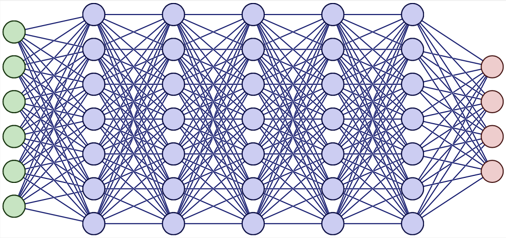
Uses of ML for Lattice Field Theories



L Del Debbio

Higgs Centre for Theoretical Physics
University of Edinburgh

work in collaboration with

D Albandea, P Hernandez, R Kenway, J Marsh-Rossney, A Ramos, M Wilson

seminal work by *MIT group*

and more work by *Bacchio et al*, and *Lehner & Wettig*

for details, see papers [ldd et al 21], [albandea et al 23], [albergo et al 19], [bacchio et al 22], [lehner & wettig 23]

focus on two applications

- sampling of field configurations
- inversion of the Dirac operator
- crucial ingredients for simulations of LFT on exascale machines?
- significant progress in the last 4 years
- understand the ingredients and the recipe

Monte Carlo sampling

$$\langle O \rangle = \frac{1}{Z} \int \mathcal{D}\phi \, e^{-S(\phi)} \, O(\phi)$$

$$\hookrightarrow \left\{ \phi^{(1)}, \ldots, \phi^{(N)} \right\} \sim p(\phi) = \frac{1}{Z} e^{-S(\phi)} \quad \text{(physical/target distribution)}$$

estimator for $\langle O \rangle$

$$\bar{O} = \frac{1}{N} \sum_{n=1}^{N} O\left(\phi^{(n)}\right)$$

$$\mathsf{Var}[\bar{O}] = \frac{2\tau_O}{N} \mathsf{Var}[O] \,, \quad \tau_O = \frac{1}{2} + \sum_t \frac{\Gamma_O(t)}{\Gamma_O(0)}$$

$$\Gamma_O(t) = \langle O^{(n+t)} O^{(n)} \rangle - \langle O \rangle^2$$

critical slowing down

$$\tau_O \sim \hat{\xi}^{z_O} \,, \quad \tau_{\text{top}} \sim \exp\left(c\xi^\theta\right)$$

trivializing flows

change of variables in the path integral [luscher 09]

$$\phi = \mathcal{F}(\tilde{\phi}) \quad \Longrightarrow \mathcal{F}^*(\tilde{\phi})_{xy} = \frac{\partial \phi_x}{\partial \tilde{\phi}_y}$$

$$\mathcal{D}\phi = \mathcal{D}\tilde{\phi}\, \det \mathcal{F}^*(\tilde{\phi})$$

yields

$$Z = \int \mathcal{D}\phi\, e^{-S(\phi)} = \int \mathcal{D}\tilde{\phi}\, e^{-S_{\mathcal{F}}(\tilde{\phi})}$$

$$S_{\mathcal{F}}(\tilde{\phi}) = S\left(\mathcal{F}(\tilde{\phi})\right) - \log\det \mathcal{F}^*(\tilde{\phi})$$

$S_{\mathcal{F}} =$ const $\quad \Longrightarrow \quad$ trivial theory!

normalizing flows à la MIT

generative model using latent variables [albergo et al 19]

$$\phi = f_\theta(z)\,, \quad z \sim r(z) \quad \text{(easy/latent distribution)}$$

$$\Longrightarrow \phi \sim p_\theta(\phi) = r\left(f_\theta^{-1}(\phi)\right) \left|\det f_\theta^*\right|^{-1} \quad \text{(model distribution)}$$

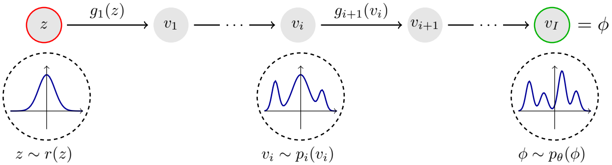
model f_θ *using Neural Networks*, find *best* transformation

$$\bar{\theta} = \arg\max_\theta D_{\mathrm{KL}}\left(p_\theta | p\right)$$

$$D_{\mathrm{KL}}\left(p_\theta | p\right) = \int \mathcal{D}\phi\, p_\theta(\phi)\, \log \frac{p_\theta(\phi)}{p_(\phi)} = \mathbb{E}_{z\sim r(z)}\left[S\left(f_\theta(z)\right) - \log\det f_\theta^*(z)\right]$$

use f_θ to generate candidate configurations + Metropolis accept/reject

expressivity vs usability: coupling layers



$$f_\theta = g_I \circ g_{I-1} \circ \ldots \circ g_1$$

$$v_{i+1} = g_i(v_i)$$

$$\hookrightarrow \quad v_{i+1,x} = \begin{cases} v_{i,x}\,, & x \in \Lambda_i^P \\ C_{i,x}\left(v_{i,x}; \mathrm{NN}_\theta(v_i^P)\right)\,, & x \in \Lambda_i^A \end{cases}$$

easy to invert and easy to compute the Jacobian

$$\log \left| \frac{\partial g_i}{\partial v_i} \right| = \sum_{x \in \Lambda_i^A} \log \left| \frac{\partial C_{i,x}}{\partial v_{i,x}} \right|$$

acceptance

Metropolis test/exact algorithm

$$A\left(\phi \to \phi'\right) = \min\left(1, \frac{q(\phi|\phi')}{q(\phi'|\phi)}\,\frac{p(\phi')}{p(\phi)}\right)$$

for normalizing flows

$$q(\phi'|\phi) = p_{\bar{\theta}}(\phi')$$

$$p_{\bar{\theta}}(\phi') = p(\phi') \quad \Longrightarrow \quad A\left(\phi \to \phi'\right) = 1$$

$$\tau_O \geq \frac{1}{\mathbb{E}_{\phi\sim p}\mathbb{E}_{\phi'\sim p_{\bar{\theta}}}[A\left(\phi \to \phi'\right)]} - \frac{1}{2}$$

rejection $\leftrightarrow$ correlation

scalar field theory

lattice action

$$S(\phi) = \sum_{x\in\Lambda} \left[-\beta \sum_{\mu} \phi_{x+\mu}\phi_x + \phi_x^2 + \lambda \left(\phi_x^2 - 1\right)^2 \right]$$

observables

$$M = \sum_x \phi_x$$

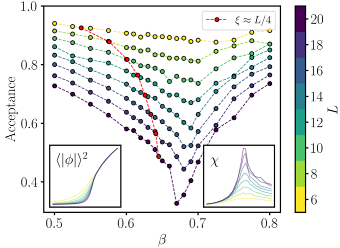
$$\chi = \frac{1}{|\Lambda|} \left\langle \left(M - \langle M \rangle\right)^2 \right\rangle$$

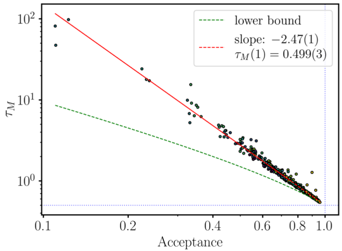
$$G(x) = \frac{1}{|\Lambda|} \sum_y \langle \phi_{y+x}\phi_y \rangle - \langle \phi \rangle^2$$

$$G(t) \sim \cosh\left(\frac{t - L/2}{\hat{\xi}}\right)$$

training efficiency

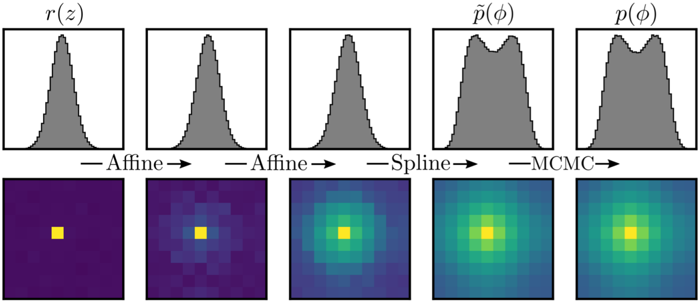
acceptance depends on parameters and the size of the system



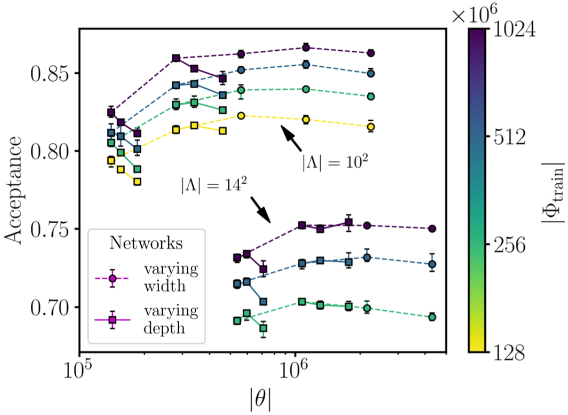


same training for all couplings: 16K iterations, 16K batch size

architecture



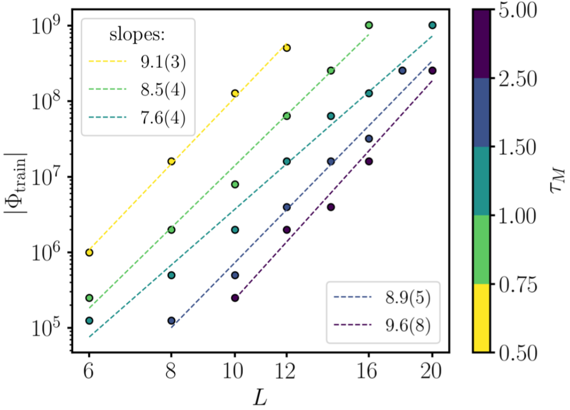
architecture



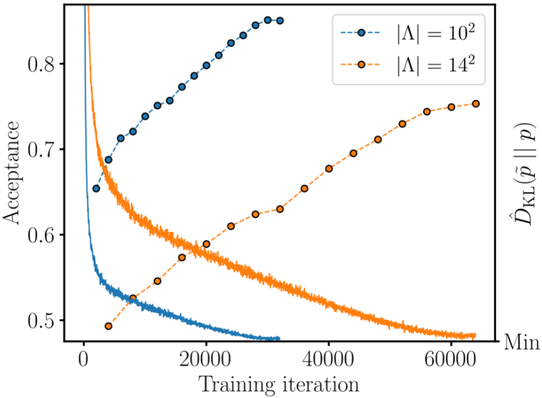
one affine block, one RQS block, 32k iterations

$$\Phi_{\mathrm{train}} = \text{batch size} \times \text{iterations}$$

cost scaling



diminishing returns



flow HMC

- use a normalizing flow to generate an approximate trivializing map

$$\mathcal{F} = f_{\bar{\theta}}^{-1}$$

- perform an HMC simulation in the 'flow' variables

$$\langle O \rangle = \frac{1}{Z} \int \mathcal{D}\tilde{\phi}\, e^{-S_{\mathcal{F}}(\tilde{\phi})}\, O\left(f_{\bar{\theta}}^{-1}(\tilde{\phi}) \right)$$

- Markov chain of $\tilde{\phi}$ configurations

$$\left\{ \tilde{\phi}_1, \ldots, \tilde{\phi}_N \right\} \sim e^{-S_{\mathcal{F}}}$$

- apply $f_{\bar{\theta}}^{-1}$ to obtain a Markov chain

$$\{\phi_1, \ldots, \phi_N\} \sim e^{-S}$$

affine/CNN layers

affine layer

$$g_i(\{\phi^P, \phi^A\}) = \{\phi^P, \phi^A \odot e^{s^{(i)}(\phi^P)} + t^{(i)}(\phi^P)\}$$

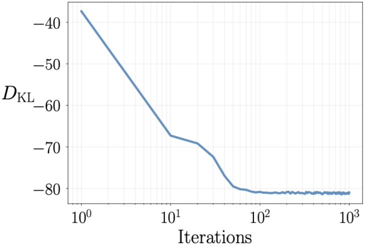
Jacobian matrix

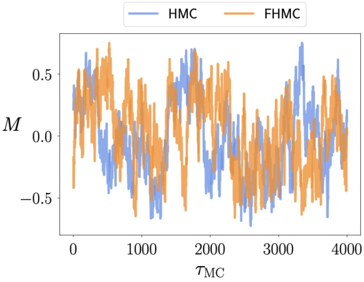
$$\left| \det \frac{\partial g_i(\phi)}{\partial \phi} \right| = \prod_{x_A} e^{s_x^{(i)}(\phi^P)}$$

CNN parametrization

$$s_x^{(i)}(\phi^P) = \tanh \left[\sum_{y \in \left[-\frac{k-1}{2}, \frac{k-1}{2}\right]^2} w^{(i)}(y) \phi^P_{x-y} \right]$$

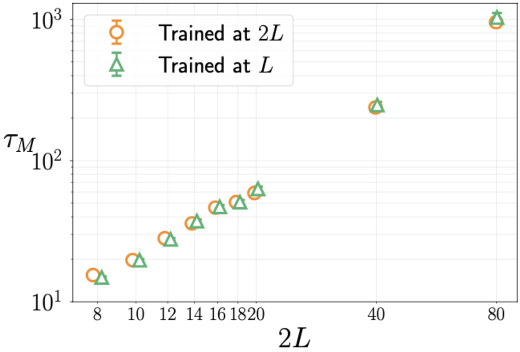
training





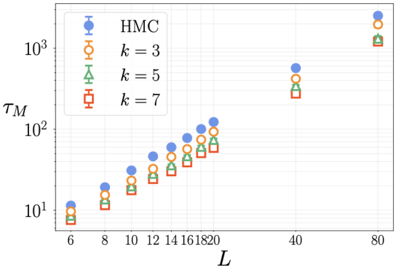
$$\tau_{M,\mathsf{FHMC}} = 74.4(3), \quad \tau_{M,\mathsf{HMC}} = 100.4(2).$$

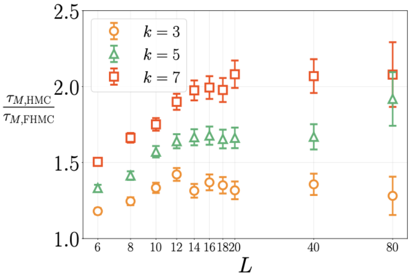
larger volumes



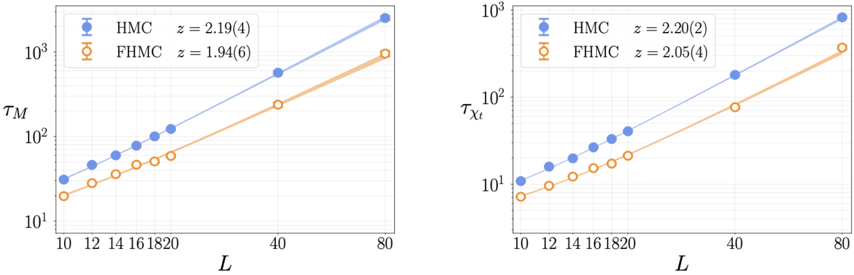
L	β	Acc. at L	Acc. at $2L$
3	0.537	0.3	0.2
4	0.576	0.04	0.001
5	0.601	0.002	0.00003
6	0.616	0.002	0.000007
7	0.626	0.0001	$< 10^{-7}$
8	0.634	0.0001	-
9	0.641	0.00007	-
10	0.645	0.00004	-

scaling at fixed architecture





scaling with $k \sim \xi$



learning gradient flows

build the map by integrating a flow eq in configuration space

$$\dot{U}_t = Z_t(U_t)U_t$$

where

$$\begin{aligned}
[Z^a(U_t)](x,\mu) &= -\partial^a_{x,\mu}\tilde{S}(U_t,t) \\
\tilde{S}(U_t,t) &= \sum_i c_i(t,\theta)\,\mathcal{W}_i(U_t) \\
\mathcal{F}_\theta(V) &= U_t
\end{aligned}$$

[bacchio et al 22]

learn parameters by gradient descent given a cost function

$$\mathcal{C}(\theta) = \langle S_{\mathcal{F}_\theta}(V)\rangle$$

$\hookrightarrow$ adjoint state method (Lagrange multiplier)

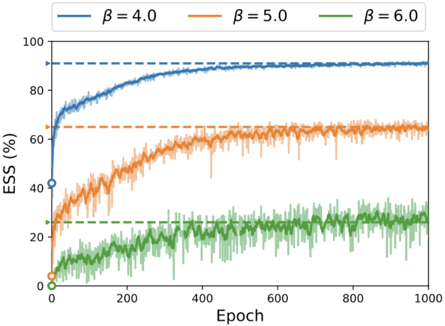
numerical results

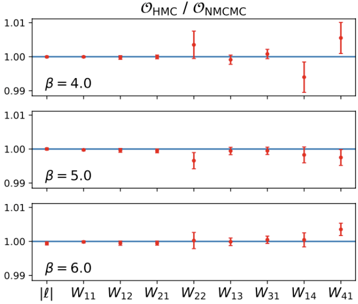
- model **A**: $2_t \times 7_W$
- model **B**: $10_t \times 42_W$
- quality of the model:

$$\text{ESS} = \frac{1}{\langle w(V)^2 \rangle} \in [0,1]\,, \quad w(V) = \frac{p(V)}{q(V)}$$

Ref.		N_{params}	ESS at β		
			4.0	5.0	6.0
Lüscher, NL [3]		8 non-zero values	42%	4%	<1%
This work	**A**	$14 \equiv 2_t \times 7_W$	91%	65%	26%
	B	$420 \equiv 10_t \times 42_W$	98%	88%	70%
Boyda *et al.* [8]		$\mathcal{O}(10^6)$ estimated	88%	75%	48%

numerical results



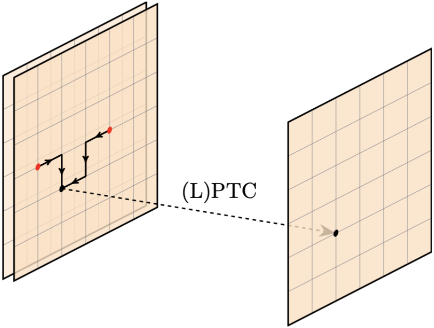


NN as preconditioners

precondition the Dirac equation

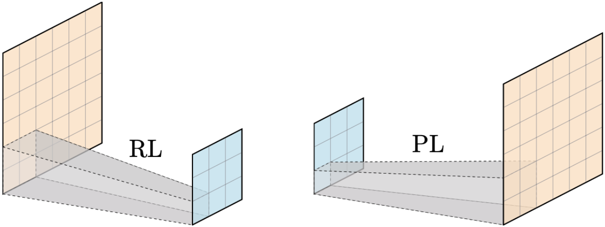
$$Du = b \quad \longrightarrow \quad (DM)\left(M^{-1}u\right) = b$$

construct a preconditioner using different types of layers [lehner & wettig 23]



- (L)PTC: $\psi_a(x) = \sum_{b,p} W_{ab}(x) T_p \phi_b(x)$

restriction/prolongation



$$\tilde{\psi}(y) = \sum_{x \in B(y)} W(y,x)\phi(x)$$

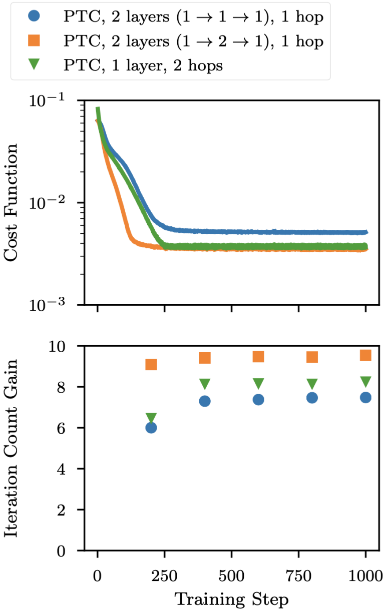
$$\psi(x) = W(y,x)^{\dagger}\tilde{\psi}(y)$$

hi-mode preconditioner

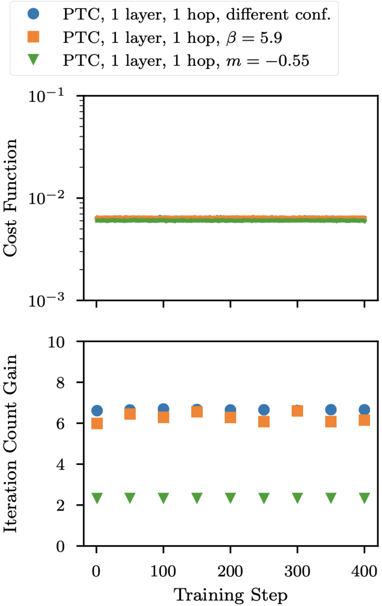
$$C = |MD_{WC}v - v|^2$$

generate training samples:

$$(v, D_{WC}v)$$



transfer learning



lo-mode preconditioner

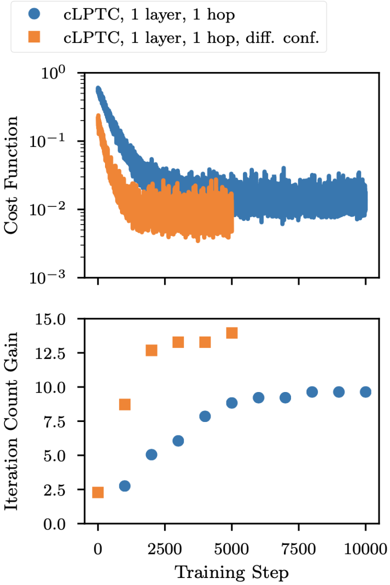
coarse-grid operator

$$\tilde{D} = R D_{WC} P$$

$$C = \left| \tilde{M} \tilde{D} v - v \right|^2$$

generate training samples:

$$\left(v, \tilde{D} v \right)$$



outlook

- ML provides interesting ways to define maps
- can be used for generative models/trivializing maps
- scaling of the cost of training
- test on systems with nontrivial topology
- tool to scale to large volumes, fine lattice spacings