# Generative models for scalar field theories: how to deal with poor scaling?

Javad Komijani & Marina Marinkovic
ETH zürich

Learning to Discover
29 April 2022

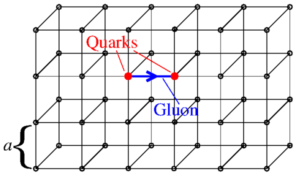
# Lattice Field Theory & Monte Carlo Simulations

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

$$\Downarrow$$

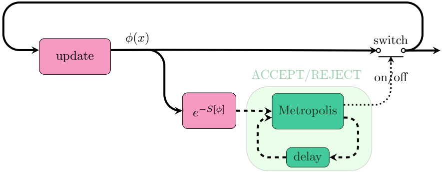
define & solve a field theory non-perturbatively

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




Lattice QCD: Monte Carlo Simulations

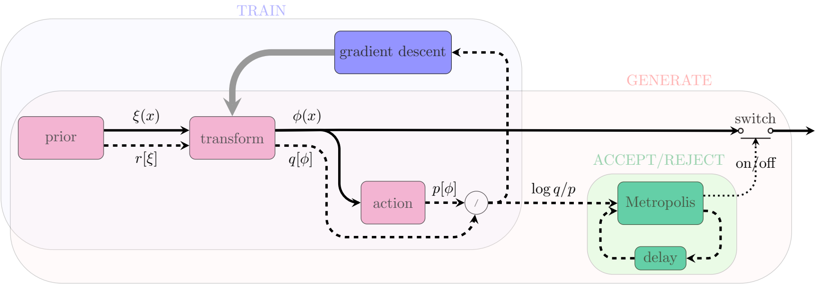
- Monte Carlo simulations:
  - Draw samples from $\frac{1}{Z} e^{-S[\phi]}$ distribution (weight of each path/configuration)
  - Methods based on local updating suffer from: critical slowing down, topological freezing, $\cdots$



# Alternative methods? Normalizing Flows, ...

- Normalizing Flows for Lattice Field Theory: new ideas (for scalar theories) are explored first in a series of papers by a group from MIT & Deep Mind: [arXiv:1904.12072, 2002.02428, & 2003.06413]

> Start from a simple stochastic process and transform it to the one of interest: $p[\phi] = e^{-S[\phi]}/Z$




- **Principle of minimum discrimination information**: (reverse) Kullback-Leibler (KL) divergences measure how similar two distributions are:

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

# *Designing Networks for Normalizing Flow*

# Designing Networks for Normalizing Flow

- Checkerboard strategy for Normalizing flow is widely used: divide the data to active & passive, update the active, and....
- Widely used layers of neural networks

**Linear (Dense) Net**

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

**Conv Net**

- Economic w.r.t parameters
- Many layers needed to correlate a big lattice

# Designing Networks for Normalizing Flow

- Checkerboard strategy for Normalizing flow is widely used: divide the data to active & passive, update the active, and....
- Widely used layers of neural networks

**Linear (Dense) Net**

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

**Conv Net**

- Economic w.r.t parameters
- Many layers needed to correlate a big lattice

- Other possibilities?

What about constructing layers inspired by effective theories to propagate correlation in more efficient ways?

# Effective Action & Power Spectral Density

- Let us consider a scalar field in $n$ spacetime dimensions with action

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

- We now define the quantum effective action

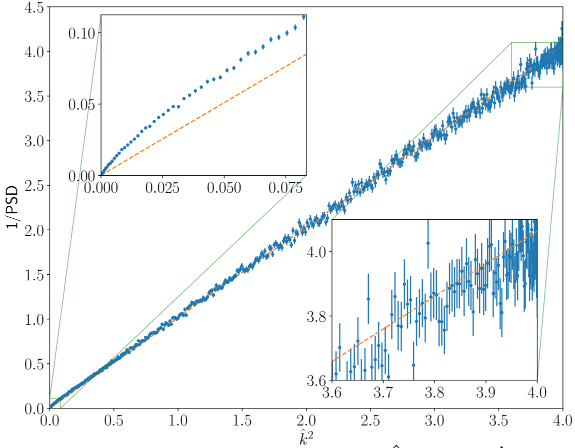
$$\Gamma[\phi] = \frac{1}{2} \int d^n k \, \tilde{\phi}(-k) \Big( k^2 + m^2 - \Pi(k^2) \Big) \tilde{\phi}(k) + \cdots$$

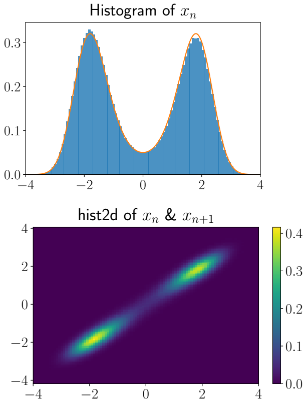
  The quantum action has the property that the tree-level Feynman diagrams it generates give the complete scattering amplitude of the original theory.

- $\Big( k^2 + m^2 - \Pi(k^2) \Big)$ is the inverse of two-point correlator/Green's function.
- $\Big( k^2 + m^2 - \Pi(k^2) \Big)$ is the inverse of power spectral density.

A close look to PSD:

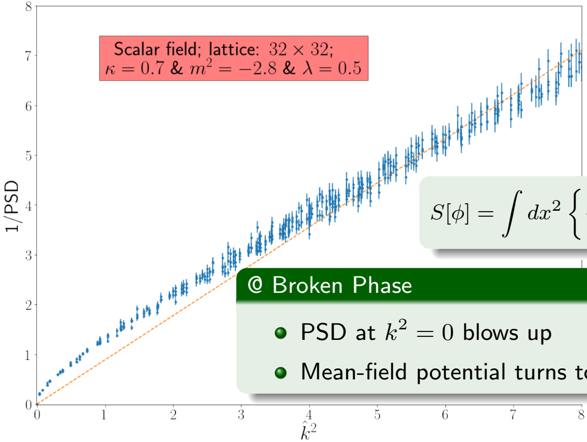
- The inverse of PSD of a 1-dim double-well potential (from MC simulation)




- Looks like a line with respect to $\hat{k} = 2\sin\frac{k}{2}$, but it is not!
- 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)




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

**@ Broken Phase**

- PSD at $k^2 = 0$ blows up
- Mean-field potential turns to a double-well potential

**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 action

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

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

**Goal:**

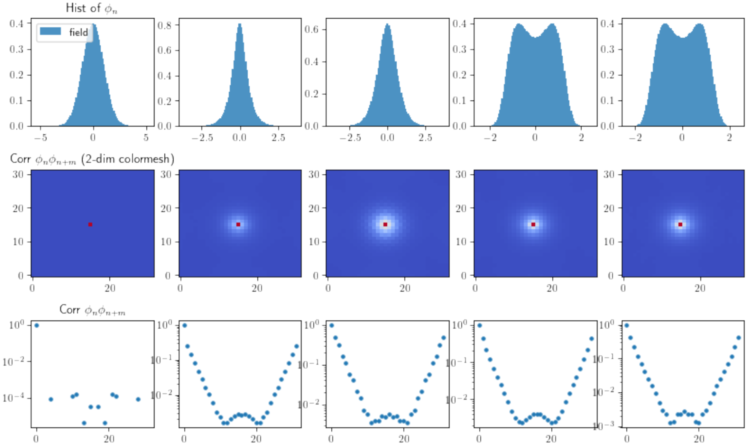
Following suggestions inspired by effective theories, we aim to construct neural networks that are

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

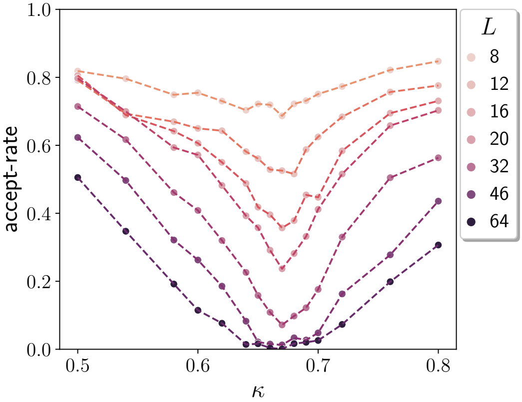
A model with

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

The last column shows only accepted configurations ($\kappa = 0.6$ & $L = 32$)

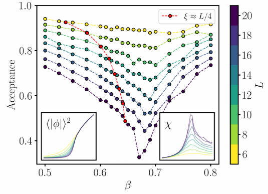


# Acceptance rate & critical point & large volume



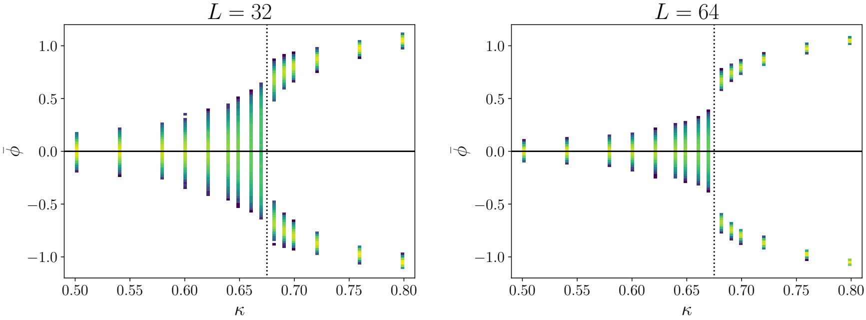

- $L \in [8, 64]$
- # parameters $\approx$ 3.4K for all cases
- Trained with transfer learning

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



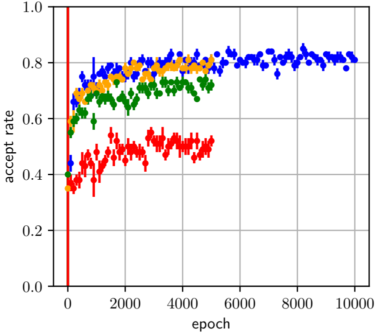

- $L \in [6, 20]$
- # parameters $\sim$ 100K

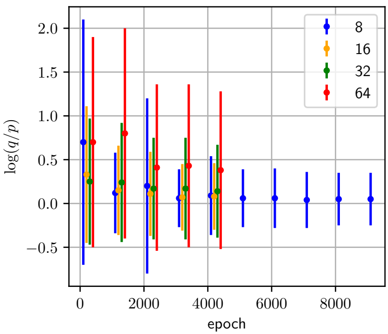
# Magnetization & critical point & (un)broken phase



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

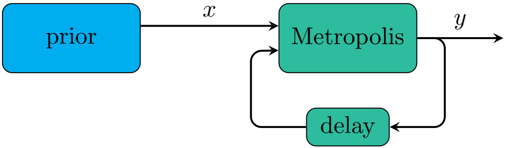
- The uncertainty in $\log(q/p)$ determines acceptance rate
- The uncertainty in $\log(q/p)$ scales with $\sqrt{\text{volume}}$ at large volumes
- Justification: divide the lattice into $n$ blocks with almost independent fluctuations
- Optimization for $\kappa = 0.5$ for $L \in \{8, 16, 32, 64\}$:

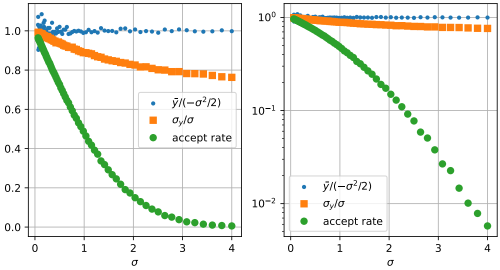


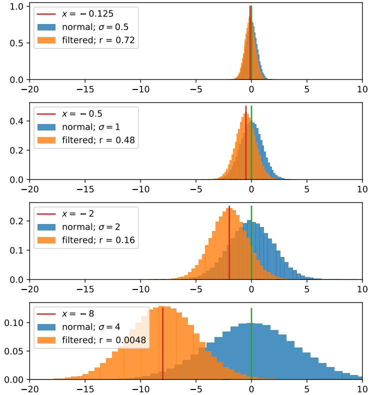



# Uncertainty in $\log(q/p)$ & acceptance rate

- Toy model: $x \sim N(0, \sigma^2)$ and $y$ is the output of the “Metropolis Filter”



# Concluding points & dealing with poor acceptance rate

- We briefly discussed how to use effective theories to design layers that affect the data at both long&short scales with limited number of parameters.
- Still, the acceptance rate drops fast as the lattice volume increases.
- Suggestion: Divide&Conquer
  - Instead of proposing a completely new sample, we can divide the current sample into blocks & update block by block.
  - The block size can be chosen such that the acceptance rate of updating a blocks gets relatively large (about 1/4 or so).
  - We can follow the strategy of *hit-till-accepted* for each block.
  - Needs to be checked: auto-correlation; in progress...