EPJ Web of Conferences **258**, 09004 (2022) https://doi.org/10.1051/epjconf/202225809004
A Virtual Tribute to Quark Confinement and the Hadron Spectrum (vConf21)

Preserving gauge invariance in neural networks

Matteo Favoni[1,*] *Andreas* Ipp[1,**] *David I.* Müller[1,2,***] and *Daniel* Schuh[1,****]

[1]Institute for Theoretical Physics, TU Wien,
Wiedner Hauptstr. 8-10, 1040 Vienna, Austria
[2]Speaker and corresponding author

Abstract. In these proceedings we present lattice gauge equivariant convolutional neural networks (L-CNNs) which are able to process data from lattice gauge theory simulations while exactly preserving gauge symmetry. We review aspects of the architecture and show how L-CNNs can represent a large class of gauge invariant and equivariant functions on the lattice. We compare the performance of L-CNNs and non-equivariant networks using a non-linear regression problem and demonstrate how gauge invariance is broken for non-equivariant models.

1 Introduction

The success of convolutional neural networks (CNNs) in image recognition has demonstrated that adapting machine learning methods to the specifics of the problem at hand can lead to better performing models with fewer parameters. For example, the problem of image classification (e.g. determining in a series of images if a particular image shows a certain animal or determining a number shown in an image of hand-written digits) often does not require knowledge about where in the image a certain feature is detected, but just that the image contains it somewhere. In that sense, these problems are invariant under spatial translations. The main idea behind CNNs is to make use of this symmetry by restricting neural network layers to be translationally equivariant. In this context, translational equivariance refers to the fact that applying a spatial translation to an input image yields an appropriately shifted output of the CNN. The concept of equivariance in neural networks has been extended to more general global symmetries (e.g. rotations, reflections) in the framework of group equivariant CNNs [1] or G-CNNs, where G refers to a symmetry group. Neural networks have also been generalized to local symmetry, specifically in the case of data defined on curved manifolds [2]. More generally, geometric deep learning [3, 4] is used as an umbrella term for the concept of incorporating the geometry of a machine learning problem in the choice of network architecture.

Neural network architectures that exhibit symmetry properties are well suited for applications in physics (see e.g. [5] for a pedagogical introduction), in particular methods that are tailored to preserve the symmetries of physical theories. For example, CNNs (or more

*e-mail: favoni@hep.itp.tuwien.ac.at
**e-mail: ipp@hep.itp.tuwien.ac.at
***e-mail: dmueller@hep.itp.tuwien.ac.at
****e-mail: schuh@hep.itp.tuwien.ac.at

generally G-CNNs) can be applied to problems in lattice field theory [6–12], which typically exhibit translation, rotation and reflection symmetry. Recently, machine learning models with built-in local symmetry have also been applied in the context of Abelian and non-Abelian lattice gauge theories, e.g. as generative models for Monte Carlo simulations [13–15]. In these proceedings we review some results of our recent work [16] on lattice gauge equivariant convolutional neural networks (L-CNNs), which is a general framework for networks that by construction preserve lattice gauge symmetry in pure SU(N_c) gauge theory. We first review some notation for lattice gauge theory in Sec. 2 and then introduce the L-CNN in Sec. 3. Finally, we demonstrate the performance of L-CNNs compared to non-equivariant CNNs in Sec. 4.

2 Lattice gauge theory

The link formalism of lattice gauge theory due to Wilson [17] allows us to construct lattice discretizations of non-Abelian Yang-Mills theory with exact lattice gauge symmetry. In the lattice formulation, gauge fields A_μ are replaced by (gauge) link variables $U_{x,\mu} \in \mathrm{SU}(N_c)$, defined on the edges of a hypercubic lattice Λ with lattice spacing a, which connect the starting lattice site x to the end at $x + \mu$.[1] For explicitness, we assume the lattice to be finite with periodic boundary conditions. We denote the dimension of the lattice by $D + 1$, where $D \geq 1$ refers to the spatial dimensions and $\mu = 0$ is the (imaginary) time direction. The size of the lattice is given by $N_t \cdot N_s^D$. In terms of the gauge field $A_\mu(x) \in \mathfrak{su}(N_c)$, a gauge link $U_{x,\mu}$ is given by the path-ordered exponential[2]

$$U_{x,\mu} = \mathcal{P} \exp\left(i \int_0^1 \mathrm{d}s \frac{\mathrm{d}x^\nu(s)}{\mathrm{d}s} A_\nu(x(s)) \right), \tag{1}$$

where $x(s) : [0, 1] \to \mathbb{R}^4$ defines the straight-line path connecting x to $x+\mu$, and $\mathbb{R}^4$ is the four-dimensional Euclidean spacetime. The geometric interpretation of gauge links is that they describe parallel transport along the edges of the lattice. Under general gauge transformations $\Omega : \mathbb{R}^4 \to \mathrm{SU}(N_c)$ of the gauge field,

$$A_\mu(x) \to \Omega(x) \left(A_\mu(x) - i\partial_\mu \right) \Omega^\dagger(x), \tag{2}$$

the gauge links transform according to

$$U_{x,\mu} \to \Omega_x U_{x,\mu} \Omega^\dagger_{x+\mu}. \tag{3}$$

For concreteness, we use the fundamental representation of $\mathfrak{su}(N_c)$ and SU(N_c) to represent A_μ and $U_{x,\mu}$ as complex matrices. The inverse link is denoted by $U_{x+\mu,-\mu} = U^\dagger_{x,\mu}$. In the limit of small lattice spacing $a \to 0$, gauge links can be approximated with the matrix exponential $U_{x,\mu} \simeq \exp\left(iaA_\mu(x + \frac{1}{2}\mu)\right)$ at the mid-point $x + \frac{1}{2}\mu$.

Multiple links connecting consecutive points can be multiplied to form Wilson lines along arbitrary paths on the lattice, and, in particular, closed paths or Wilson loops, where the start and end point coincide. The simplest loop is the 1×1 loop called plaquette

$$U_{x,\mu\nu} = U_{x,\mu} U_{x+\mu,\nu} U_{x+\mu+\nu,-\mu} U_{x+\mu,-\mu}, \tag{4}$$

[1] We use $x + \mu$ to denote the point $x + a\hat{e}_\mu$, where $\hat{e}_\mu$ is a Euclidean basis vector on the lattice.

[2] Here we use the convention that the path ordering operator $\mathcal{P}$ shifts fields earlier in the path to the left, and fields later to the right of the product, i.e. $\mathcal{P}(O(a)O(b)) = O(a)O(b)$ if $a < b$ and $\mathcal{P}(O(a)O(b)) = O(b)O(a)$ if $b < a$.

which under gauge transformations transforms according to

$$U_{x,\mu\nu} \to \Omega_x U_{x,\mu\nu} \Omega_x^\dagger. \tag{5}$$

In the continuum limit, the plaquette approximates the non-Abelian field strength tensor $F_{\mu\nu} = \partial_\mu A_\nu - \partial_\nu A_\mu - i\left[A_\mu, A_\nu\right]$ via $U_{x,\mu\nu} \simeq \exp\left(ia^2 F_{\mu\nu}\right)$. The Yang-Mills action can then be approximated by the Wilson action [17]

$$S_W[U] = \frac{2}{g^2} \sum_{x\in\Lambda} \sum_{\mu<\nu} \mathrm{Re}\,\mathrm{Tr}\left[\mathbb{1} - U_{x,\mu\nu}\right], \tag{6}$$

where $g > 0$ is the Yang-Mills coupling constant. Because of the trace over locally transforming plaquettes, the Wilson action is invariant under general lattice gauge transformations. This gauge invariant action can be used to perform Monte Carlo sampling of the path integral and compute expectation values of observables in lattice QCD at finite temperature [18].

3 Lattice gauge equivariant convolutional neural networks

This section reviews some aspects of the L-CNN architecture [16]. Similar to conventional CNNs, we construct our architecture from elementary layers, which explicitly respect lattice gauge symmetry. First, we define the data points for L-CNNs as tuples $(\mathcal{U}, \mathcal{W})$, where $\mathcal{U} = \{U_{x,\mu}\}$ is the set of gauge links of a particular lattice configuration, and $\mathcal{W} = \{W_{x,i}\}$ with $W_{x,i} \in \mathbb{C}^{N_c \times N_c}$, $\forall x \in \Lambda$, $1 \le i \le N_{\mathrm{ch}}$ is a set of locally transforming complex matrices with $N_{\mathrm{ch}} \in \mathbb{N}$ channels. Under lattice gauge transformations, we require that the matrices $\mathcal{W}$ transform locally

$$W_{x,i} \to \Omega_x W_{x,i} \Omega_x^\dagger. \tag{7}$$

Comparing L-CNNs to CNNs, the $\mathcal{W}$ matrices can be thought of as feature maps, while the links $\mathcal{U}$ provide the geometrical information to compare data at different lattice sites via parallel transport. We consider data points that are related by gauge transformations to be gauge equivalent.

Next, we need to specify what data the $\mathcal{W}$ matrices represent. In practice (see Sec. 4.1) we use plaquettes $U_{x,\mu\nu}$ at each point x on the lattice as input $\mathcal{W}$'s such that each face defined by the pair (μ, ν) is assigned to a particular channel. More generally, it is also possible to include all Polyakov loops (which are non-contractible loops wrapping around the periodic boundary of the lattice) at each lattice site in the input. We show why this is a particularly useful choice for input data in Sec. 3.2, where we prove that arbitrary contractible and non-contractible Wilson loops can be generated with L-CNNs using just two elementary layers.

3.1 Equivariant convolutions and bilinear layers

We introduce two gauge equivariant operations acting on data points $(\mathcal{U}, \mathcal{W})$: lattice gauge equivariant convolutions (L-Convs) and lattice gauge equivariant bilinear layers (L-Bilin). We formulate the layers in such a way that the output of these operations transforms consistently under Eqs. (3) and (7).

The L-Conv operation maps a data point $(\mathcal{U}, \mathcal{W})$ to a new data point $(\bar{\mathcal{U}}, \bar{\mathcal{W}})$ given by

$$\begin{aligned} \bar{U}_{x,\mu} &= U_{x,\mu}, \\ \bar{W}_{x,i} &= \sum_{j,\mu,k} \omega_{i,j,\mu,k} U_{x,k\cdot\mu} W_{x+k\cdot\mu,j} U^\dagger_{x,k\cdot\mu}, \end{aligned} \tag{8}$$

where $\omega_{i,j,\mu,k} \in \mathbb{C}$ are the trainable weight parameters (or kernel weights) with indices for output channels $1 \leq i \leq N_{\text{ch,out}}$, input channels $1 \leq j \leq N_{\text{ch,in}}$, lattice directions $0 \leq \mu \leq D$, and distances $-K \leq k \leq K$. Here, $K \in \mathbb{N}$ is the size of the kernel, which determines the receptive field, i.e. how many lattice points are considered when computing the convolution. The $\mathcal{W}$ matrices at different lattice sites are parallel transported to the common point x along straight paths using the Wilson lines constructed from links:

$$U_{x,k\cdot\mu} = \prod_{i=0}^{k-1} U_{x+i\cdot\mu,\mu} = U_{x,\mu} U_{x+\mu,\mu} U_{x+2\cdot\mu,\mu} \dots U_{x+(k-1)\cdot\mu,\mu}. \tag{9}$$

Under gauge transformations, Eqs. (3) and (7), we find

$$\bar{W}_{x,i} \to \sum_{j,\mu,k} \omega_{i,j,\mu,k} \Omega_x U_{x,k\cdot\mu} \Omega^\dagger_{x+k\cdot\mu} \Omega_{x+k,\mu} W_{x+k\cdot\mu,j} \Omega^\dagger_{x+k\cdot\mu} \Omega_{x+k\cdot\mu} U^\dagger_{x,k\cdot\mu} \Omega^\dagger_x = \Omega_x \bar{W}_{x,i} \Omega^\dagger_x, \tag{10}$$

which shows that gauge equivariance is satisfied. Analogous to convolutional layers in standard CNNs, the output of L-Conv is also equivariant under lattice translations.

The second layer we introduce is a local bilinear operation. The L-Bilin layer maps two input data points $(\mathcal{U}, \mathcal{W})$ and $(\mathcal{U}', \mathcal{W}')$ with $\mathcal{U}' = \mathcal{U}$ to a new data point $(\bar{\mathcal{U}}, \bar{\mathcal{W}})$

$$\begin{aligned} \bar{U}_{x,\mu} &= U_{x,\mu}, \\ \bar{W}_{x,i} &= \sum_{j,k} \alpha_{i,j,k} W_{x,j} W'_{x,k}, \end{aligned} \tag{11}$$

where $\alpha_{i,j,k} \in \mathbb{C}$ are trainable weights with indices $1 \leq i \leq N_{\text{ch,out}}$, $1 \leq j \leq N_{\text{ch,in}}$, and $1 \leq k \leq N'_{\text{ch,in}}$. The bilinear operation multiplies two locally transforming matrices at the same lattice site, which guarantees gauge equivariance:

$$\bar{W}_{x,i} \to \sum_{j,k} \alpha_{i,j,k} \Omega_x W_{x,j} \Omega^\dagger_x \Omega_x W'_{x,k} \Omega^\dagger_x = \Omega_x \bar{W}_{x,i} \Omega^\dagger_x. \tag{12}$$

As in the case of L-Conv, L-Bilin is equivariant under translations as well. The bilinear operation reduces to a quadratic layer when we use the same data point for both arguments, i.e. $(\mathcal{U}, \mathcal{W}) = (\mathcal{U}', \mathcal{W}')$.

Both operations only modify the $\mathcal{W}$ part of the tuple $(\mathcal{U}, \mathcal{W})$. It is also possible to formulate equivariant layers that modify the links $\mathcal{U}$ (see e.g. L-Exp in Ref. [16]), but for the purposes of these proceedings we only focus on L-Conv and L-Bilin. We also note that the expressivity of these layers can be further increased by adding additional channels to $\mathcal{W}$ prior to the operation. Specifically, before applying L-Conv or L-Bilin one may add unit elements and hermitian conjugates:

$$(W_{x,1}, W_{x,2}, \dots, W_{x,N_{\text{ch,in}}}) \to (\mathbb{1}, W_{x,1}, W_{x,2}, \dots, W_{x,N_{\text{ch,in}}}, W^\dagger_{x,1}, W^\dagger_{x,2}, \dots, W^\dagger_{x,N_{\text{ch,in}}}). \tag{13}$$

This increases the number of channels from N_{ch} to $2N_{\text{ch}} + 1$. The unit elements generate bias terms for L-Conv, and bias and linear terms for L-Bilin. Hermitian conjugates correspond to changing the orientation of input plaquettes or Polyakov loops.

3.2 Generating arbitrary loops

It is important to address which class of functions can be represented by neural network architectures. Universality theorems exist for fully connected neural networks [19] and deep

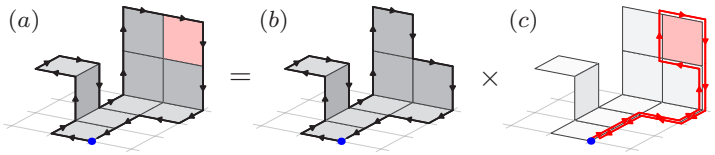

Figure 1. An arbitrary contractible Wilson loop on the lattice consisting of n squares (a) can be decomposed into a product of a loop with $n-1$ squares missing one plaquette (b) and a parallel transported single plaquette (c). Figure from [16].

CNNs [20], which demonstrate that neural networks are universal function approximators. In order to establish that L-CNNs are able to represent a large class of gauge equivariant functions, we prove by induction that arbitrary contractible and non-contractible Wilson loops can be constructed from repeated applications of L-Conv and L-Bilin.

Consider a data point tuple $(\mathcal{U}, \mathcal{W})$, where the set of $\mathcal{W}$ matrices are populated with all possible plaquettes at each lattice site. Initializing $\mathcal{W}$ in this manner trivially generates all possible 1×1 loops or loops of size 1. Next, consider an arbitrary contractible loop of size n (i.e. it consists of n squares) on the lattice, see Fig. 1 (a). Such a loop can be factorized into a loop of size $n-1$ (Fig. 1 (b)) with one missing plaquette and a transported single plaquette where the path traces along parts of the boundary of the original loop (Fig. 1 (c)). The factorization here refers to the composition of paths, which in the case of Wilson loops is realized by multiplying their matrix representations. L-CNNs can realize this factorization explicitly: Stacks of multiple L-Conv layers can generate arbitrary parallel transported plaquettes. For example, the parallel transported plaquette in Fig. 1 (c) can be generated with four L-Conv layers. On the other hand, the multiplication of the loop of size $n-1$ and the single transported plaquette can be realized by a single L-Bilin layer. By induction, every contractible loop can be constructed with L-Conv and L-Bilin operations starting from elementary plaquettes. This construction can be generalized to non-contractible loops (which wrap around the boundary of the lattice) by including, in addition to plaquettes, all possible straight line Polyakov loops at every lattice site in the initial set of $\mathcal{W}$ matrices.

Generally, L-CNNs consisting of stacks of L-Conv and L-Bilin layers can represent arbitrary linear combinations of Wilson loops of various shapes and sizes. The largest possible size and shape is determined by the kernel sizes and the number of layers. By including gauge equivariant non-linear activation functions in the L-CNN architecture (as introduced in our paper [16]), it is possible to represent non-linear functions of loops as well. Finally, if the desired output of an L-CNN is supposed to be gauge invariant (e.g. in the case of a regression problem for gauge invariant observables), it is possible to simply compute the trace of all matrices in $\mathcal{W}$. This renders the output of an L-CNN invariant because of the cyclic property of the trace.

4 Results and discussion

In order to test our new architecture, we designed a non-linear regression problem and tried out various L-CNN and non-equivariant CNN (baseline) architectures to solve it. The performance of L-CNNs and baseline networks are compared, and we investigate the breaking of gauge symmetry in baseline CNNs using adversarial attacks.

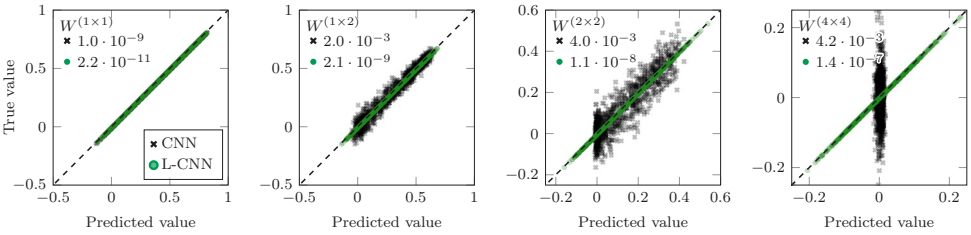

Figure 2. Scatter plots of our best L-CNN and baseline CNN models according to validation MSE on $8 \cdot 8$ lattices. True values of the Wilson loop are plotted against the predictions of the L-CNN model (green dots) and the baseline CNN (black crosses). High accuracy is achieved when all points are distributed close to the 45° line. The MSE is stated in the upper left corner of each panel. We observe that our best L-CNN models achieve low MSE for all studied loop sizes, while baseline CNNs perform worse at larger loop sizes. Figure adapted from [16].

4.1 Wilson loop regression

The proposed regression problem consists of two-dimensional ($D = 1$) lattice configurations sampled from an SU(2) Monte Carlo simulation, from which networks should compute the real value of the trace of $n \times m$ Wilson loops, i.e.

$$W_{x,\mu\nu}^{(m\times n)} = \frac{1}{N_c} \mathrm{Re}\,\mathrm{Tr}\left[U_{x,\mu\nu}^{(m\times n)}\right], \tag{14}$$

which is a gauge invariant observable. In our computational experiments we use 1×1 (as a trivial example), 1×2, 2×2 and 4×4 loops. We sample configurations for various values of the coupling constant on an $N_t \cdot N_s = 8 \cdot 8$ lattice to create the training and validation set. Test sets are generated for lattice sizes up to $64 \cdot 64$. For each architecture, trainable weights are randomly initialized multiple times and independently trained. We use mean squared error (MSE) as the objective (or loss) function to optimize during training. Both L-CNNs and baseline CNNs are provided with links and plaquettes in the input layer. For more details regarding architectures and the training process, we refer to our original paper and its forthcoming supplementary materials [16].

Summarizing the main results of our baseline study, we find that L-CNNs are indeed able to solve the regression problem for all sizes of the loop to a high degree of accuracy and also exhibit generalization to larger lattices. Baseline CNNs typically are not able to find adequate solutions, and the quality of their fits deteriorates with increased loop size. In Fig. 2 we show scatter plots for the best models of each type, where true values (i.e. labels in the dataset) are plotted against the predictions made by the networks.

4.2 Adversarial attacks for non-equivariant networks

While our L-CNN architectures are gauge invariant by construction (up to numerical precision), the baseline CNNs must learn this symmetry during the training process. Therefore, it is expected that even well-trained baseline CNNs at best only exhibit approximate gauge symmetry. The extent to which gauge symmetry is broken can be studied by applying gauge transformations to the input layer

$$U_{x,\mu} \to \Omega_x U_{x,\mu} \Omega_{x+\mu}^\dagger, \qquad W_{x,i} \to \Omega_x W_{x,i} \Omega_x^\dagger, \tag{15}$$

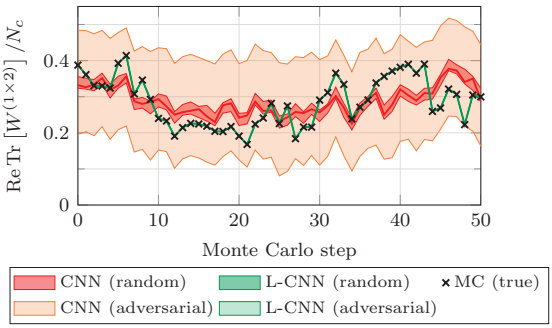

Figure 3. Uncertainty in predictions due to broken gauge symmetry in L-CNNs (green bands) and baseline CNNs (red bands) on 8·8 test data for the 1×2 loop regression problem. The true values (labels) are shown as black crosses. We use the same models as in Fig. 2. L-CNNs are invariant by construction and therefore only show deviations on the order of numerical precision. Baseline CNNs show much larger deviations, where the dark red band corresponds to errors due to random transformations, and the light red band shows the result of adversarial attacks. Figure from [16].

with $\Omega_x \in \mathrm{SU}(N_c)$ and determining how much the predictions of a non-equivariant model change as a result. For a given model and a given lattice configuration, we apply two types of transformations: multiple random gauge transformations, where each Ω_x is sampled randomly, and adversarial attacks, where Ω_x is optimized to produce the largest deviation between the original prediction and the gauge transformed prediction. The adversarial attack is unique to each configuration and model, while random transformations do not depend on either. We show the results of such an attack for our best L-CNN and baseline CNN models for the 1×2 loop in Fig. 3. The L-CNN model is entirely unaffected by transformations (up to numerical precision), but the predictions of the baseline network show large errors in the case of adversarial attacks.

5 Conclusions

In these proceedings we have reviewed some aspects of the L-CNN architecture, particularly the L-Conv and L-Bilin layer, which in combination can be used to generate arbitrarily shaped Wilson loops on the lattice. We have also presented comparisons between L-CNNs and non-equivariant CNNs using a non-linear regression task for Wilson loops and demonstrated the breaking of gauge invariance in non-equivariant networks.

In this work we have mainly focused on the formulation and some properties of L-CNNs, but from a practical perspective, it would be interesting to apply the L-CNN architecture to normalizing [13–15] and continuous [12] flow models. On the other hand, in order to provide a more solid mathematical foundation, it would be worthwhile to understand L-CNNs as a special case of CNNs on principal bundles in the vein of gauge equivariant CNNs for curved manifolds [2] or gauge equivariant convolutions defined in [4].

DM thanks Jimmy Aronsson for many helpful discussions regarding equivariance, neural networks and geometric deep learning. This work has been supported by the Austrian Science Fund FWF No. P32446-N27, No. P28352 and Doctoral program No. W1252-N27. The Titan V GPU used for this research was donated by the NVIDIA Corporation.

References

[1] T.S. Cohen, M. Welling, *Group Equivariant Convolutional Networks*, in *Proceedings of The 33rd International Conference on Machine Learning* (JMLR, 2016), Vol. 48, pp. 2990–2999, 1602.07576

[2] T.S. Cohen, M. Weiler, B. Kicanaoglu, M. Welling, *Gauge Equivariant Convolutional Networks and the Icosahedral CNN*, in *Proceedings of the 36th International Conference on Machine Learning* (JMLR, 2019), Vol. 97, pp. 1321–1330, 1902.04615

[3] M.M. Bronstein, J. Bruna, T. Cohen, P. Veličković (2021), 2104.13478

[4] J.E. Gerken, J. Aronsson, O. Carlsson, H. Linander, F. Ohlsson, C. Petersson, D. Persson (2021), 2105.13926

[5] P. Mehta, M. Bukov, C.H. Wang, A.G. Day, C. Richardson, C.K. Fisher, D.J. Schwab, Physics Reports **810**, 1 (2019), a high-bias, low-variance introduction to Machine Learning for physicists

[6] K. Zhou, G. Endrődi, L.G. Pang, H. Stöcker, Phys. Rev. D **100**, 011501 (2019), 1810.12879

[7] D.L. Boyda, M.N. Chernodub, N.V. Gerasimeniuk, V.A. Goy, S.D. Liubimov, A.V. Molochkov, Phys. Rev. D **103**, 014509 (2021), 2009.10971

[8] S. Blücher, L. Kades, J.M. Pawlowski, N. Strodthoff, J.M. Urban, Phys. Rev. D **101**, 094507 (2020), 2003.01504

[9] D. Bachtis, G. Aarts, B. Lucini, Phys. Rev. E **102**, 053306 (2020), 2007.00355

[10] S. Bulusu, M. Favoni, A. Ipp, D.I. Müller, D. Schuh, Phys. Rev. D **104**, 074504 (2021), 2103.14686

[11] D. Bachtis, G. Aarts, B. Lucini, Phys. Rev. D **103**, 074510 (2021), 2102.09449

[12] P. de Haan, C. Rainone, M. Cheng, R. Bondesan (2021), 2110.02673

[13] G. Kanwar, M.S. Albergo, D. Boyda, K. Cranmer, D.C. Hackett, S. Racanière, D.J. Rezende, P.E. Shanahan, Phys. Rev. Lett. **125**, 121601 (2020), 2003.06413

[14] D. Boyda, G. Kanwar, S. Racanière, D.J. Rezende, M.S. Albergo, K. Cranmer, D.C. Hackett, P.E. Shanahan, Phys. Rev. D **103**, 074504 (2021), 2008.05456

[15] M.S. Albergo, D. Boyda, D.C. Hackett, G. Kanwar, K. Cranmer, S. Racanière, D.J. Rezende, P.E. Shanahan (2021), 2101.08176

[16] M. Favoni, A. Ipp, D.I. Müller, D. Schuh (2020), 2012.12901

[17] K.G. Wilson, Phys. Rev. D **10**, 2445 (1974)

[18] K.G. Wilson, *Monte-Carlo calculations for the lattice gauge theory*, in *Recent Developments in Gauge Theories. Proceedings, Nato Advanced Study Institute, Cargese, France, August 26 - September 8, 1979*, edited by G. 't Hooft, C. Itzykson, A. Jaffe, H. Lehmann, P. Mitter, I. Singer, R. Stora (1980), Vol. 59, pp. 363–402

[19] Z. Lu, H. Pu, F. Wang, Z. Hu, L. Wang (2017), 1709.02540

[20] D.X. Zhou, Applied and Computational Harmonic Analysis **48**, 787 (2020), 1805.10769