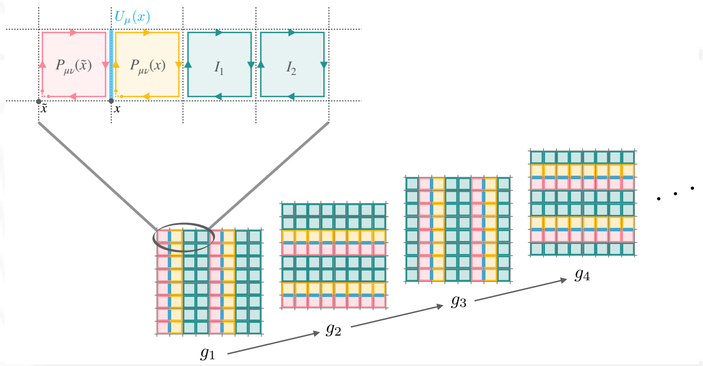
Exact **center symmetry** due to choice of loops:

- Plaquettes invariant under center symm
- Should include Polyakov loops if center symmetry explicitly broken in theory

Large subgroup of **translational symmetry**

- $\mathbb{Z}_4 \times \mathbb{Z}_4$ breaking due to 4-site spacing in masking pattern
- 16-elt residual group to be learned, independent of volume





Scalar theory on a 2D lattice

$$S_E(\phi) = \sum_x \left(\sum_y \phi(x) \Box(x,y) \phi(y) + m^2 \phi(x)^2 + \lambda \phi(x)^4 \right)$$

- Real DoF per lattice site, $\phi(x) \in \mathbb{R}$
- 6×6 through 14×14 lattices studied
- Scalar particle mass tuned to give correlation length ~ $L/4$
- Affine coupling layers for flow models:

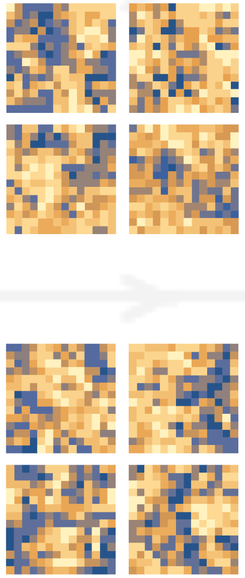
$$\phi'(x) = e^s \phi(x) + t$$

Context functions, implemented using NNs acting on frozen sites (checkboard pattern)

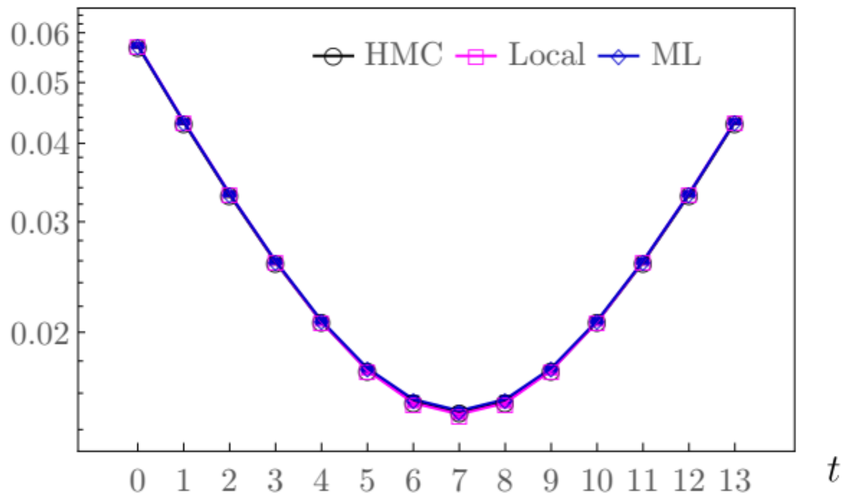
Flow model samples

~ vs ~

HMC samples



$\tilde{G}_c(0,t)$ Connected Green's function…



… and several other observables all consistent

Related approaches

Generative Adversarial Networks (GANs):

- Highly expressive!
- Work in the direction of GANs for lattice

  [Urban, Pawlowski 1811.03533]

  [Zhou, Endrődi, Pang, Stöcker 1810.12879]

Variational AutoEncoders (VAEs):

- Can also learn meaningful directions in the prior variables

However: No access to $q(U)$… hard to make exact!

[Karras, Lane, Aila / NVIDIA 1812.04948]

These are machine learned faces!

[Shen & Liu 1612.05363]

These are machine learned faces!

Optimizing ("training") the model

Must not require a large number of samples from real distribution to optimize!

Self-training:

- Optimize model params using stochastic gradient descent on a loss function
- Loss function = modified Kullback-Leibler (KL) divergence

Measures difference between probability distributions

$$D_{\mathrm{KL}}(q\,||\,p) := \int \mathscr{D}U\, q(U)\big[\log q(U) - \log p(U)\big] \geq 0$$

Constant shift removes unknown normalization

$$D'_{\mathrm{KL}}(q\,||\,p) := \int \mathscr{D}U\, q(U)\big[\log q(U) + S(U)\big] \geq -\log Z$$

- To estimate loss for grad. descent, draw samples from the **model**, measure sample mean of $\big[\log q(U) + S(U)\big]$

Translational equivariance with CNNs

1. Make context functions Convolutional Neural Nets.

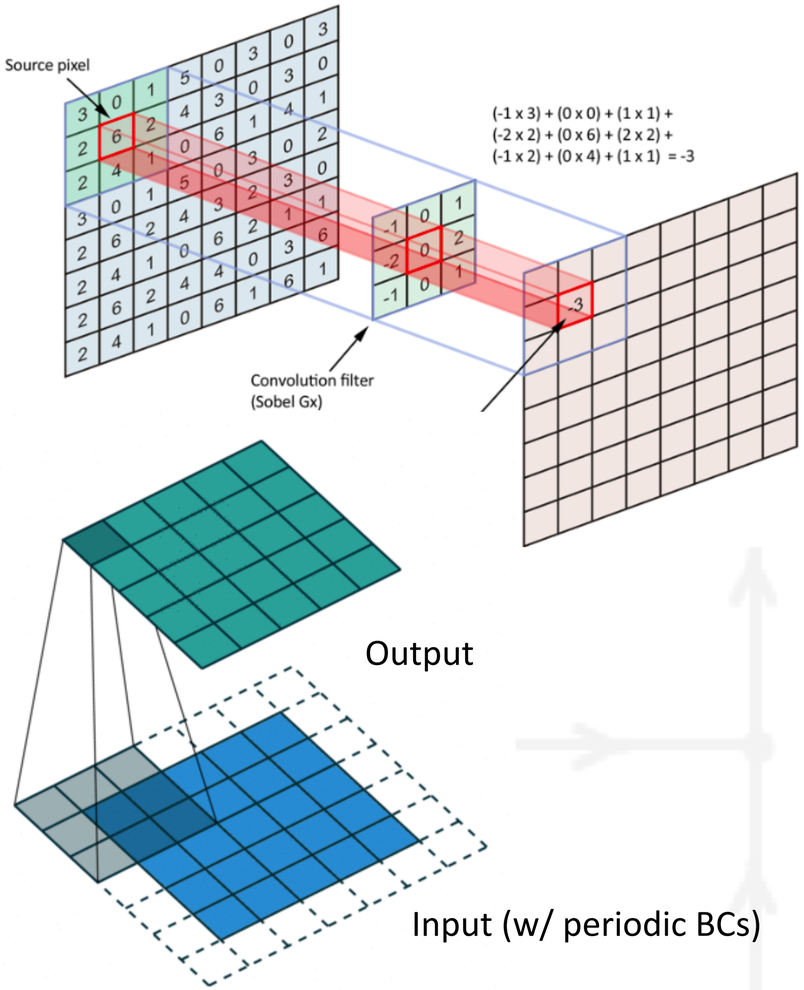
CNNs:

- Compute output value for each site from linear transform of nearby DOF only
- Reuse same weights, scanning kernel across the lattice

CNNs are equivariant under translations.

2. Make masking pattern (mostly) translationally invariant.

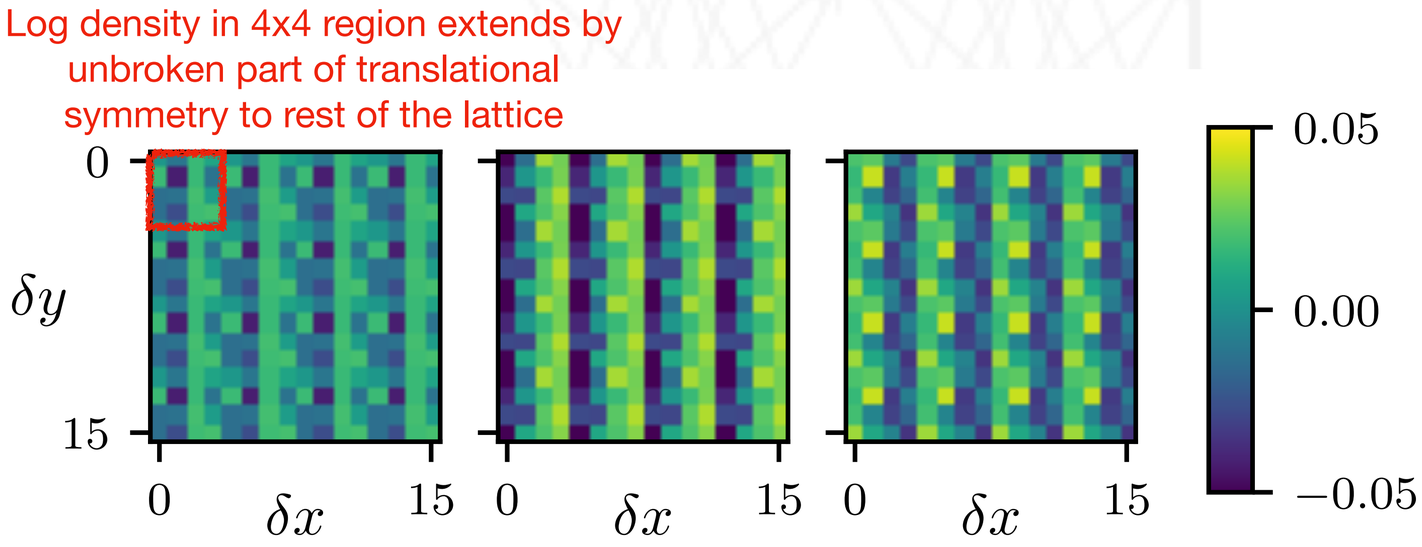
- Required to ensure whole coupling layer is equiv
- Our application: translational equiv modulo $\mathbb{Z}_4 \times \mathbb{Z}_4$



[Images from freecodecamp.org]

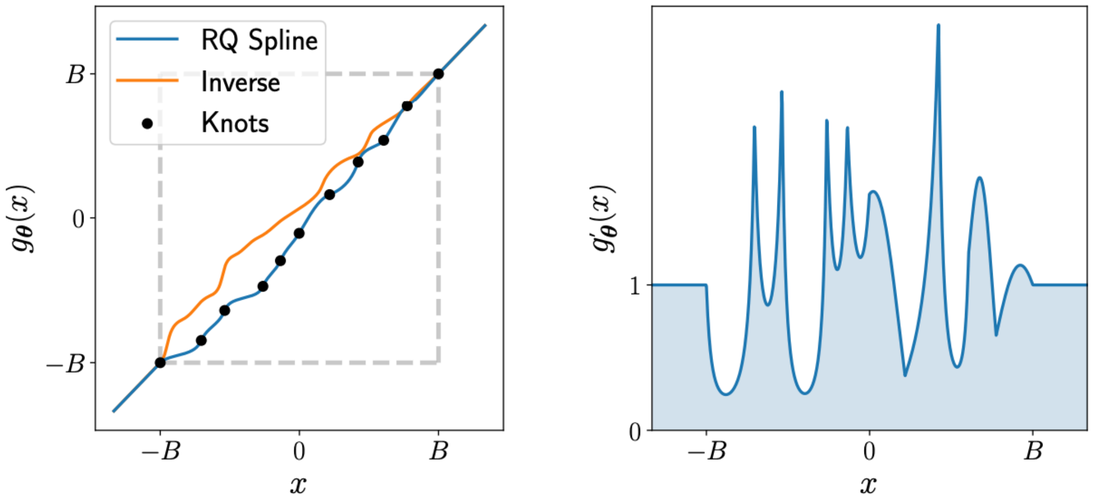
Translational symmetry breaking pattern

- Masking patten = repeating tile of size 1×4
- Rotate / translate the pattern between layers
- $\mathbb{Z}_4 \times \mathbb{Z}_4$ symmetry breaking



Details of $SU(2)$ models

- Inner flow on open box Ω is a spline flow with **4 knots**
  - B and $-B$ boundaries align to 0 and 1 edges of the open box

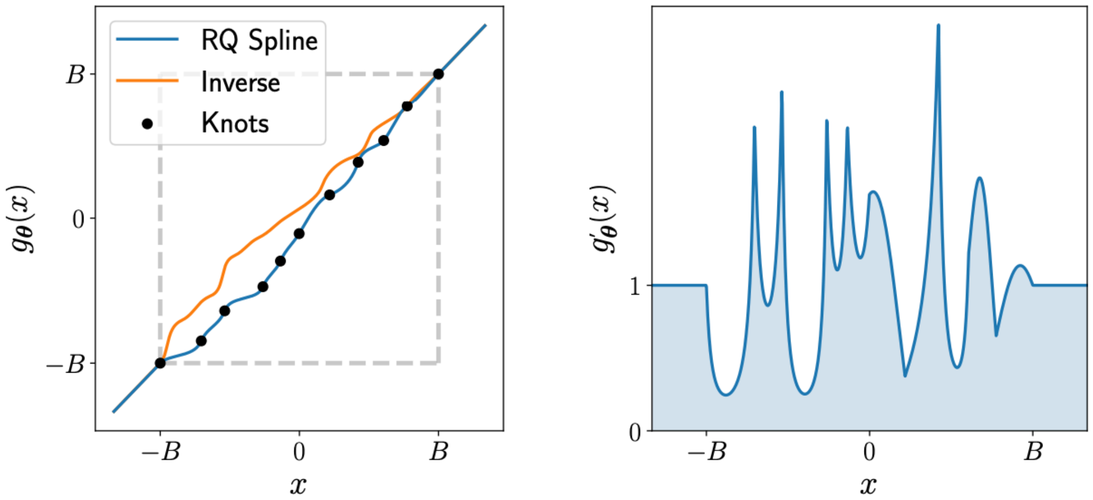


[Durkan, Bekasov, Murray, Papamakarios 1906.04032]

- CNNs to compute the knot locations
  - 32 hidden channels
  - 2 hidden layers

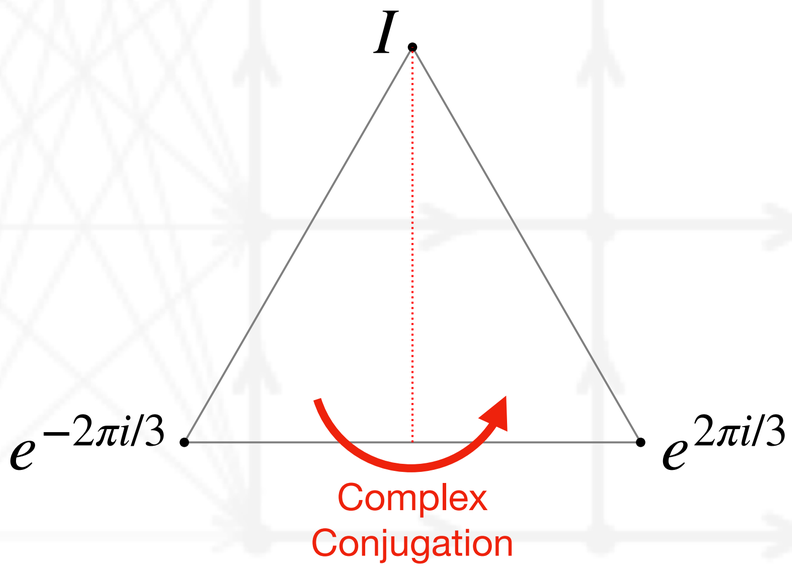
Details of $SU(3)$ models

- Inner flow on open box Ω is a spline flow with **16 knots**
  - B and $-B$ boundaries align to 0 and 1 edges of the open box



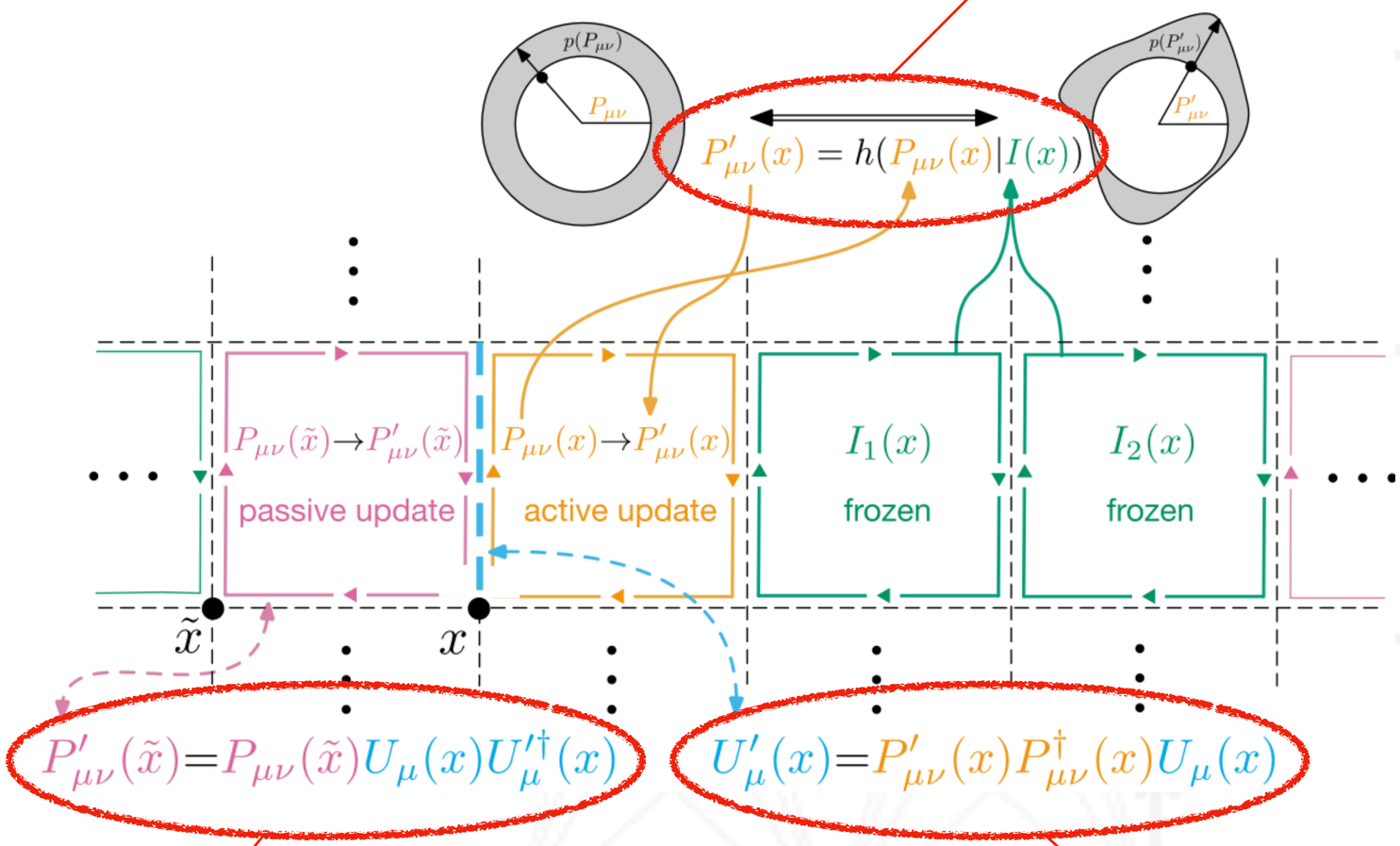
[Durkan, Bekasov, Murray, Papamakarios 1906.04032]

- CNNs to compute the knot locations
  - 32 hidden channels
  - 2 hidden layers
- Exact conjugation equivariance also imposed



Gauge equivariance (details)

1. Kernel acts on subset of untraced plaquettes



3. Some plaquettes **passively** updated as a result of link changing

2. Link **absorbs** update via left-multiplication (invertible!)

Map into canonical cell

Want to permute eigenvalues into canonical order

- Sorting doesn't work directly: discontinuities when θ_k jumps across the $\pm\pi$ boundary
- Simply trying all $N!$ permutations is slow for large N
- Need to ensure permutation taking points in the same cell to canonical is the same

Short algorithm based on sorting works; Algorithm 1 of [‡]

[‡] [Boyda, **GK**, Racanière, Rezende, Albergo, Cranmer, Hackett, Shanahan 2008.05456]

Transform within canonical cell

Require: boundary-preserving transformation within the cell

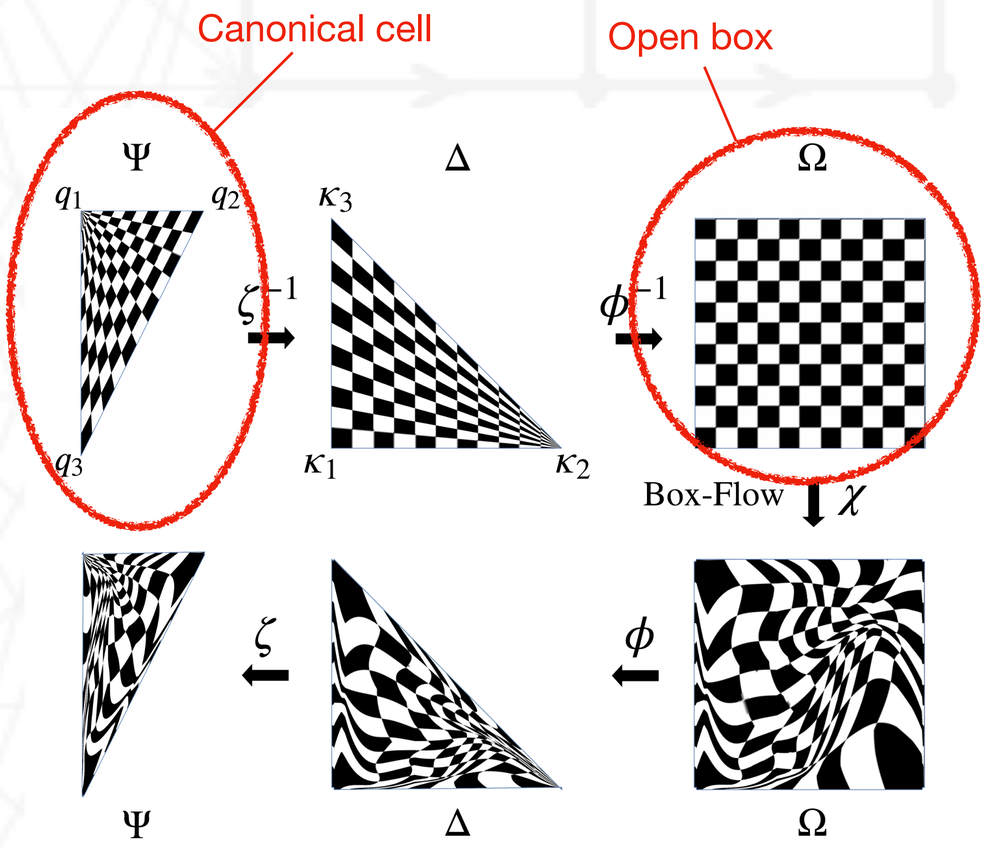
- Cell is a simplex bounded by vertices $\{y_k\}$,

$$[y_k]_j := 2\pi\left(\frac{k}{N} - \delta_{k\geq j}\right)$$

- Hard to transform directly, instead we first change coordinates to an open box $\Omega = (0,1)^N$

Composition of two maps, ζ and ϕ, give the change of coordinates

Boundary preserving map using **fixed-interval spline transformations** on each coordinate of Ω

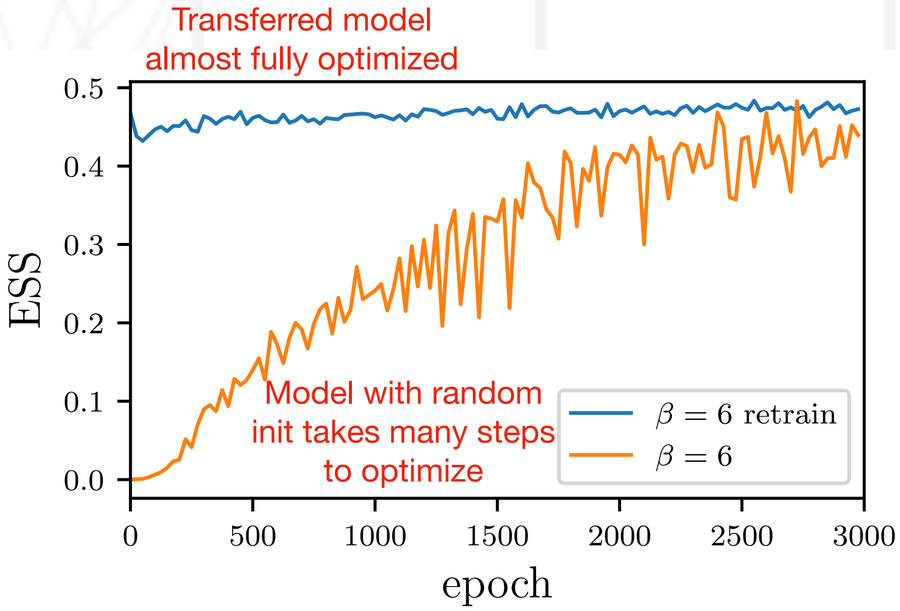


Gauge theory model training

- Adam optimizer ~ stochastic grad. descent with momentum
  - Batches of size 3072 per gradient descent step
  - Monitored value of **effective sample size** (ESS)

$$\mathrm{ESS} = \frac{\left(\frac{1}{n}\sum_i w(U_i)\right)^2}{\frac{1}{n}\sum_i w(U_i)^2}, \quad U_i \sim q(U)$$

$$w(U) = p(U)/q(U)$$ "reweighting factors"



- Transfer learning: model trained first on 8×8 then used to initialize model for training on 16×16