Neural quantum states for lattice field theory

Thomas Spriggs, Eliska Greplova, Jannes Nys

TU Delft, QuTech, ETH-Zurich

QT4HEP

In a nutshell

- We want to find a *representation* of the ground state wavefunction of the SU(2) Yang-Mills Hamiltonian
- Our end product will be a neural network that will do the following:
  - When given a specific configuration of the degrees of freedom, it tells us the amplitude this state has in the wavefunction*
- So, not the full wavefunction, but we can get each element as we see fit
- **This is enough to do physics**

*This is similar to the role of the action in standard lattice QCD

In a nutshell
(Example from a spin model)

- For two spins, the ground state wavefunction in a given basis may look like

$$\psi = \frac{1}{4}|00> + \frac{1}{4}|01> + \frac{1}{4}|10> + \frac{\sqrt{13}}{4}|11>$$

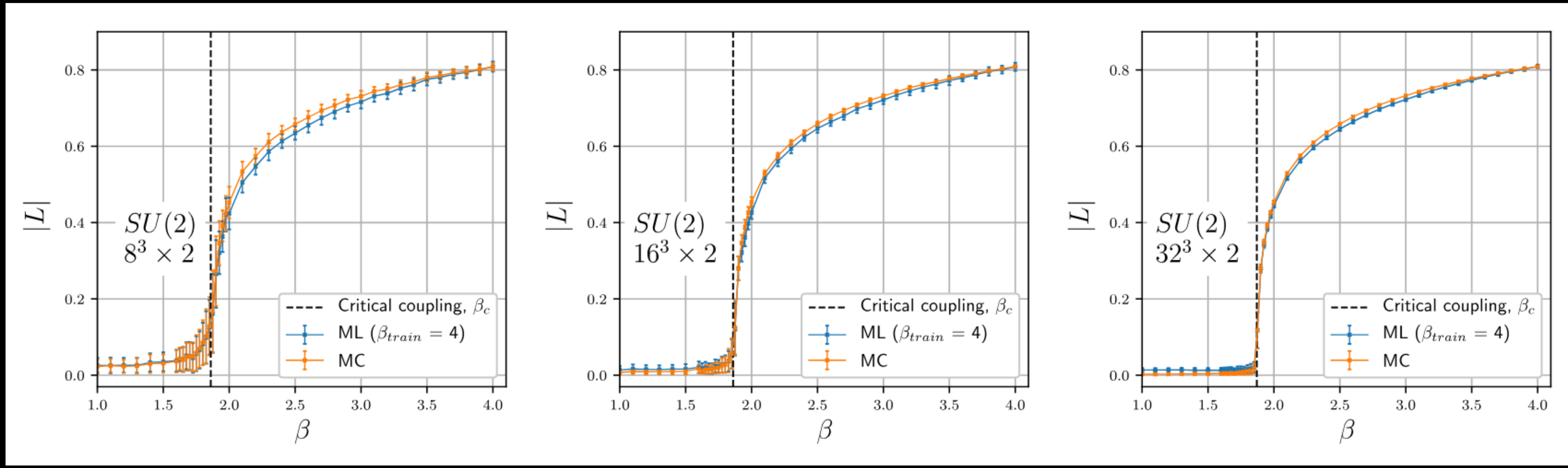
- Once our network is trained, we should be able to ask:
    - Me: “I am looking at the $|11>$ state, what’s its amplitude?”
    - Network: “It’s $\frac{\sqrt{13}}{4}$ mate”
    - Me: “Nice one, cheers”

Machine learning for lattice gauge theories

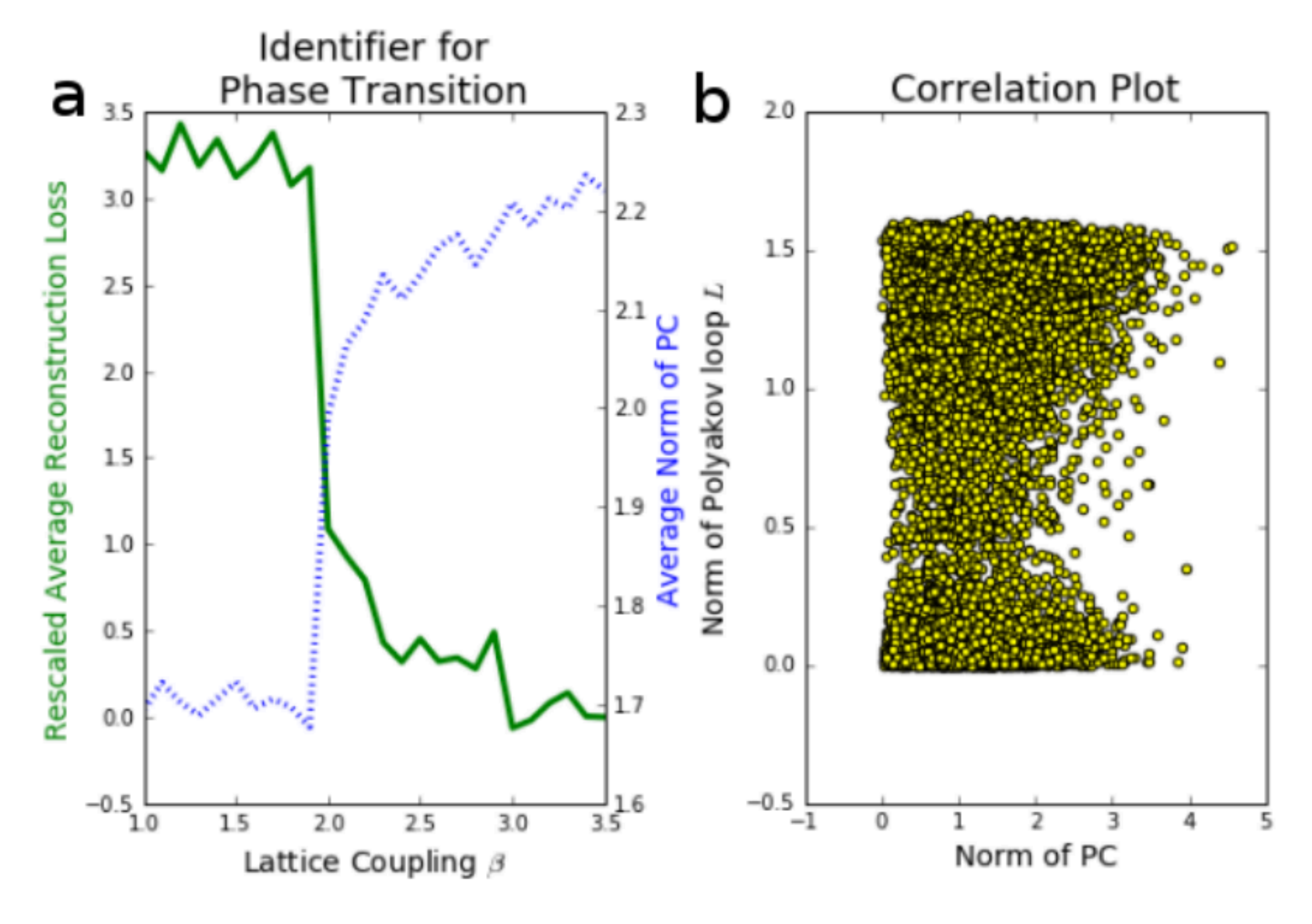
This is not an exhaustive list

(Nice review) Aarts, G. *et al.* Nat. Phys. Rev. 1-10

Phase transition detection

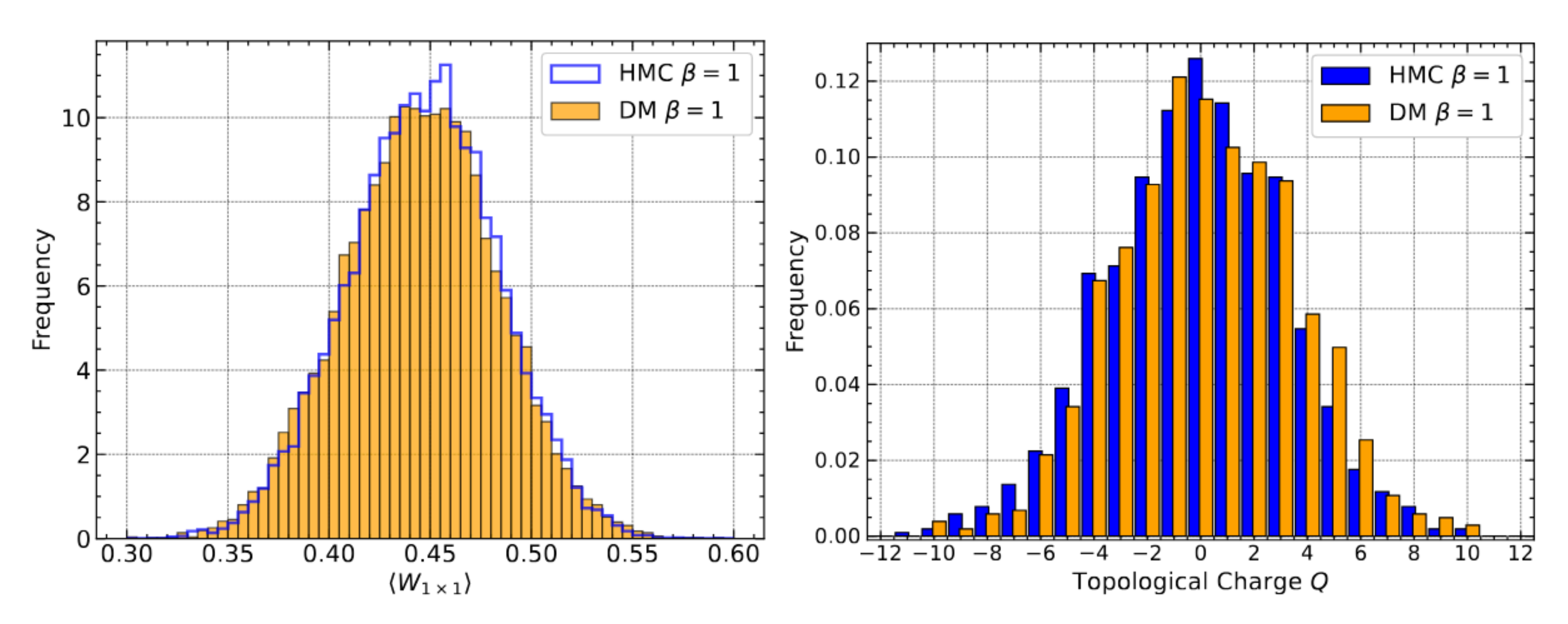


Neural network correctly identifying confinement phase transition despite only being trained far from the transition.
From Boyda, D.L. *et al.* Phys. Rev. D **103**, 014509

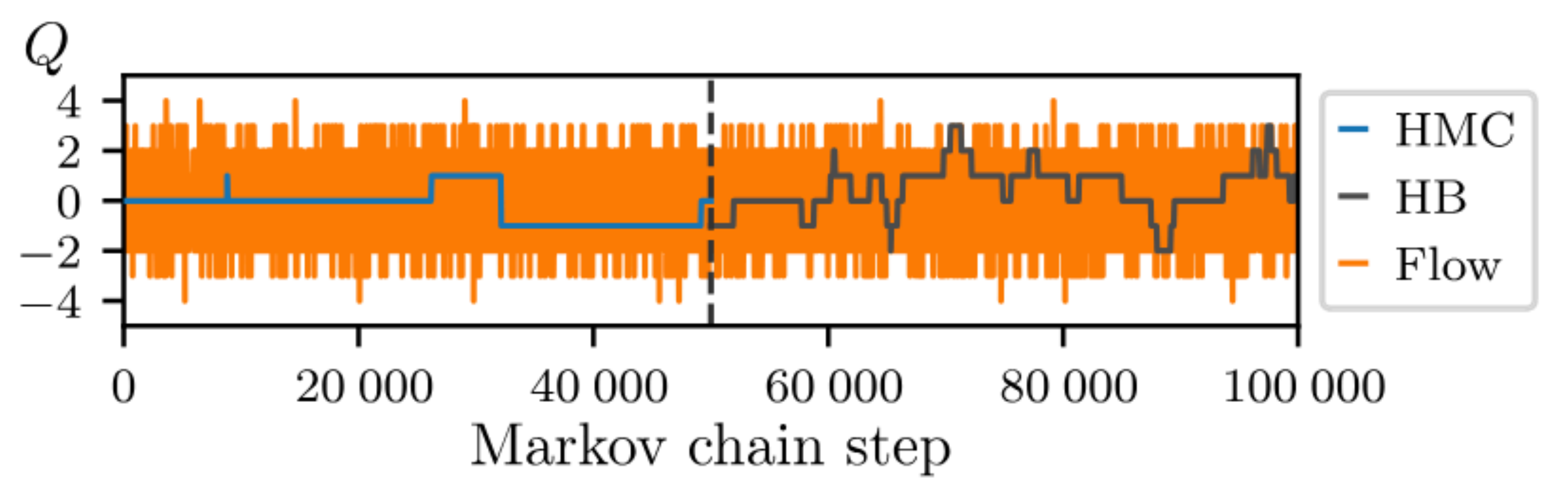


Principle components acting as an order parameter of the confinement phase transition.
From Wetzel, S.J., Scherzer, M. Phys. Rev. B **96**, 184410

Configuration generation (Euclidean)

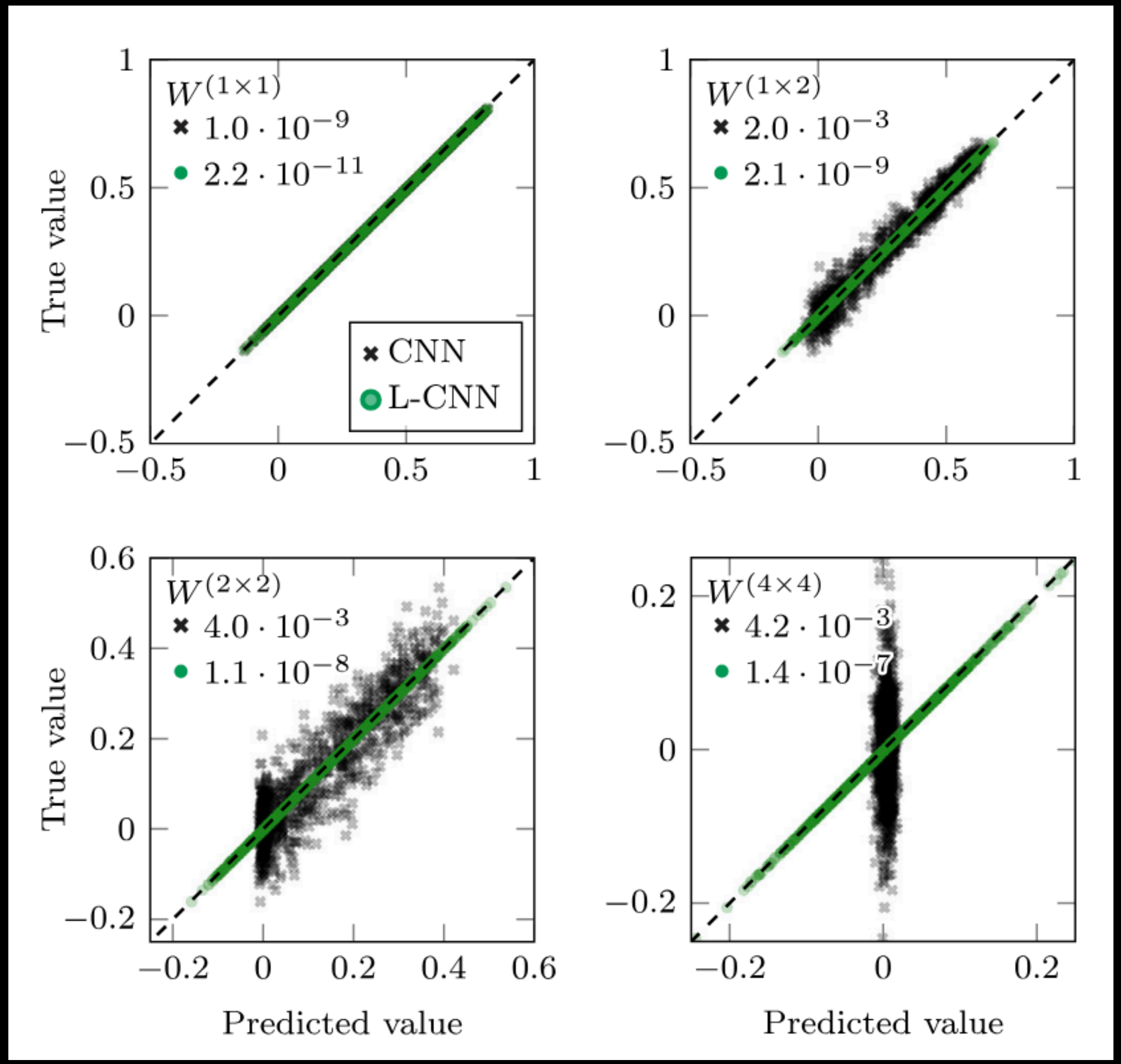


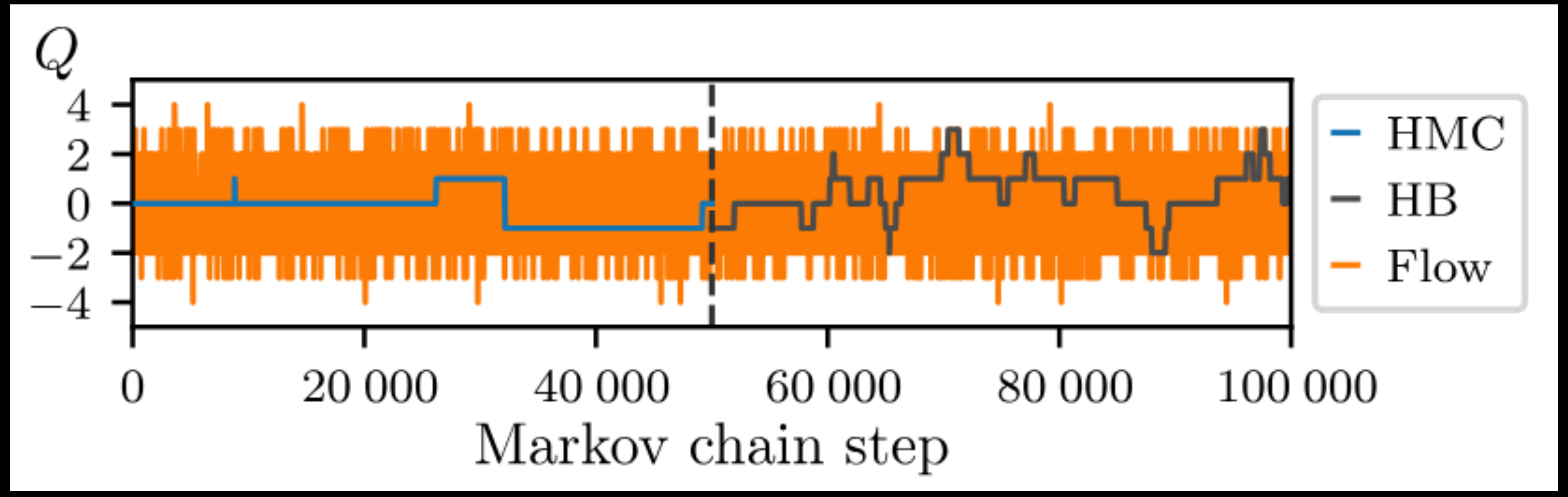
Diffusion models generating the same distribution found by standard methods.
From Zhu, Q. *et al.* NeurIPS 2024, arXiv:2410.19602



Normalising flows generating ensembles that explore the Hilbert space more evenly than conventional methods.
From Kanwar, G. *et al.* Phys. Rev. Lett. **125**, 121601

SU(N)-specific architectures





Gauge equivariant convolutional network identifying the average Wilson loop when a standard convolutional network fails.
From Favoni, M. *et al.* Phys. Rev. Lett. **128**, 032003

Normalising flows generating ensembles that explore the Hilbert space more evenly than conventional methods.
From Kanwar, G. *et al.* Phys. Rev. Lett. **125**, 121601

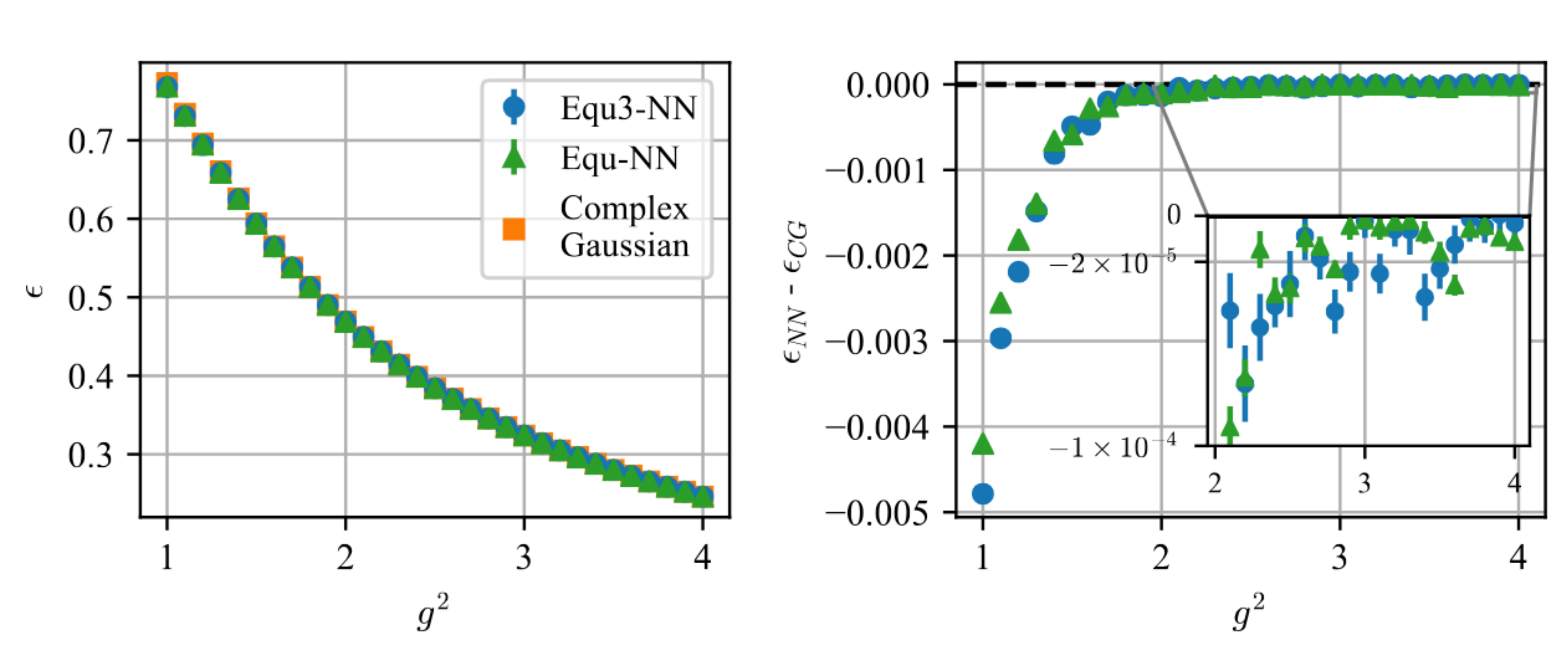
& this work

Drawback

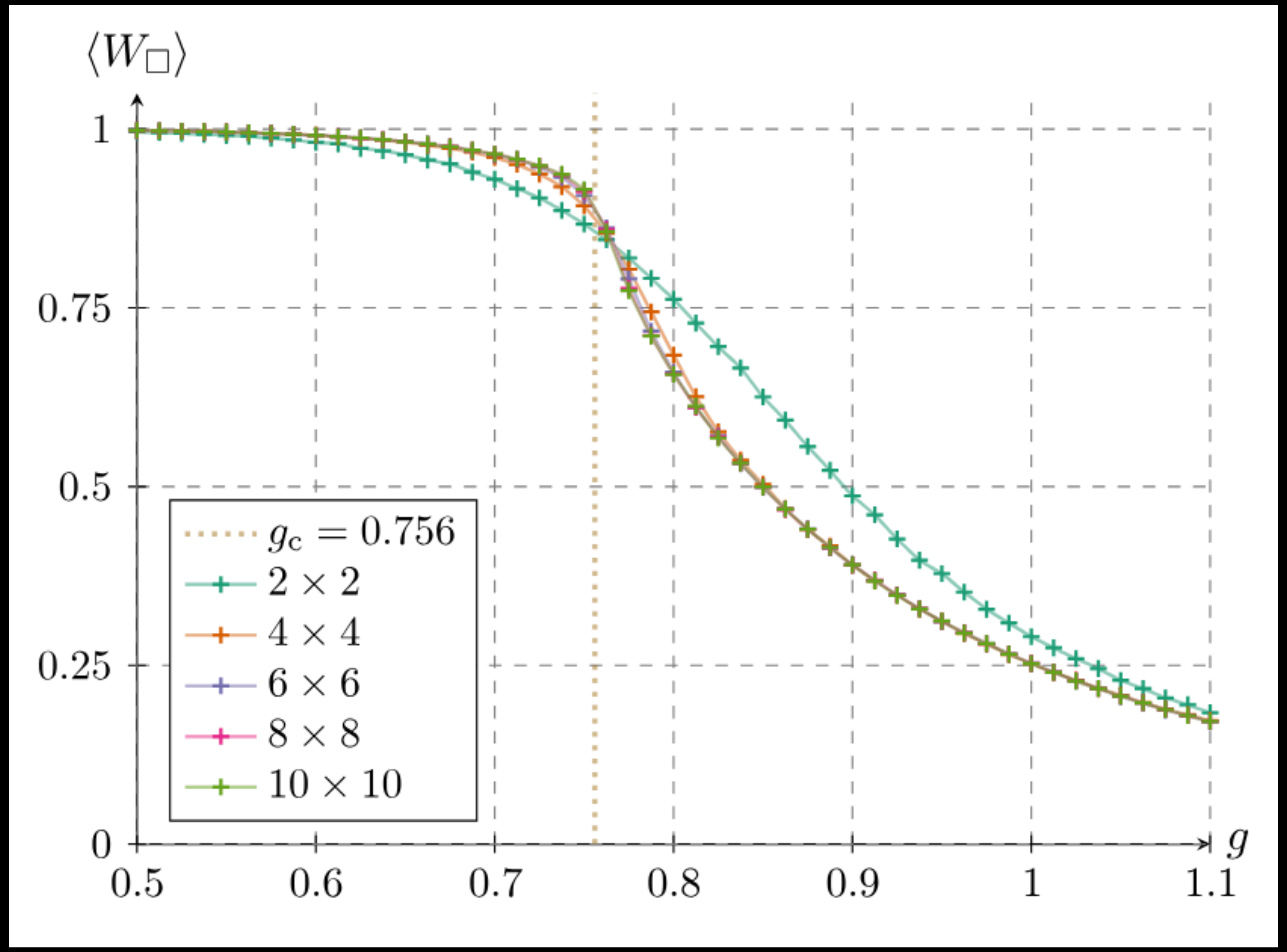
These are all using the Lagrangian formulation of SU(N)

- Time is imaginary, discrete, and the number of points in time must be set *a priori*
- Many physical observables are only accessible through some contrived correlators
- Any imaginary component in the action causes issues (the sign problem)

Learning the wavefunction (Hamiltonian)



Computing the ground state energy of 2+1 dimensional U(1) gauge theory.
From Luo, D. arXiv:2211.03198



Observing finite size scaling of a phase transition in Z2 gauge theory.
From Apte, A. Phys. Rev. B **110**, 165133

& this work

Lagrangian vs Hamiltonian formalisms

Lagrangian vs Hamiltonian formalisms

- Collect a set of configurations distributed in some way to act as a representative sample of all possible configurations

Lagrangian vs Hamiltonian formalisms

- Collect a set of configurations distributed in some way to act as a representative sample of all possible configurations

- The **known** action is the distribution that you want to mimic

Lagrangian vs Hamiltonian formalisms

- Collect a set of configurations distributed in some way to act as a representative sample of all possible configurations
- The **known** action is the distribution that you want to mimic
- You have to **learn** the distribution (this can introduce a bias)
    - To learn the ground state distribution you want to minimise $E = <\psi|\mathscr{H}|\psi>$

Lagrangian vs Hamiltonian formalisms

- Collect a set of configurations distributed in some way to act as a representative sample of all possible configurations
- The **known** action is the distribution that you want to mimic
- You have to **learn** the distribution (this can introduce a bias)
    - To learn the ground state distribution you want to minimise $E = < \psi | \mathscr{H} | \psi >$
- $P(\mathbf{U}) \sim \exp(-S(\mathbf{U}))$

Lagrangian vs Hamiltonian formalisms

- Collect a set of configurations distributed in some way to act as a representative sample of all possible configurations
- The **known** action is the distribution that you want to mimic
- You have to **learn** the distribution (this can introduce a bias)
  - To learn the ground state distribution you want to minimise $E = <\psi|\mathscr{H}|\psi>$
- $P(\mathbf{U}) \sim \exp(-S(\mathbf{U}))$
- $P(\mathbf{U}) \sim |\psi(\mathbf{U})|^2$

Lagrangian vs Hamiltonian formalisms

- Collect a set of configurations distributed in some way to act as a representative sample of all possible configurations
- The **known** action is the distribution that you want to mimic
- You have to **learn** the distribution (this can introduce a bias)
  - To learn the ground state distribution you want to minimise $E = < \psi | \mathscr{H} | \psi >$
- $P(\mathbf{U}) \sim \exp(-S(\mathbf{U}))$
- $P(\mathbf{U}) \sim |\psi(\mathbf{U})|^2$

Neural network

Neural quantum states

Variational wavefunction

Variational quantum states

- Let $\psi = \sum_i a_i | \phi_i >$

- 2 qubits, spin basis: $\psi = a_0 | 00 > + a_1 | 01 > + a_2 | 10 > + a_3 | 11 >$

- "Finding ground state" == finding a function that gives four numbers

$$\psi = \sum_i f(\phi_i) | \phi_i >$$

Neural quantum states

Carleo, G., Troyer, M., Science **355**, 602
(Nice review) Medvidović, M., Moreno, J.R., Eur. Phys. J. Plus **139**, 631

Neural quantum states

- Want to *learn* the function, f, by using a neural network

Carleo, G., Troyer, M., Science **355**, 602
(Nice review) Medvidović, M., Moreno, J.R., Eur. Phys. J. Plus **139**, 631

Neural quantum states

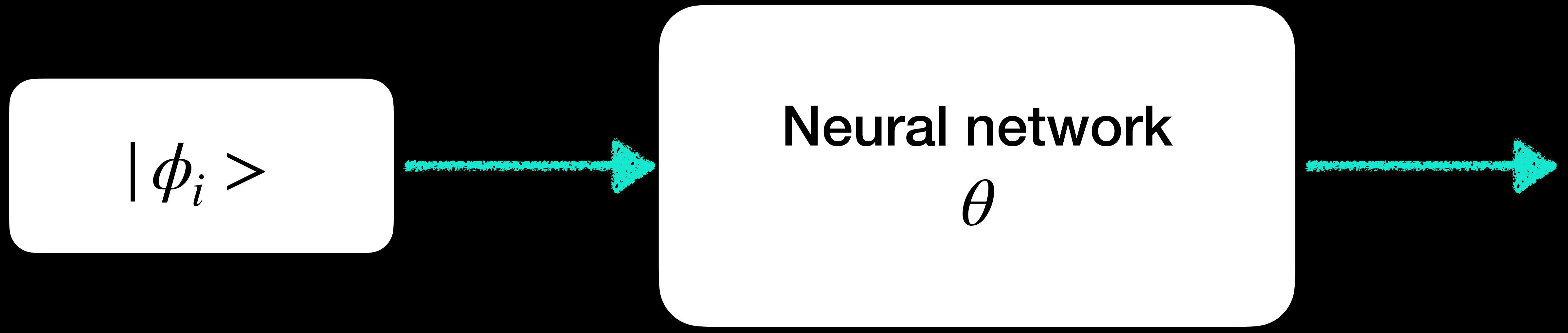
- Want to *learn* the function, f, by using a neural network

$$|\phi_i>$$

Carleo, G., Troyer, M., Science **355**, 602
(Nice review) Medvidović, M., Moreno, J.R., Eur. Phys. J. Plus **139**, 631

Neural quantum states

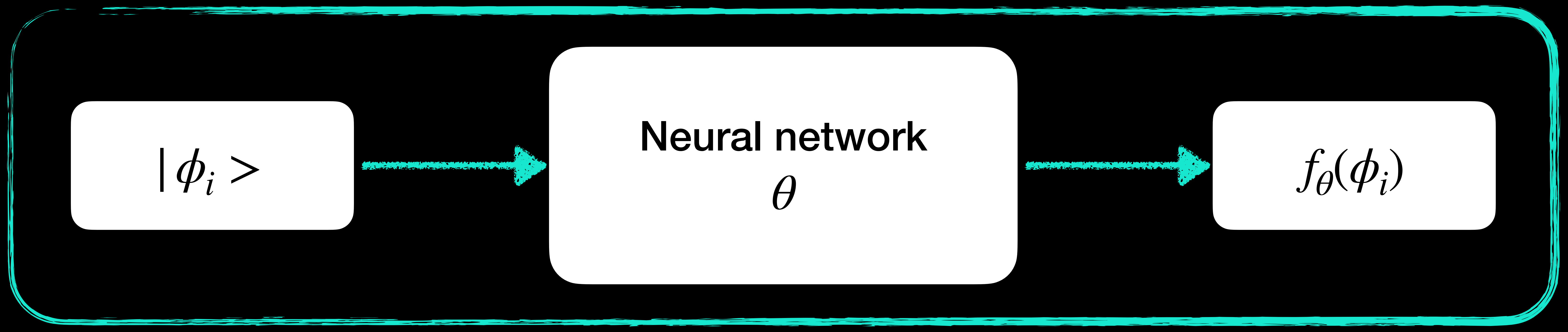
- Want to *learn* the function, f, by using a neural network



Carleo, G., Troyer, M., Science **355**, 602
(Nice review) Medvidović, M., Moreno, J.R., Eur. Phys. J. Plus **139**, 631

Neural quantum states

- Want to *learn* the function, f, by using a neural network



Carleo, G., Troyer, M., Science **355**, 602
(Nice review) Medvidović, M., Moreno, J.R., Eur. Phys. J. Plus **139**, 631

Neural quantum states

- Want to *learn* the function, f, by using a neural network

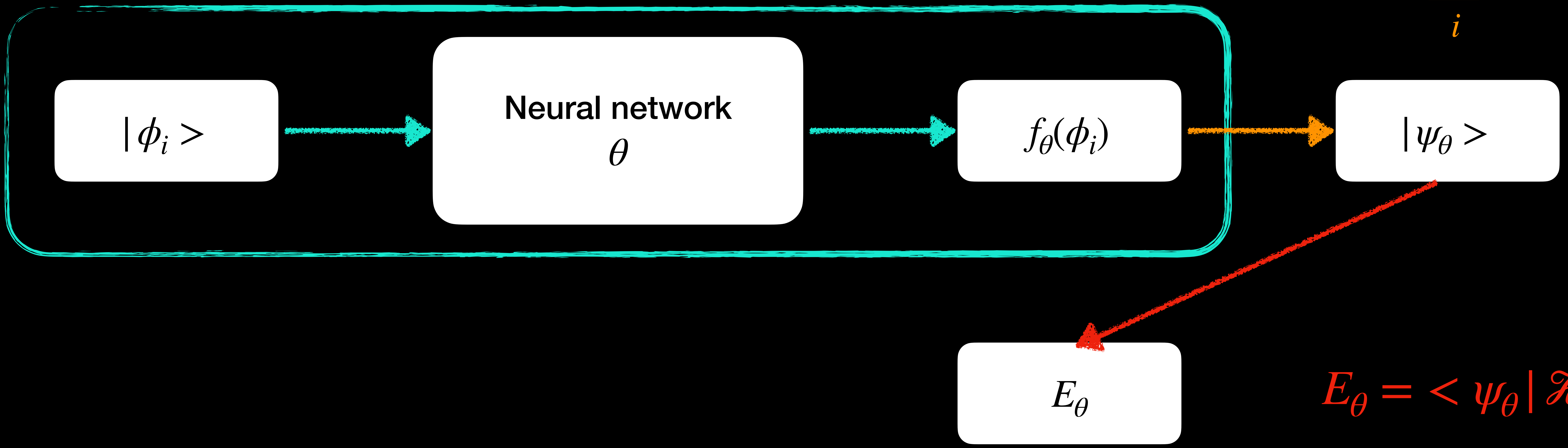
$$\psi_\theta = \sum_i f_\theta(\phi_i) \,|\, \phi_i >$$

Carleo, G., Troyer, M., Science **355**, 602
(Nice review) Medvidović, M., Moreno, J.R., Eur. Phys. J. Plus **139**, 631

Neural quantum states

- Want to *learn* the function, f, by using a neural network

$$\psi_\theta = \sum_i f_\theta(\phi_i)\,|\,\phi_i >$$



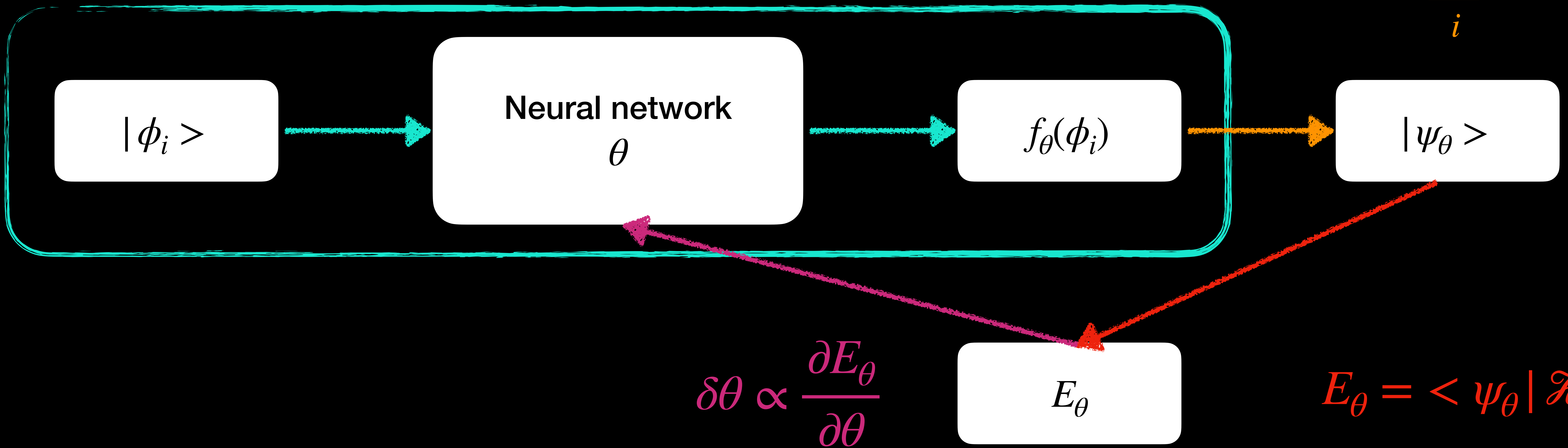
$$E_\theta = <\psi_\theta\,|\,\mathscr{H}\,|\,\psi_\theta>$$

Carleo, G., Troyer, M., Science **355**, 602
(Nice review) Medvidović, M., Moreno, J.R., Eur. Phys. J. Plus **139**, 631

Neural quantum states

- Want to *learn* the function, f, by using a neural network

$$\psi_\theta = \sum_i f_\theta(\phi_i) \,|\, \phi_i >$$

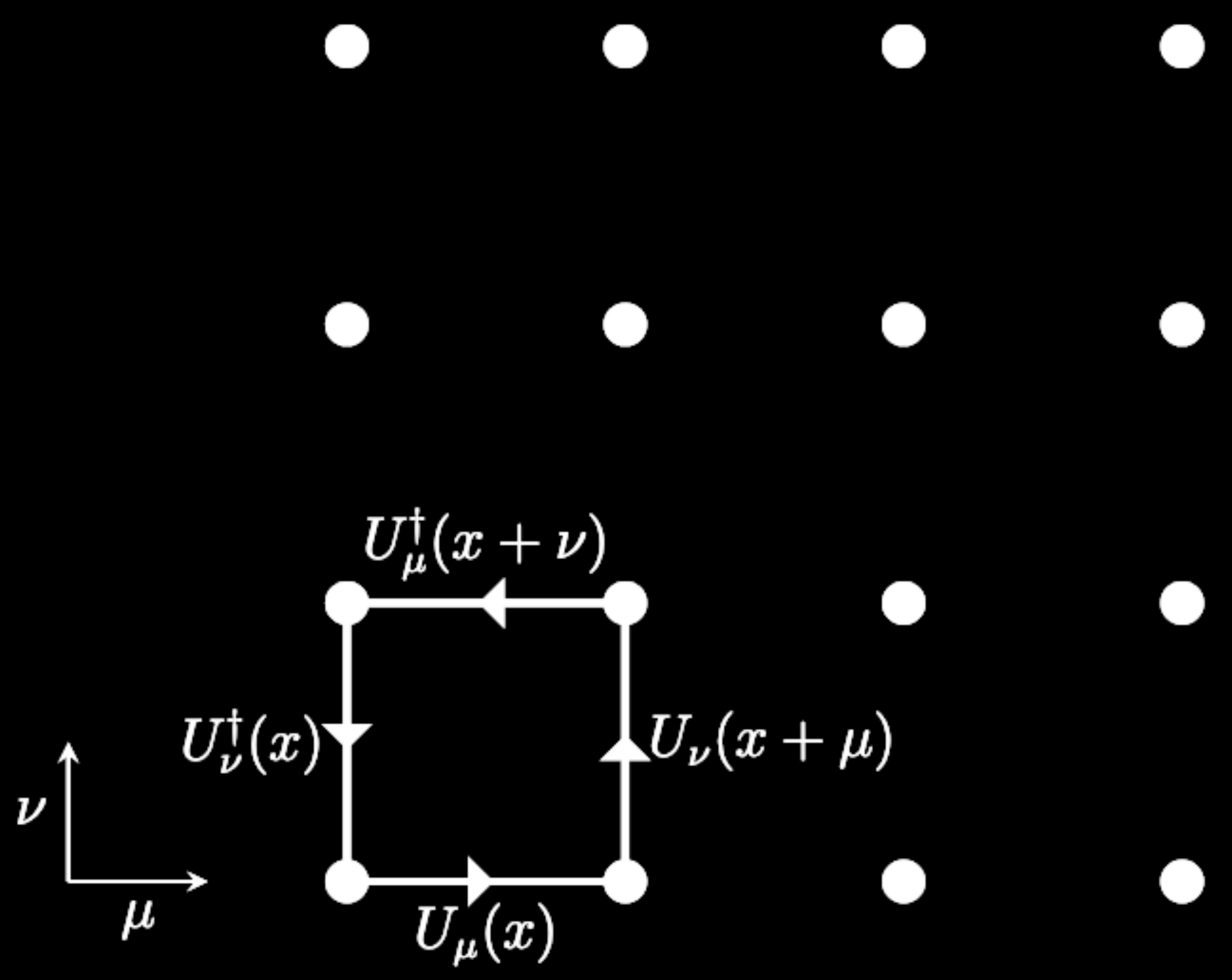


$$\delta\theta \propto \frac{\partial E_\theta}{\partial \theta}$$

$$E_\theta = <\psi_\theta \,|\, \mathscr{H} \,|\, \psi_\theta>$$

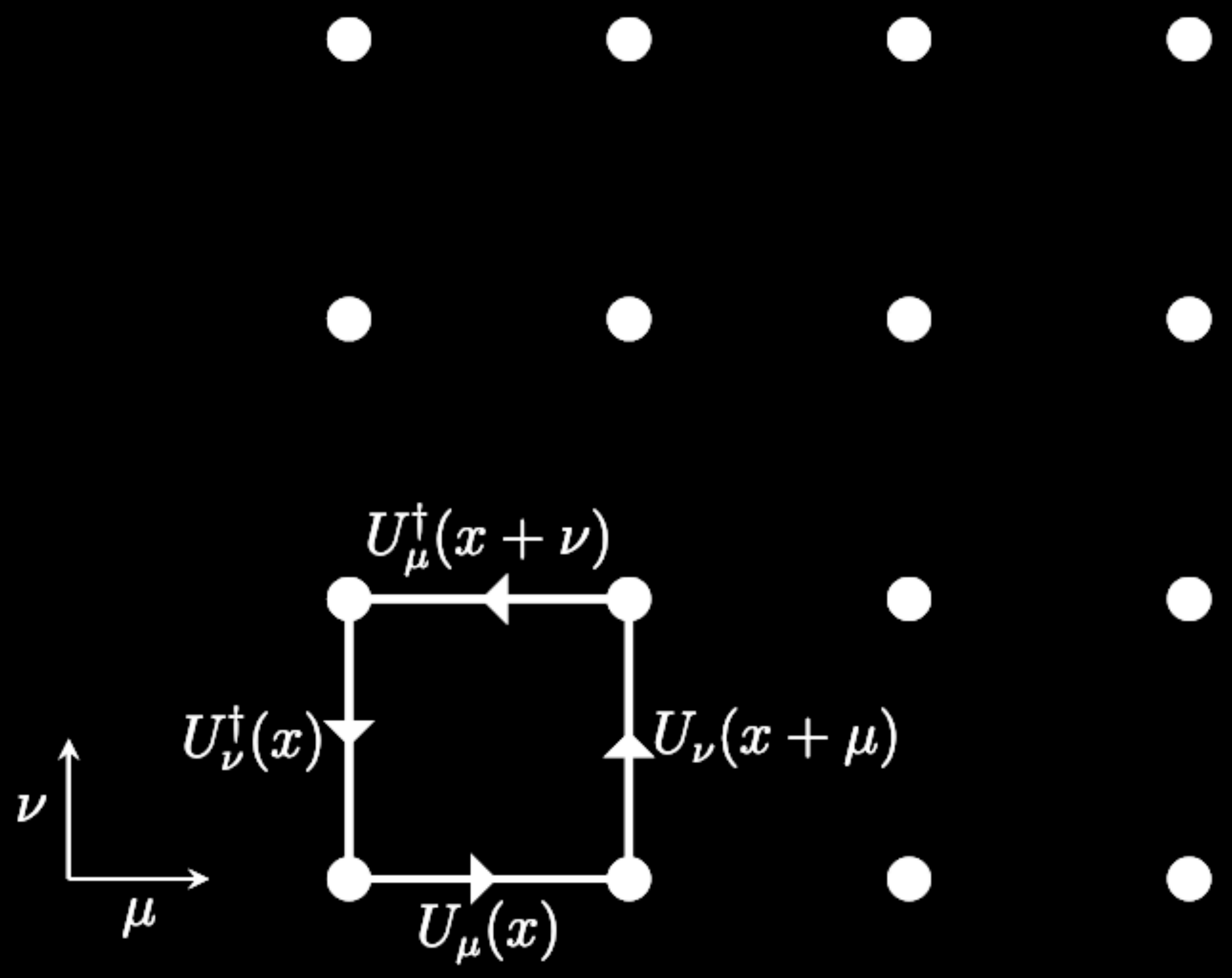
Carleo, G., Troyer, M., Science **355**, 602
(Nice review) Medvidović, M., Moreno, J.R., Eur. Phys. J. Plus **139**, 631

SU(2) Yang-Mills in mostly equations

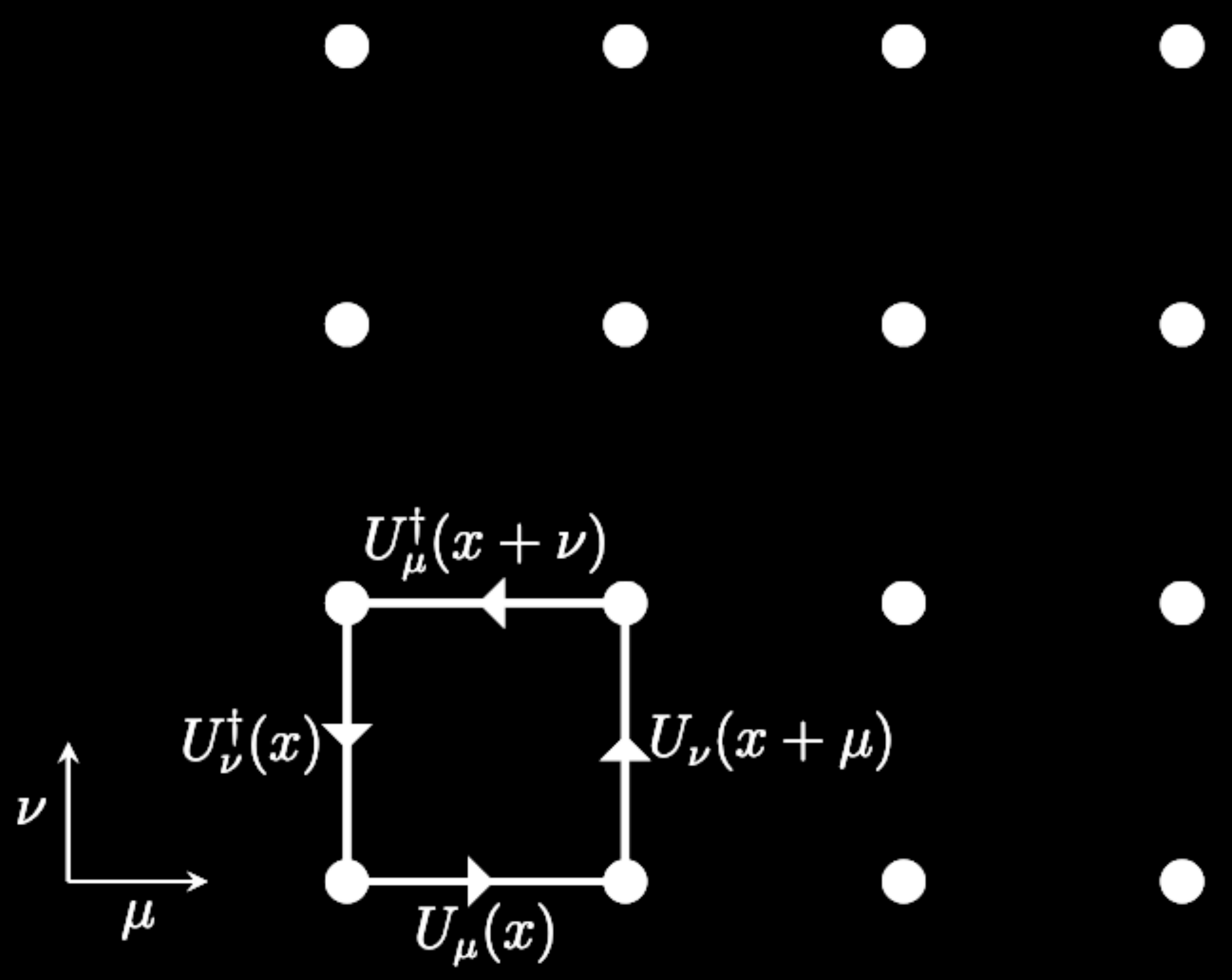


Chin, S.A. *et al.* Phys. Rev. D. **31**, 3201
Kogut, J., Susskind, L. Phys. Rev. D **11**, 395

$U_\mu(x) \in SU(2)$



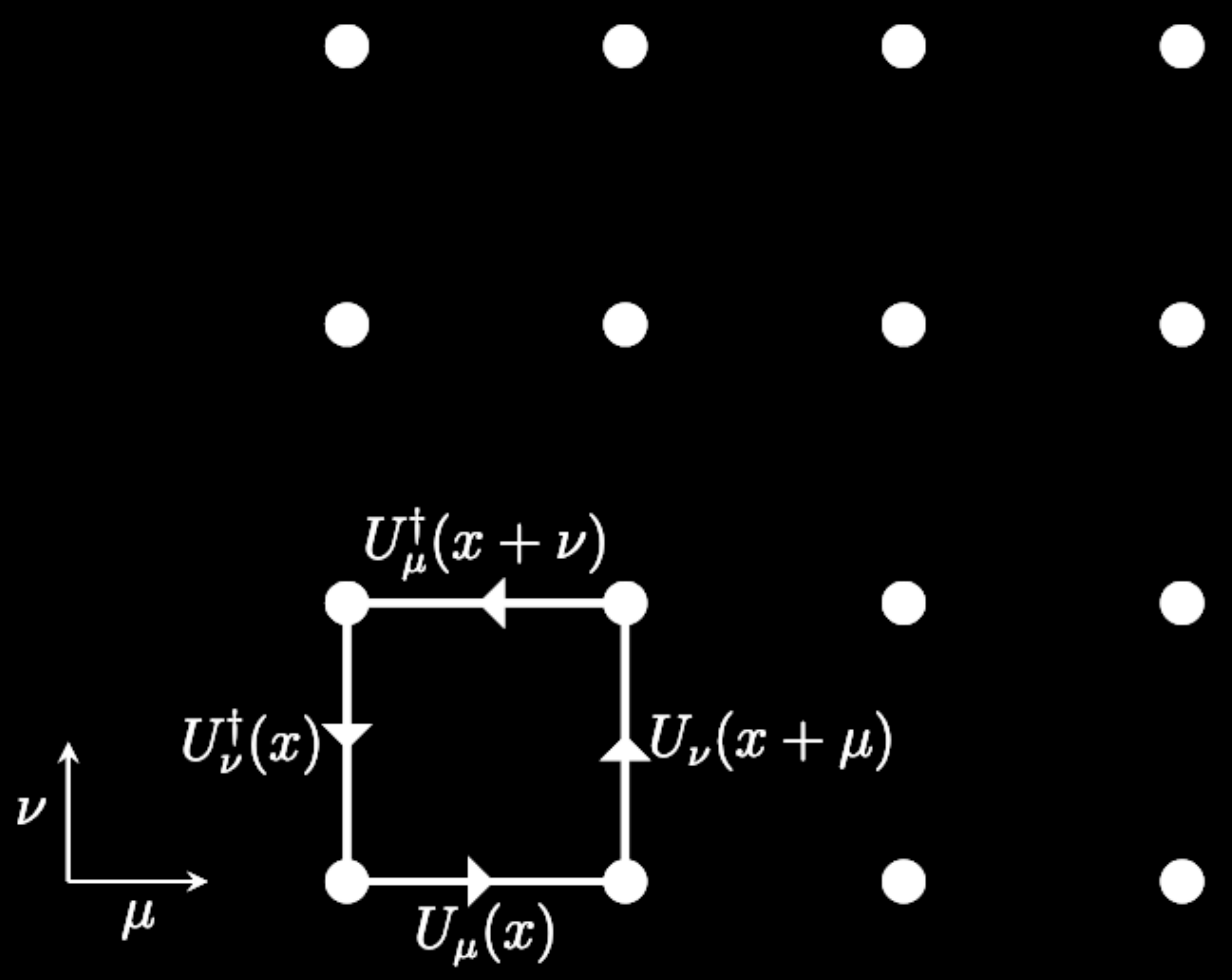
Chin, S.A. *et al.* Phys. Rev. D. **31**, 3201
Kogut, J., Susskind, L. Phys. Rev. D **11**, 395



$$U_\mu(x) \in SU(2)$$

$$U_\mu(x) = \exp(-i\sigma_a A^a_\mu(x))$$

Chin, S.A. *et al.* Phys. Rev. D. **31**, 3201
Kogut, J., Susskind, L. Phys. Rev. D **11**, 395

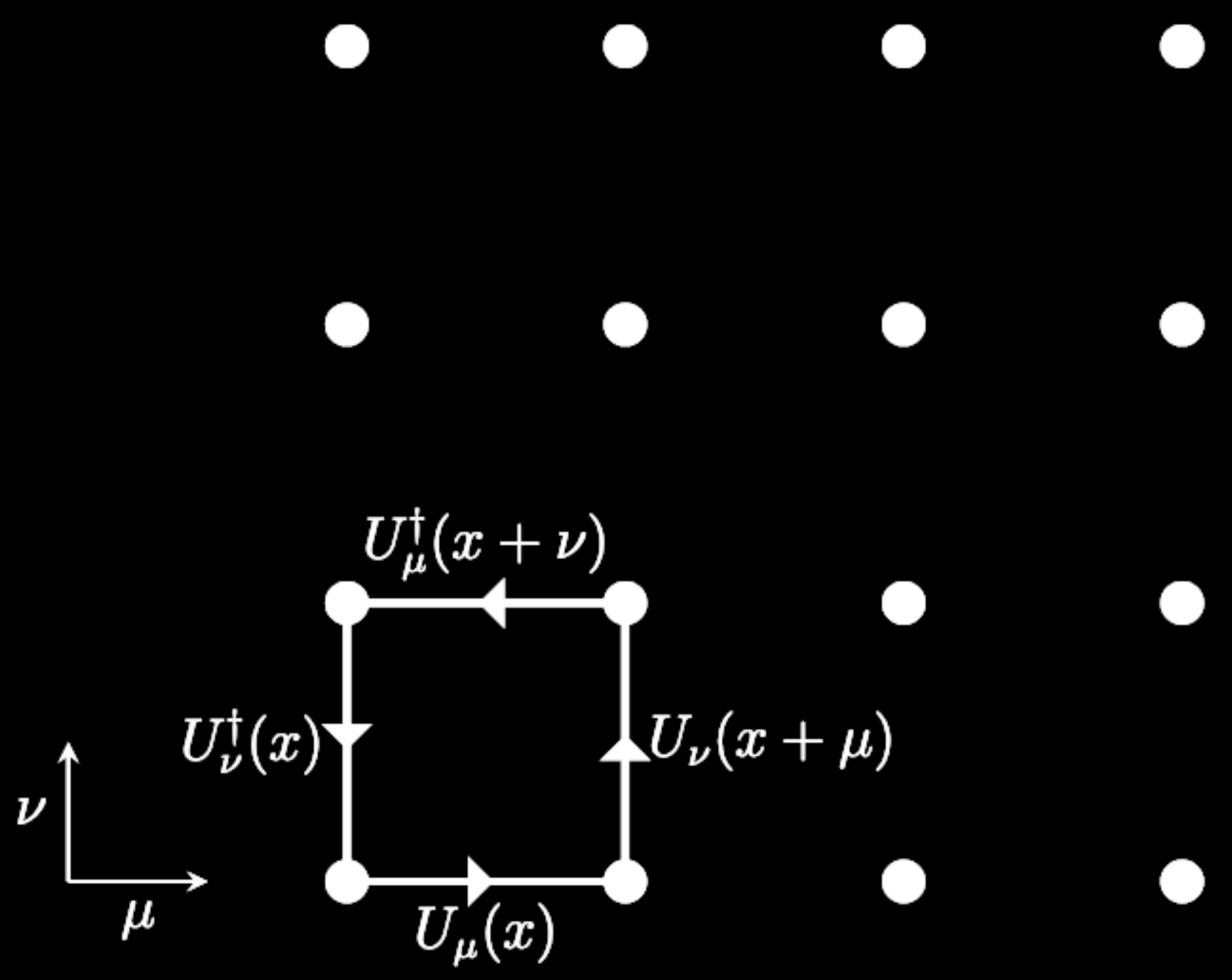


$$U_\mu(x) \in SU(2)$$

$$U_\mu(x) = \exp(-i\sigma_a A_\mu^a(x))$$

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

Chin, S.A. *et al.* Phys. Rev. D. **31**, 3201
Kogut, J., Susskind, L. Phys. Rev. D **11**, 395



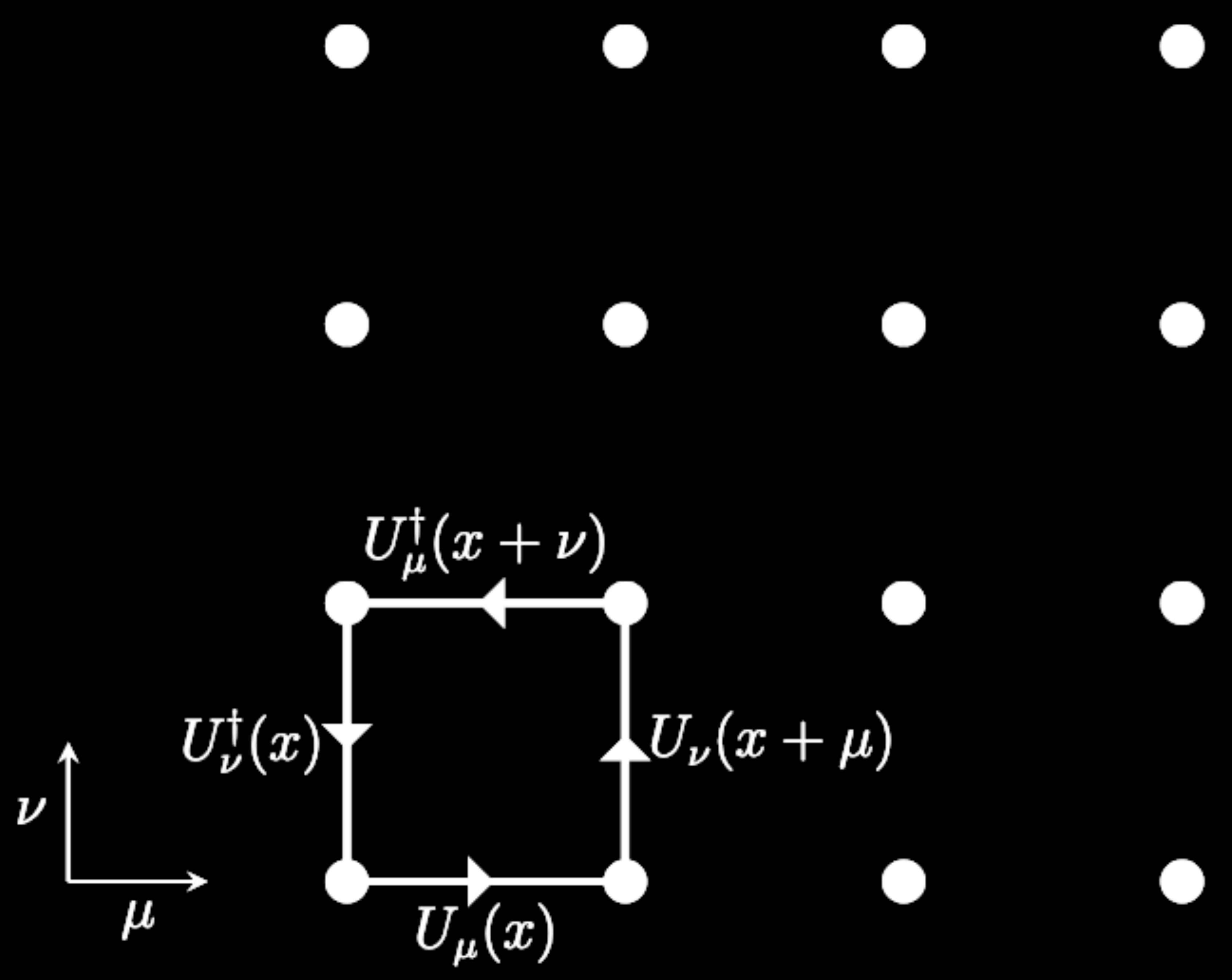
$$U_\mu(x) \in SU(2)$$

$$U_\mu(x) = \exp(-i\sigma_a A_\mu^a(x))$$

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

$$\mathscr{H} = -\frac{1}{2}\sum_l \nabla_l^2 + \lambda \sum_p (1 - \frac{1}{2}\mathrm{ReTr}P_p)$$

Chin, S.A. *et al.* Phys. Rev. D. **31**, 3201
Kogut, J., Susskind, L. Phys. Rev. D **11**, 395



$$U_\mu(x) \in SU(2)$$

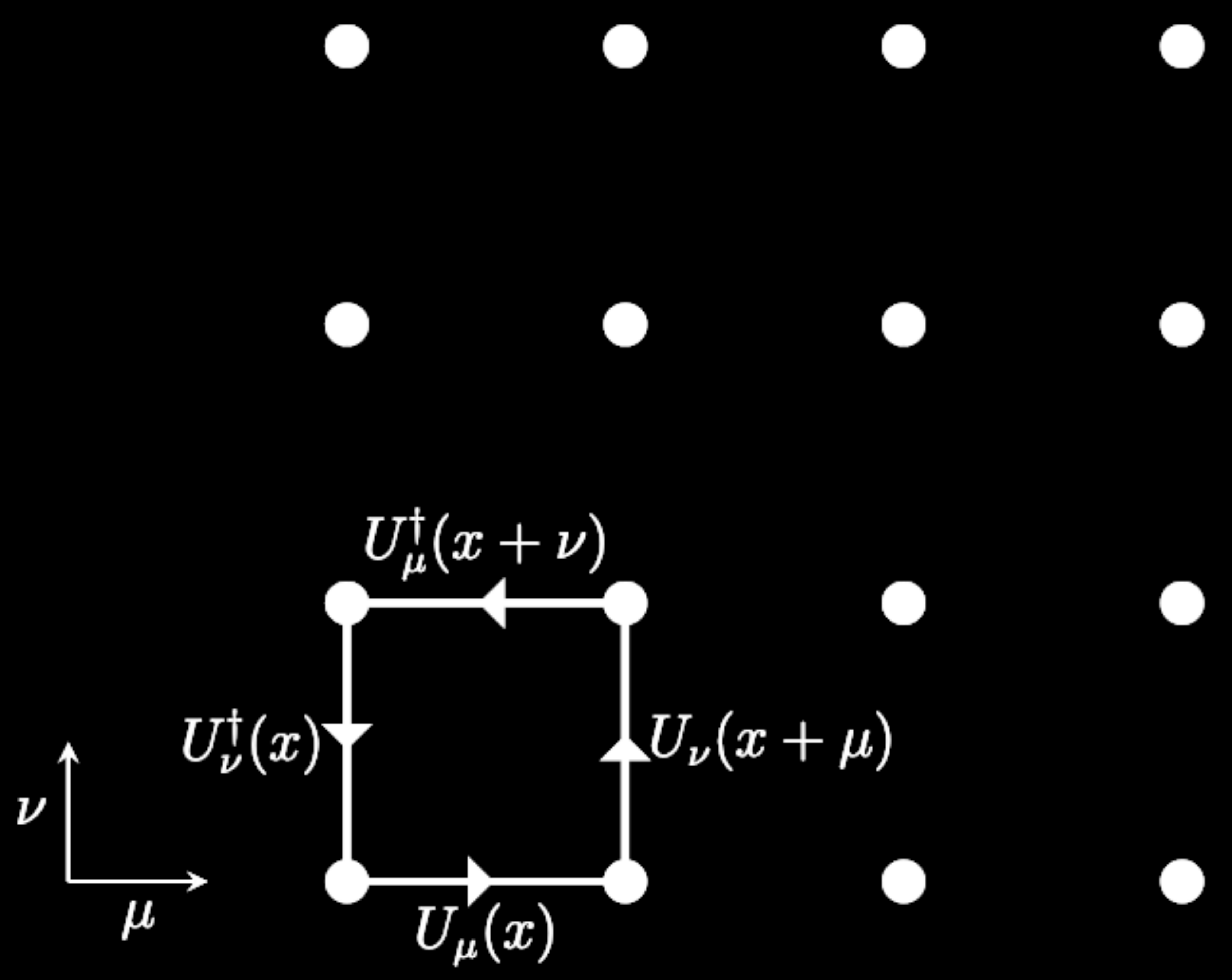
$$U_\mu(x) = \exp(-i\sigma_a A_\mu^a(x))$$

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

$$\mathscr{H} = -\frac{1}{2}\sum_l \nabla_l^2 + \lambda \sum_p (1 - \frac{1}{2}\mathrm{ReTr}P_p)$$

$$\nabla \sim \frac{\partial}{\partial A_\mu}$$

Chin, S.A. *et al.* Phys. Rev. D. **31**, 3201
Kogut, J., Susskind, L. Phys. Rev. D **11**, 395



$$U_\mu(x) \in SU(2)$$

$$U_\mu(x) = \exp(-i\sigma_a A_\mu^a(x))$$

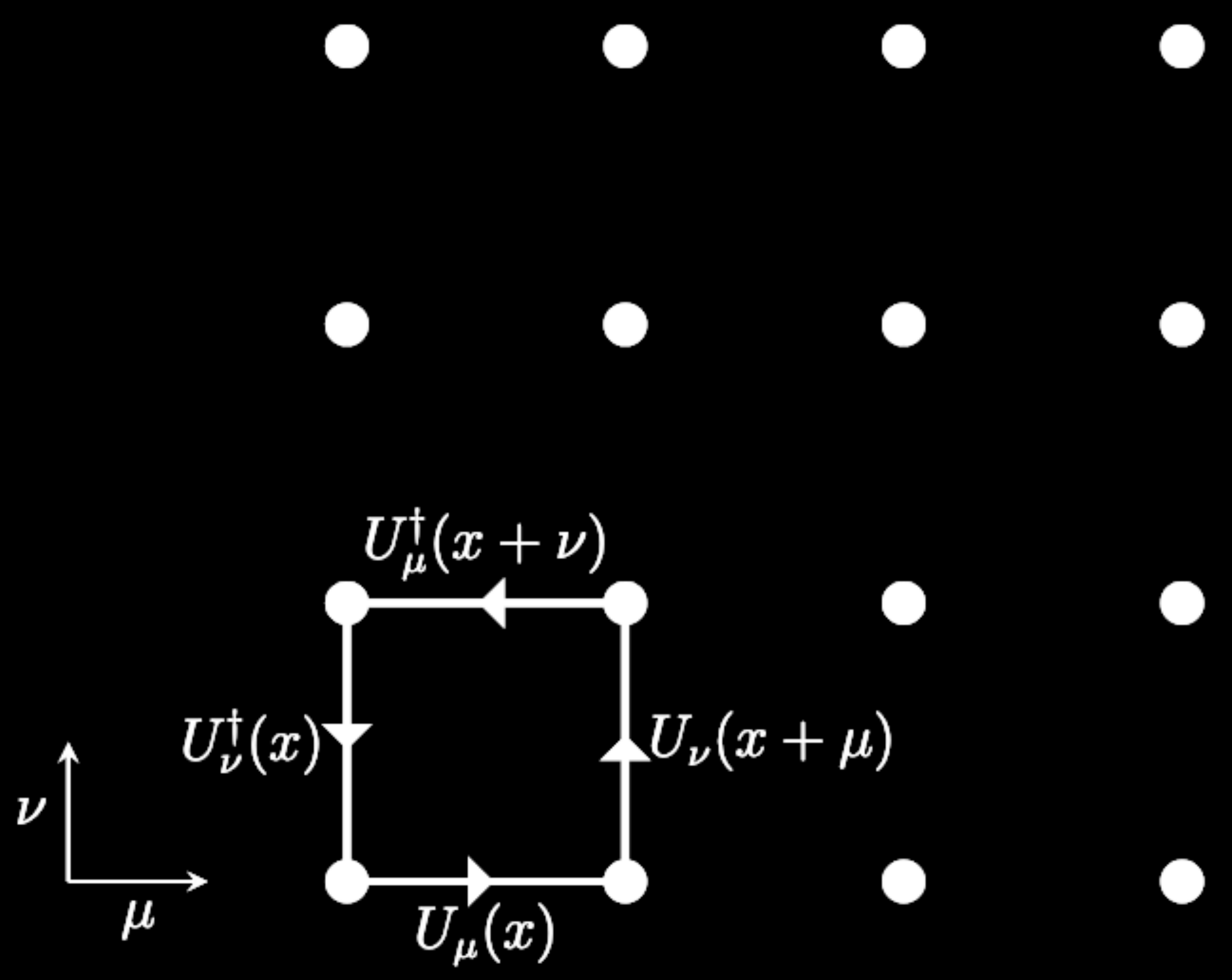
$$P_{\mu,\nu}(x) = U_\mu(x)U_\nu(x+\mu)U_\mu^\dagger(x+\nu)U_\nu^\dagger(x)$$

$$\mathscr{H} = -\frac{1}{2}\sum_l \nabla_l^2 + \lambda \sum_p (1 - \frac{1}{2}\mathrm{ReTr}P_p)$$

$$\nabla \sim \frac{\partial}{\partial A_\mu}$$

Laplace-Beltrami operator on S_3

Chin, S.A. *et al.* Phys. Rev. D. **31**, 3201
Kogut, J., Susskind, L. Phys. Rev. D **11**, 395



$\mathbf{U}$ = the whole configuration of $U_\mu(x)$s

$$< \psi | \mathscr{H} | \psi > \approx \frac{1}{n} \sum_{p(\mathbf{U})} \mathscr{H}[\mathbf{U}]$$

If $p(\mathbf{U})$ is the ground state wavefunction, this overlap is the ground state energy

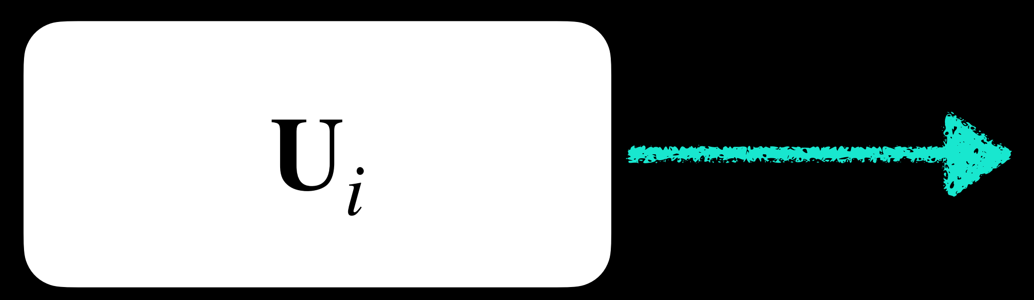
Neural quantum states for lattice gauge theories

Neural quantum states for lattice gauge theories

- Want to *learn* the function, f, by using a neural network

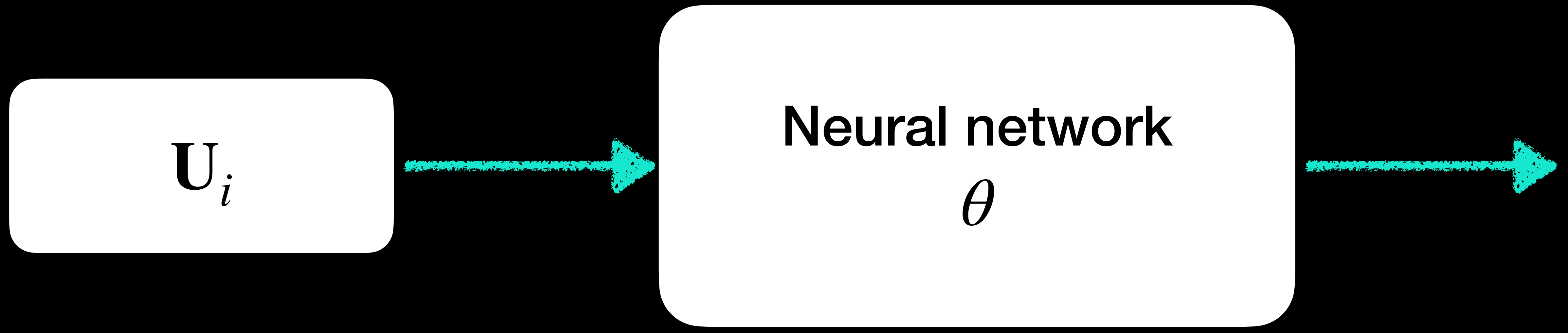
Neural quantum states for lattice gauge theories

- Want to *learn* the function, f, by using a neural network



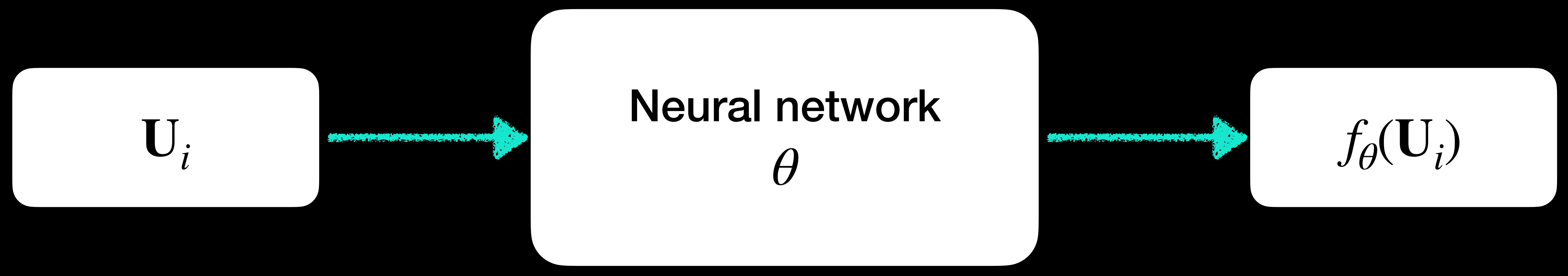
Neural quantum states for lattice gauge theories

- Want to *learn* the function, f, by using a neural network



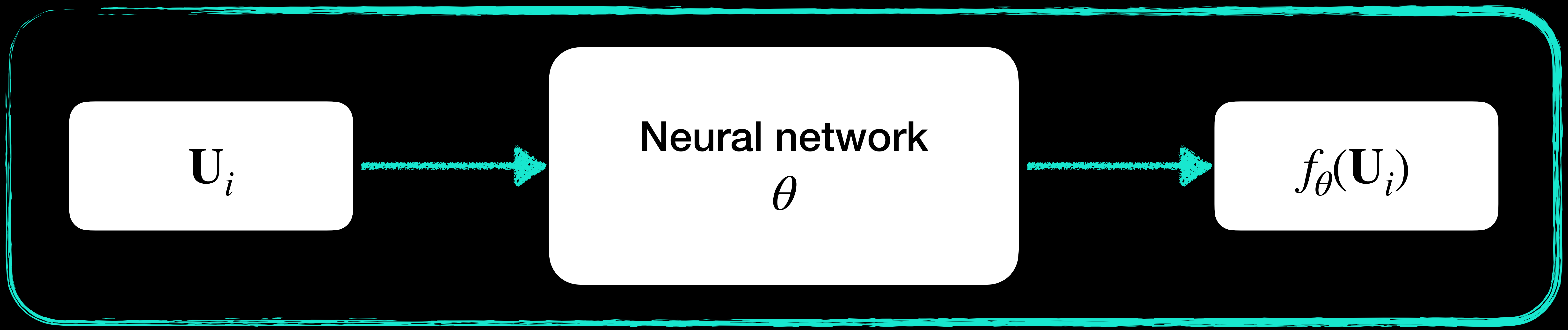
Neural quantum states for lattice gauge theories

- Want to *learn* the function, f, by using a neural network



Neural quantum states for lattice gauge theories

- Want to *learn* the function, f, by using a neural network



Neural quantum states for lattice gauge theories

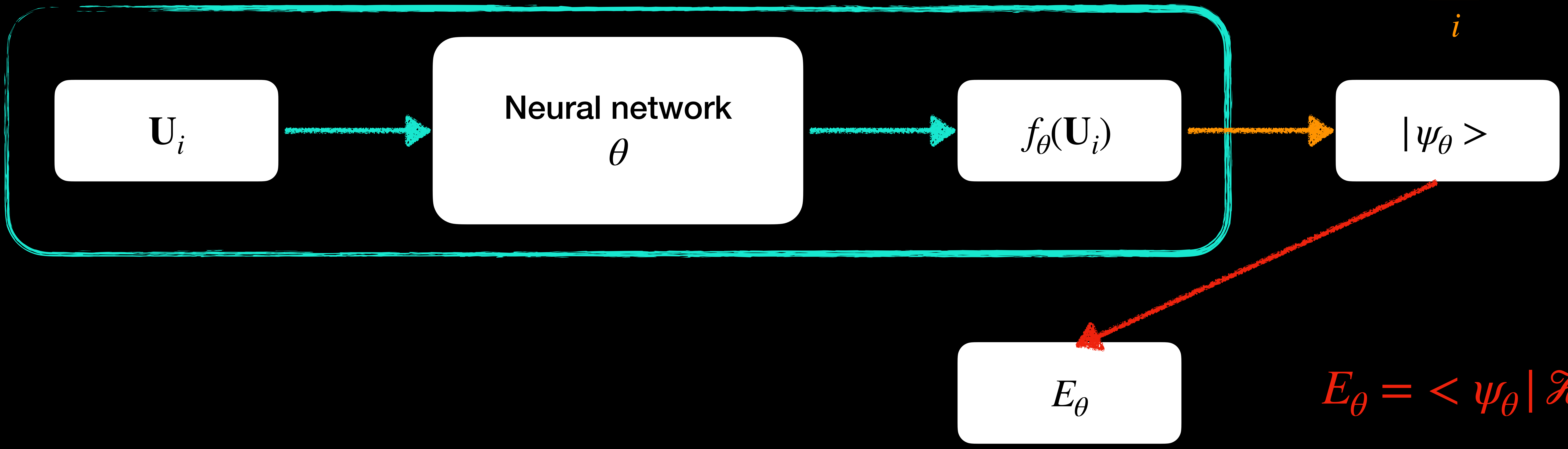
- Want to *learn* the function, f, by using a neural network

$$\psi_\theta = \sum_i f_\theta(\mathbf{U}_i)\,|\,\mathbf{U}_i >$$

Neural quantum states for lattice gauge theories

- Want to *learn* the function, f, by using a neural network

$$\psi_\theta = \sum_i f_\theta(\mathbf{U}_i)\,|\,\mathbf{U}_i >$$

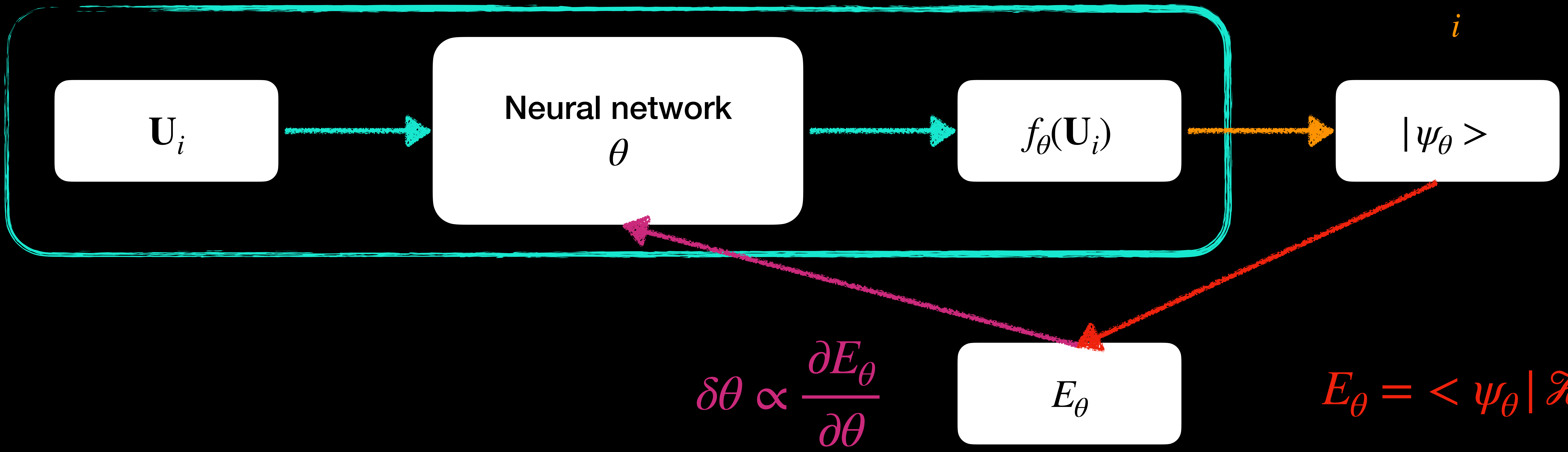


$$E_\theta = <\psi_\theta\,|\,\mathscr{H}\,|\,\psi_\theta>$$

Neural quantum states for lattice gauge theories

- Want to *learn* the function, f, by using a neural network

$$\psi_\theta = \sum_i f_\theta(\mathbf{U}_i)\,|\,\mathbf{U}_i >$$



$$\delta\theta \propto \frac{\partial E_\theta}{\partial \theta}$$

$$E_\theta = <\psi_\theta\,|\,\mathscr{H}\,|\,\psi_\theta>$$

The Ansätze

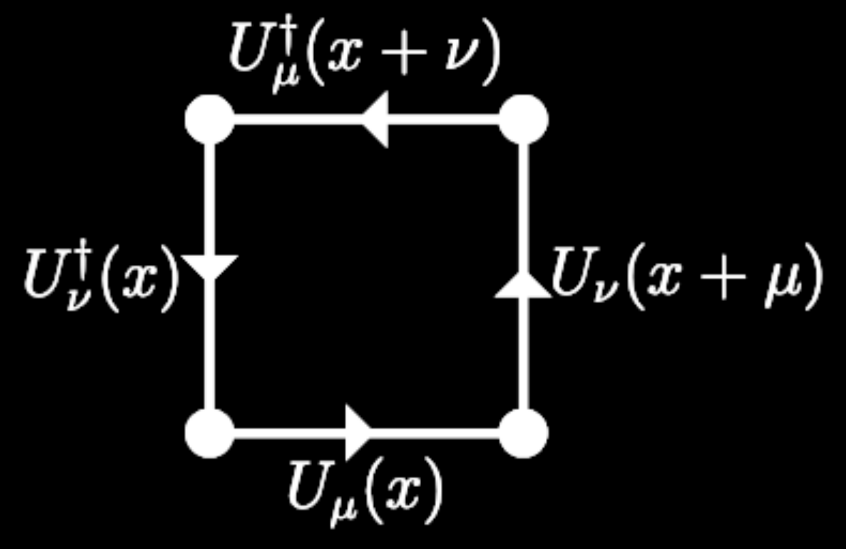
($f_\theta(\mathbf{U})$ from the other slides)

Mean-field plaquette Ansatz

(Simple yet good model)

- No neural network, but we guess our wavefunction coefficients through

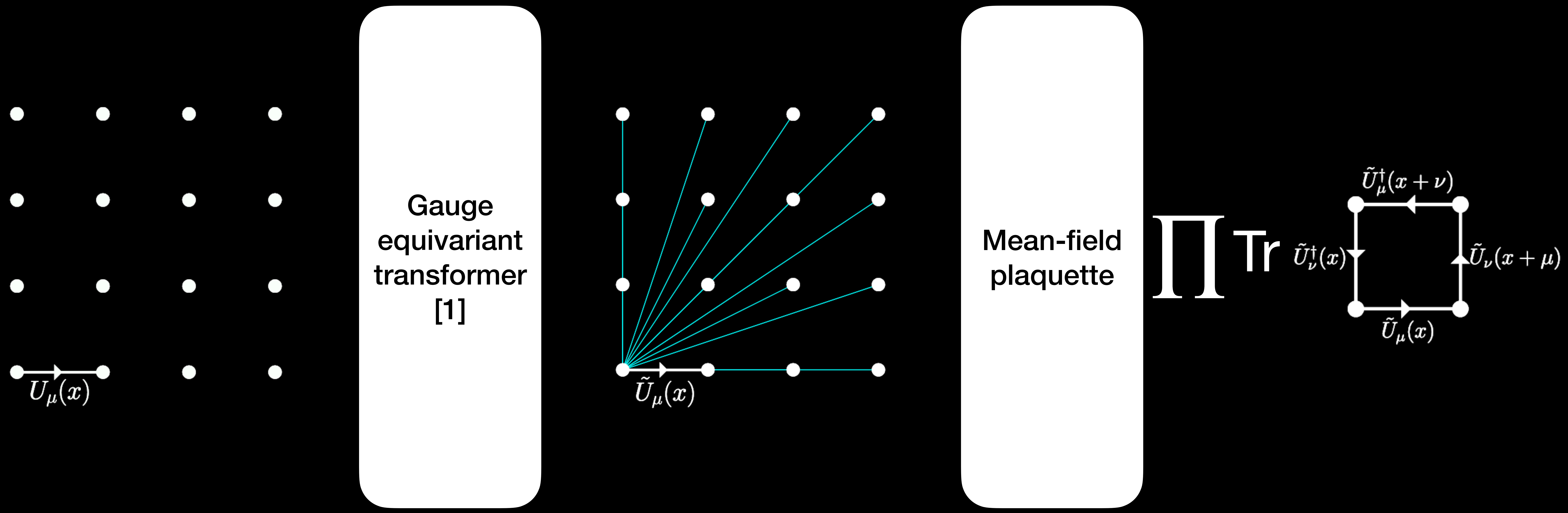
$$f(\mathbf{U}) = \prod_p e^{\alpha \frac{1}{2}\mathrm{ReTr} P_p} \qquad \prod \mathrm{Tr}$$



= the product of exponential of traced plaquettes

- Notably, this is gauge invariant (the wavefunction does not change under a gauge transformation of $\mathbf{U}$)

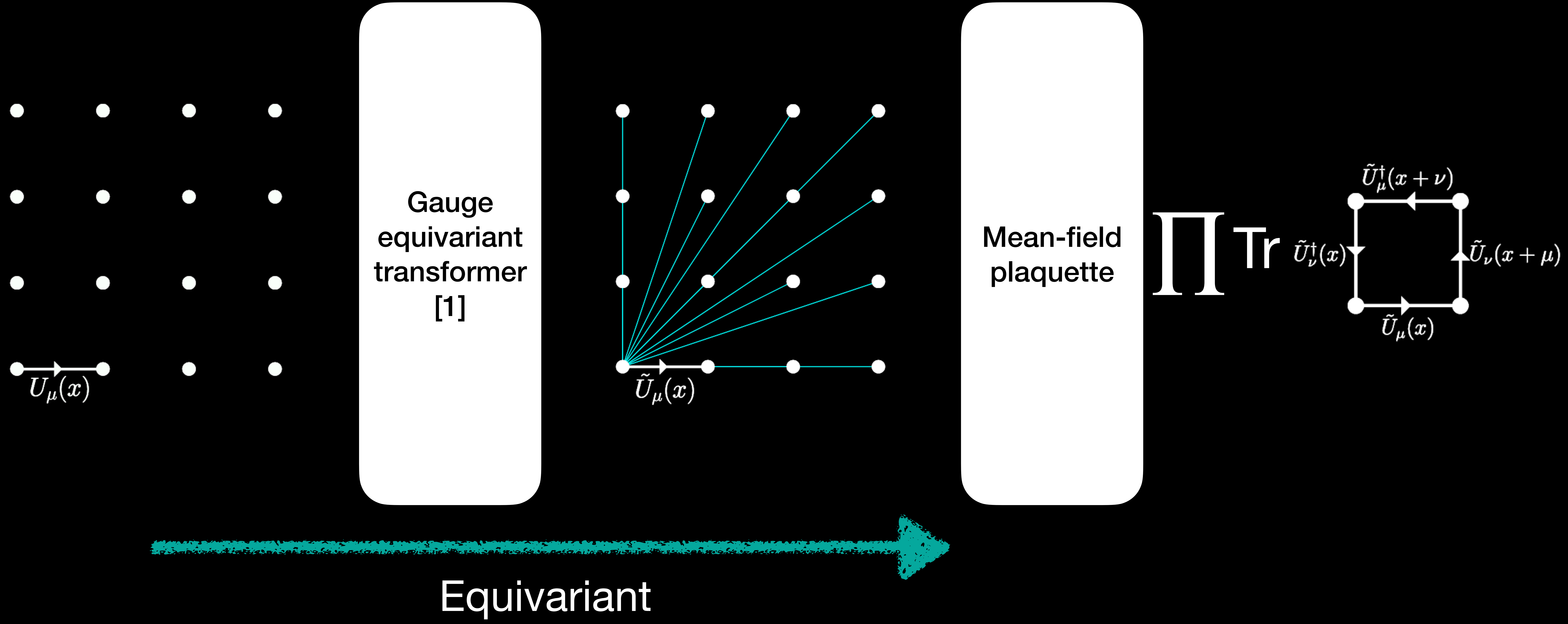
Gauge equivariant transformer
(Our main contribution)



[1] Transformer adaptation of Kanwar, G. *et al.* Phys. Rev. Lett. **125**, 121601 & Boyda, D. L. *et* al. Phys. Rev. D **103**, 074504

Gauge equivariant transformer

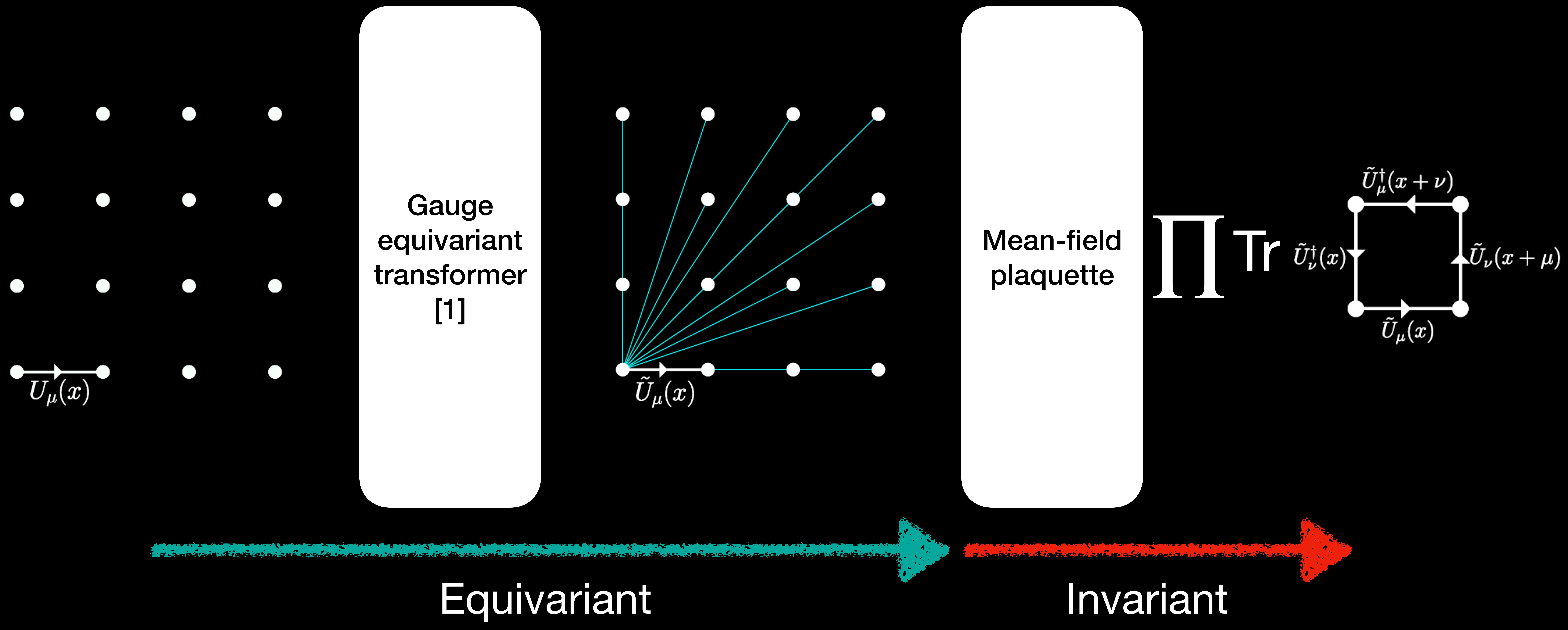
(Our main contribution)



[1] Transformer adaptation of Kanwar, G. *et al.* Phys. Rev. Lett. **125**, 121601 & Boyda, D. L. *et* al. Phys. Rev. D **103**, 074504

Gauge equivariant transformer

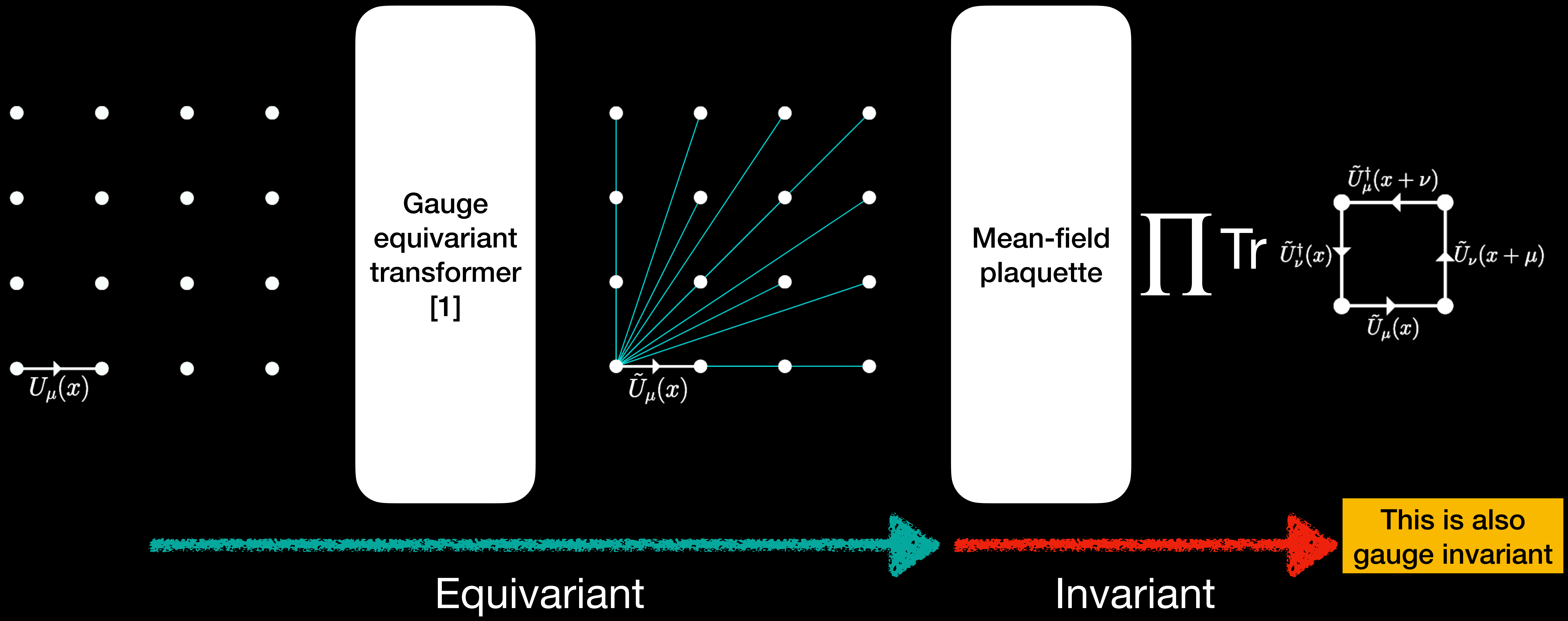
(Our main contribution)



[1] Transformer adaptation of Kanwar, G. *et al.* Phys. Rev. Lett. **125**, 121601 & Boyda, D. L. *et* al. Phys. Rev. D **103**, 074504

Gauge equivariant transformer

(Our main contribution)



[1] Transformer adaptation of Kanwar, G. *et al.* Phys. Rev. Lett. **125**, 121601 & Boyda, D. L. *et* al. Phys. Rev. D **103**, 074504

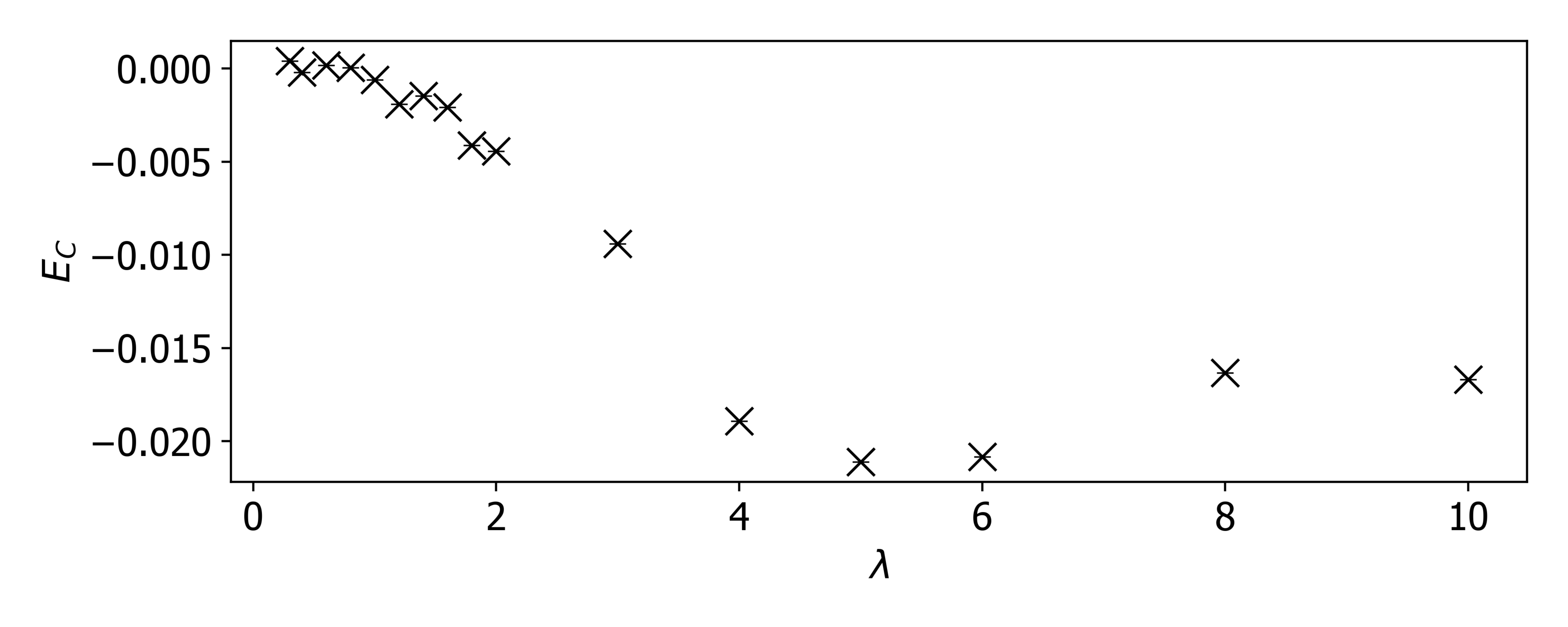
Results

$$\mathscr{H} = -\frac{1}{2}\sum_l \nabla_l^2 + \lambda \sum_p (1 - \frac{1}{2}\mathrm{ReTr}P_p)$$

Ground state energy

- 4x4x4
- Correlation energy:

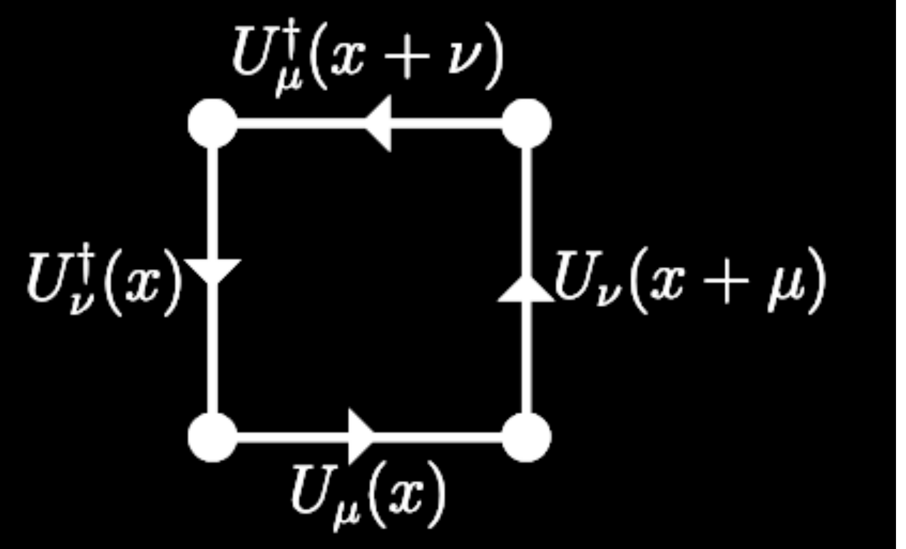
$$E_C = \frac{E_{\mathrm{equivariant}} - E_{\mathrm{MF}}}{E_{\mathrm{MF}}}$$

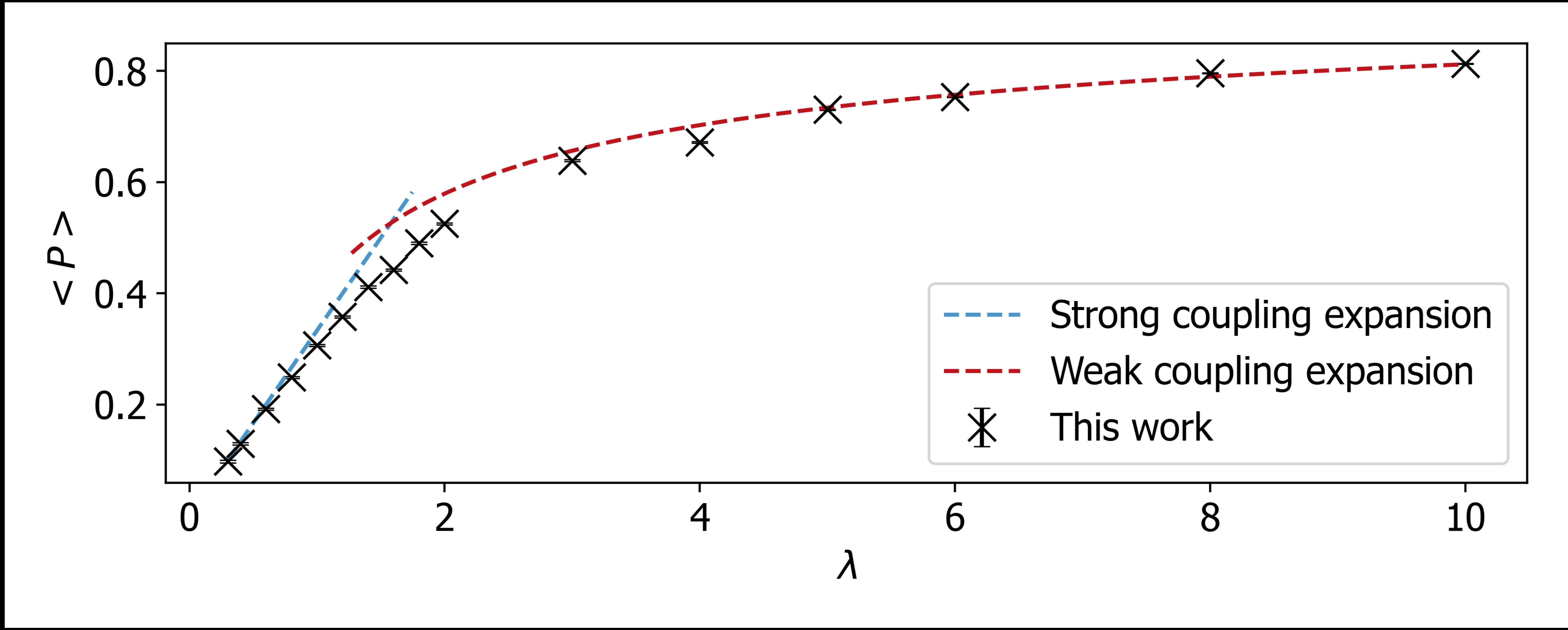


Plaquette expectation

$$< P > = \frac{1}{2}\mathrm{ReTr}[P_{\mu,\nu}(x)]$$

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





What now?

Future outlook

- Large 2D simulations
- Phase transitions through non-trivial Wilson loops
    - Finite size effects
- More complicated observables like the Creutz ratio
- Time evolution?

Thanks for listening

Backup slides

Equivariant layer

- We want an ‘image to image’ like function that acts on a lattice of links and outputs a lattice of transformed links
- We want this to be gauge equivariant, meaning
  - If the action of a gauge transformation is given by $g(U)$ and the equivariant layer is given by $f(U)$, then we require

$$f(g(U)) = g(f(U))$$

Equivariant layer

- We cannot simply act on a link alone, it doesn't have the right transformation properties for the equivariant relation to hold
- But, the plaquette does
  - Notably, the eigenvalues of the plaquette are invariant under gauge transformations
  - The eigenvalues of $P_{\mu,\nu}(x)$ are the same as those of $\Omega(x)P_{\mu,\nu}(x)\Omega^\dagger(x)$ [which is $g(P_{\mu,\nu}(x))$]
  - We exploit this to make our network gauge equivariant

Equivariant layer

$U'_\mu(x) = P'_{\mu,\nu}(x) * (U_\nu(x+\mu)U^\dagger_\mu(x+\nu)U^\dagger_\nu(x))^{-1}$

- So, within our layer, we do:

$$U_\mu(x) \rightarrow P_{\mu,\nu}(x) \rightarrow \lambda_P \rightarrow \lambda'_P \rightarrow P'_{\mu,\nu}(x) \rightarrow U'_\mu(x)$$

Eigenvalue recomposition

Transformer

Eigenvalue decomposition

$P_{\mu,\nu}(x) = U_\mu(x)U_\nu(x+\mu)U^\dagger_\mu(x+\nu)U^\dagger_\nu(x)$