# Applications of Normalizing Flows as Generative Models in Lattice Field Theory 

Javad Komijani (with Marina K. Marinkovic)

EHHzürich

Efficient Simulations on GPU Hardware
ETH Zürich, October 2022

## Lattice Field Theory \& Monte Carlo Simulations

- Path integral formulation \& imaginary time \& discretization \& Monte Carlo simulations

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



Lattice QCD: Monte Carlo Simulations

- Monte Carlo simulations:
- Draw samples from $\frac{1}{Z} e^{-S[\phi]}$ distribution (weight of each path/configuration)
- Methods based on local updating suffer from: critical slowing down, topological freezing, ...



## Lattice Field Theory \& Trivializing Maps

## Trivializing Maps, the Wilson Flow and the HMC Algorithm

## Martin Lüscher

CERN, Physics Department, 1211 Geneva 23, Switzerland. E-mail: luscher@mail.cern.ch

Received: 9 August 2009 / Accepted: 24 September 2009
Published online: 24 November 2009 - © Springer-Verlag 2009

Abstract: In lattice gauge theory, there exist field transformations that map the theory to the trivial one, where the basic field variables are completely decoupled from one another. Such maps can be constructed systematically by integrating certain flow equations in field space. The construction is worked out in some detail and it is proposed to combine the Wilson flow (which generates approximately trivializing maps for the Wilson gauge action) with the HMC simulation algorithm in order to improve the efficiency of lattice QCD simulations.
the substitution $U \rightarrow V$ of the integration variables in the functional integral maps the theory to the trivial one where the link variables are completely decoupled from one another. The expectation values (2.1) are then given by

$$
\begin{equation*}
\langle\mathcal{O}\rangle=\int \mathrm{D}[V] \mathcal{O}(\mathcal{F}(V)) \tag{2.9}
\end{equation*}
$$

Such trivializing maps thus contain the entire dynamics of the theory.
Although the remark is likely to remain an academic one, an intriguing observation is that the integral (2.9) can be simulated simply by generating uniformly distributed random gauge fields. Subsequent field configurations are uncorrelated in this case and

## Inverse Transform Sampling \& Normalizing Flows

- Inverse transform sampling (ITS) as a method to generate a random variable with a flexible distribution:

$$
y \triangleq F_{Y}^{-1} \circ F_{X}(x)
$$




- Normalizing flows (NFs) as a generalization of ITS to higher dimensions with a series of learnable, invertible transformations



## Normalizing Flows \& \{Statistical Physics, Lattice Field Theory\}

- NF for (a dual version of) Ising model in 2 dim [arXiv:1802.02840]
- NF for scalar theories in 2-dim [arXiv:1904.12072, 2002.02428, 2003.06413]


## Requirements

- Prior PDF: $f_{\Xi}(\xi)$
- Target PDF: $f_{\Phi}(\phi) \propto e^{-S[\phi]}$
- NF neural network $\phi=T(\xi)$


## Train (self learning by minimizing the loss)

- Sample a batch of variables from the prior
- Transform the batch of variables \& calculate the Jacobian
- Loss $=D_{\mathrm{KL}}\left(q_{\text {transformed }} \| p_{\text {target }}\right)$

- Kullback-Leibler divergence measures how similar two distributions are:

$$
D_{\mathrm{KL}}(q \| p) \equiv \int d \phi q[\phi](\log q[\phi]-\log p[\phi]) \geq 0
$$

## Requirements

- Prior PDF: $f_{\Xi}(\xi)$
- Target PDF: $f_{\Phi}(\phi) \propto e^{-S[\phi]}$
- NF neural network $\phi=T(\xi)$

Train (self learning by minimizing the loss)

- Sample a batch of variables from the prior
- Transform the batch of variables \& calculate the Jacobian
- Loss $=D_{\text {KL }}\left(q_{\text {transformed }}| | p_{\text {target }}\right)$



## Generate samples \& ensure exactness

- The method of normalizing flows yields an approximate distribution (close to target but not identical)
- One needs to exploit other methods to ensure the exactness of the final distribution
- For an unbiased estimate, one can use the Metropolis-Hastings (MH) algorithm to accept/reject proposed configurations:

$$
p_{\text {accept }}\left(\phi^{\prime} \mid \phi^{i-1}\right)=\min \left(1, \frac{q\left(\phi^{i-1}\right)}{p\left(\phi^{i-1}\right)} \frac{p\left(\phi^{\prime}\right)}{q\left(\phi^{\prime}\right)}\right)
$$

# 1. Networks for Normalizing Flows 2. Poor Scaling at Large Volumes 

## Designing Networks for NF: Review Papers

- "Normalizing Flows: An Introduction and Review of Current Methods" [IEEE Transactions on Pattern Analysis and Machine Intelligence 43, 3964 (2021)]

Normalizing Flows should satisfy several conditions in order to be practical. They should:

- be invertible; for sampling we need $\mathbf{g}$ while for computing likelihood we need f ,
- be sufficiently expressive to model the distribution of interest,
- be computationally efficient, both in terms of computing $f$ and $g$ (depending on the application) but also in terms of the calculation of the determinant of the Jacobian.
In the following section, we describe different types of flows and comment on the above properties. An overview of the methods discussed can be seen in Fig. 2.

- "Normalizing Flows for Probabilistic Modeling and Inference" [Journal of Machine Learning Research 22, 1 (2021)]



## Designing Networks for NF: Coupling Layers

- Coupling Flows are popular
- enable highly expressive transformations
- have triangular Jacobian matrices
- e.g., can be implemented using checkerboard partitioning

1. Partition the data to blue (active) \& red (frozen)
2. Transform the active data $x_{\mathrm{a}} \rightarrow x_{\mathrm{a}}=T\left(x_{\mathrm{a}} \mid h\left(x_{\mathrm{f}}\right)\right)$
3. Change the labels and repeat 2 ( $\mathrm{w} /$ new functions)


- $T$ is typically a point-wise transformation like affine \& spline

$$
T\left(x_{\mathrm{a}} \mid h\left(x_{\mathrm{f}}\right)\right)=h_{1}\left(x_{\mathrm{f}}\right) x_{\mathrm{a}}+h_{2}\left(x_{\mathrm{f}}\right)
$$

- $h_{i}$ are typically constructed using Dense NNs or Convolutional NNs
- Several layers of coupling flows can be added sequentially

[Journal of Machine Learning Research 22, 1 (2021)]


## Designing Networks for NF: Examples with Coupling Layers

Proof of principle studies are done for several theories in 2 dimensions

- Scalar theory with quartic potential [arXiv:1904.12072, arXiv:2105.12481]

$$
S[\phi]=\int d^{2} x\left(\frac{1}{2}\left(\partial_{\mu} \phi\right)^{2}+\frac{1}{2} m^{2} \phi^{2}+\lambda \phi^{4}\right)
$$

- A Jupiter notebook can be found here [arXiv:2101.08176]
- scalar theory on a $8 \times 8$ lattice
- 16 layers of layers of affine coupling layers for $T$ and 3 layers of CNNs for $h$
- acceptance rate $\approx 0.5$
- $\mathrm{U}(1)$ gauge ( 2 dim ) [arXiv:2003.06413]
- $\operatorname{SU}(n)$ gauge theories (2 dim) [arXiv:2008.05456]
- Staggered fermions coupled to a scalar field via a Yukawa interaction (2 dim) [arXiv:2106.05934]
- $\operatorname{SU}(3)$ gauge theories with 2 flavors of fermions (2 dim)[arXiv:2207.08945]


## Designing Networks for NF: Other Architectures?

## Dense (Linear) Net

- Great for small-size lattice
- Number of parameter $\propto N^{2}$


## Conv Net

- Respect translational symmetry \& fewer parameters
- Many layers needed to correlate a big lattice


## Other possibilities?

What about constructing layers inspired by symmetries of the action \& effective theories to propagate correlation in more efficient ways?

## Effective Action \& Power Spectral Density

- A scalar field theory in $n$ spacetime dimensions:

$$
S[\phi]=\int d^{n} x\left(\frac{1}{2} \partial_{\mu} \phi \partial_{\mu} \phi+\frac{1}{2} m^{2} \phi^{2}+\sum_{j=3}^{J} g_{j} \phi^{j}\right)
$$

- The quantum effective action:

$$
\Gamma[\phi]=\frac{1}{2} \int d^{n} k \tilde{\phi}(-k)\left(k^{2}+m^{2}-\Pi\left(k^{2}\right)\right) \tilde{\phi}(k)+\cdots
$$

(Tree-level Feynman diagrams give the complete scattering amplitude) $\left(k^{2}+m^{2}-\Pi\left(k^{2}\right)\right)$ is the inverse of two-point correlator/Green's function
$\left(k^{2}+m^{2}-\Pi\left(k^{2}\right)\right)$ is the inverse of power spectral density

A close look to PSD:

- The inverse of PSD of a 1-dim double-well potential (from MC simulation)



- $1 /$ PSD can be manipulated using a positive, monotonically increasing function of $\hat{k}^{2}$; ML techniques can be employed to construct such a function
- Manipulating PSD is NOT a local operation; it affects the correlation in data at largest \& shortest scales

The inverse of PSD for a 2-dim double-well potential (from MC simulation)


## Inspired by mean-field theory

One can build a general function (a neural network) to map the mean field to a mean field of interest

For the sake of comparison with [arXiv:2105.12481, Debbio et.al.] we consider this 2-dim action

$$
S[\phi]=\int d x^{2}\left\{\frac{\kappa}{2}\left(\partial_{\mu} \phi(x)\right)^{2}+\frac{m^{2}}{2} \phi(x)^{2}+\lambda \phi(x)^{4}\right\}
$$

where $\kappa=\beta, m^{2}=-4 \beta$, and $\lambda=0.5$, with $\beta \in[0.5,0.8]$ in our simulations.

## Goal:

Following suggestions inspired by effective theories, we aim to construct neural networks that are

- economic w.r.t. parameters
- do not require many layers of ConvNet to propagate correlations


## An architecture for 2-dim scalar theories

1 an initial layer to manipulate PSD of white normal noise \& general activation
2 followed by two layers of affine coupling implemented with ConvNet \& general activation
Simulation parameters: $\kappa=0.6 \& L=32$









## Acceptance rate \& critical point \& large volume



Compare with [arXiv:2105.12481, Debbio et.al.]


- $L \in[8,64]$
- \# parameters $\approx 3.4 \mathrm{~K}$ for all cases
- trained with transfer learning
- poor scaling @ critical point




## Uncertainty in $\log (q / p)$ \& acceptance rate

- Optimization for $\kappa=0.5$ for $L \in\{8,16,32,64\}$ :

- The uncertainty in $\log (q / p)$ determines acceptance rate
- It looks like uncertainty in $\log (q / p)$ scales with $\sqrt{\text { volume }}$ at large volumes
- Justification: divide the lattice into $n$ blocks with almost independent fluctuations


## NFs with block updating

- The uncertainty in $\log (q / p)$ determines acceptance rate
- It looks like uncertainty in $\log (q / p)$ scales with $\sqrt{\text { volume }}$ at large volumes
- Justification: divide the lattice into $n$ blocks with almost independent fluctuations

- $L=64$ model is sampled block by block
- $n_{\text {blocks }}=2^{2}$ (square), acceptance rate $\sim L=32$
- $n_{\text {blocks }}=4^{2}$ (star), acceptance rate $\sim L=16$
- Asymptotic scaling \& saturated training


## Uncertainty in $\log (q / p)$ \& block size: Toy model

- Let $x_{n}$, with $n \geq 0$, be a sequence of iid normal variables with $\mathcal{N}\left(0, \sigma^{2}\right)$.
- Let $y_{n}$ is the output of the "Metropolis accept/reject"

$$
y_{n}=f\left(x_{n}, y_{n-1}\right)= \begin{cases}x_{n} & \text { with probability } e^{-\operatorname{Relu}\left(x_{n}-y_{n-1}\right)} \\ y_{n-1} & \text { otherwise }\end{cases}
$$







## Conclusion \& Outlook

- Effective theories to design layers changing the data at long\&short scales
- Still, the acceptance rate drops fast as the lattice volume increases
- Suggestion: Divide\&Conquer
- Divide the current sample into blocks \& update block by block
- Optimum block size
- In progress...

- Outlook: $\operatorname{SU}(n)$ gauge theories



## back-up slides

## Magnetization \& critical point \& (un)broken phase




## Block updating \& autocorrelation time



