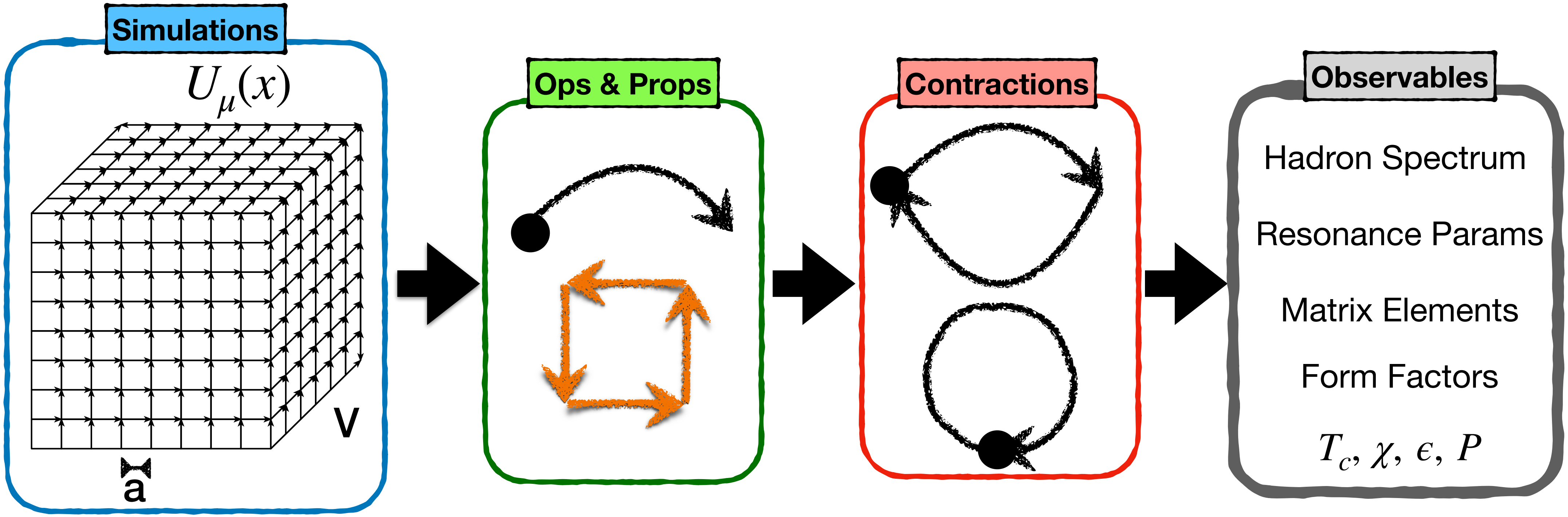
From Configurations to Correlators

— Lect. 2 : Machine Learning in Measurements

Srijit Paul
Maryland Center for Fundamental Physics

Satellite School on Lattice Gauge Theories,
Indian Institute of Science, Bangalore, India
29th October, 2025

Machine learning in Measurements



$$\langle \mathcal{O} \rangle_{a,V,n_f} = \frac{1}{\mathcal{Z}} \int \mathbb{D}[U]^V\ \mathbb{D}[\psi\bar{\psi}]^{Vn_f}\ \mathcal{O}(U,\psi,\bar{\psi})\ e^{-S(a;U,\psi,\bar{\psi})}$$

Machine learning in Measurements : Prerequisite

$e^{-S[U]}$ —MCMC→ $[U^k]$ —How can you use ML?→ $E[\mathcal{O}] = \frac{\sum_k \mathcal{O}[U^{(k)}]}{N_{\text{cfg}}}$

Ordered set

$\left[U_1, U_2, U_3, \cdots\ U_N\right]$

- ML methods expect **independent, identically distributed samples**
- Sequential MCMC generates **dependent, identically distributed samples.**
- Need **Bias correction.**

Machine learning in Measurements : Prerequisite

$e^{-S[U]}$ —MCMC→ $[U^k]$ —How can you use ML?→ $E[\mathcal{O}] = \frac{\sum_k \mathcal{O}[U^{(k)}]}{N_{\text{cfg}}}$

Ordered set

$\left[U_1, U_2, U_3, \cdots\ U_N\right]$

Cost of config. Generation?

[High]

[Low]

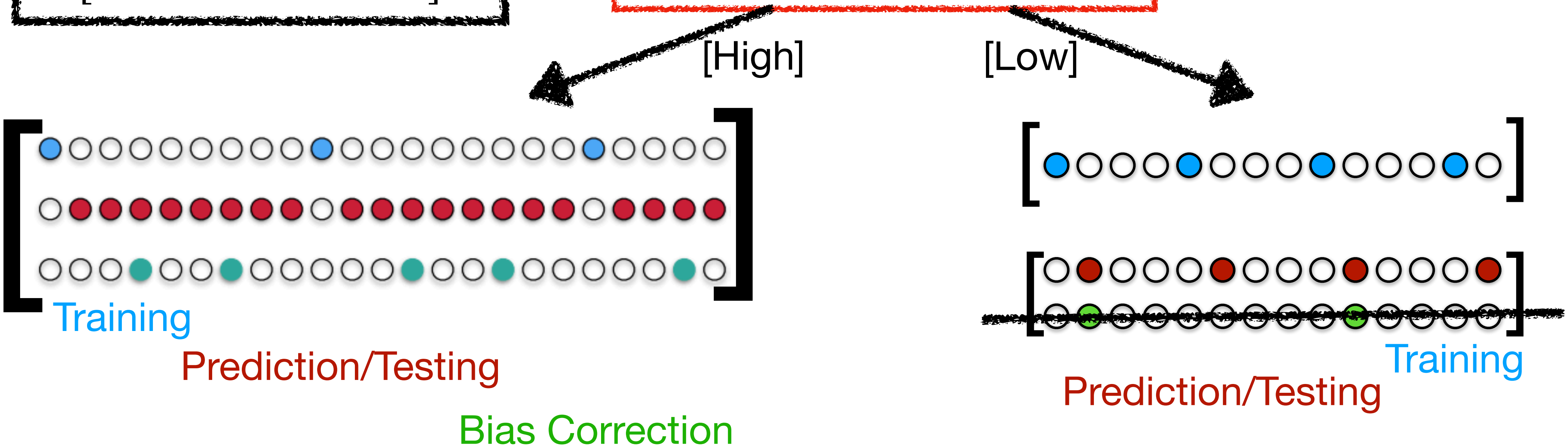
Machine learning in Measurements : Prerequisite

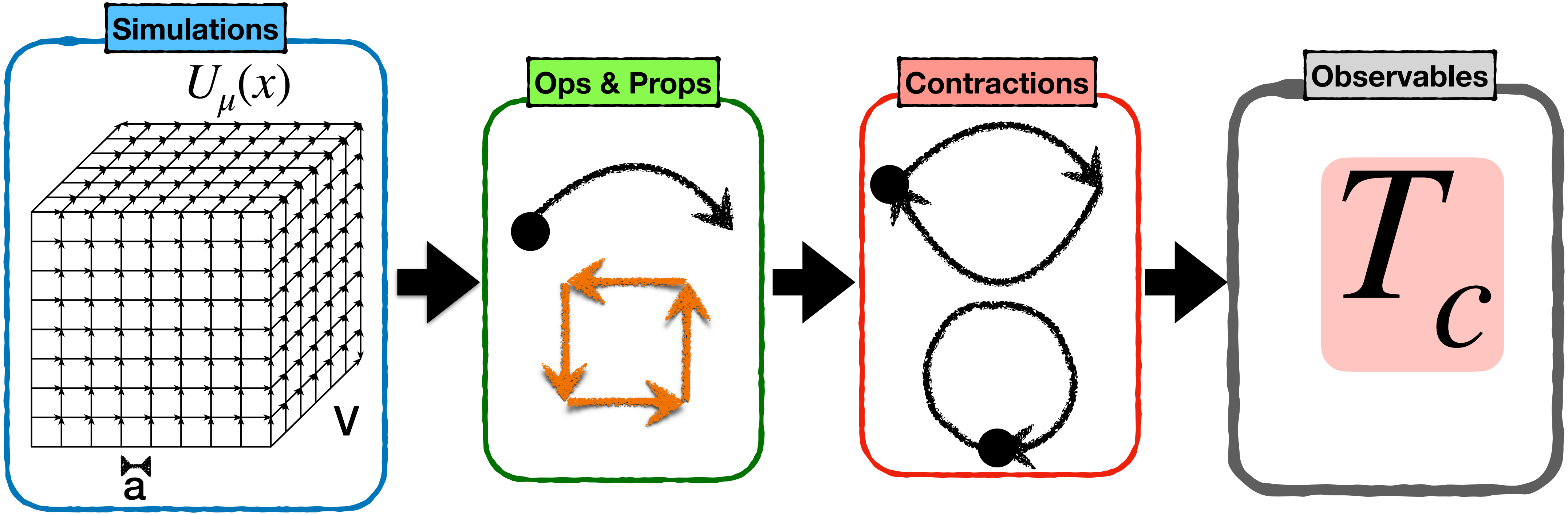
$e^{-S[U]}$ —MCMC→ $[U^k]$ —How can you use ML?→ $E[\mathcal{O}] = \frac{\sum_k \mathcal{O}[U^{(k)}]}{N_{\text{cfg}}}$

Ordered set

$[U_1, U_2, U_3, \cdots U_N]$

Cost of config. Generation?



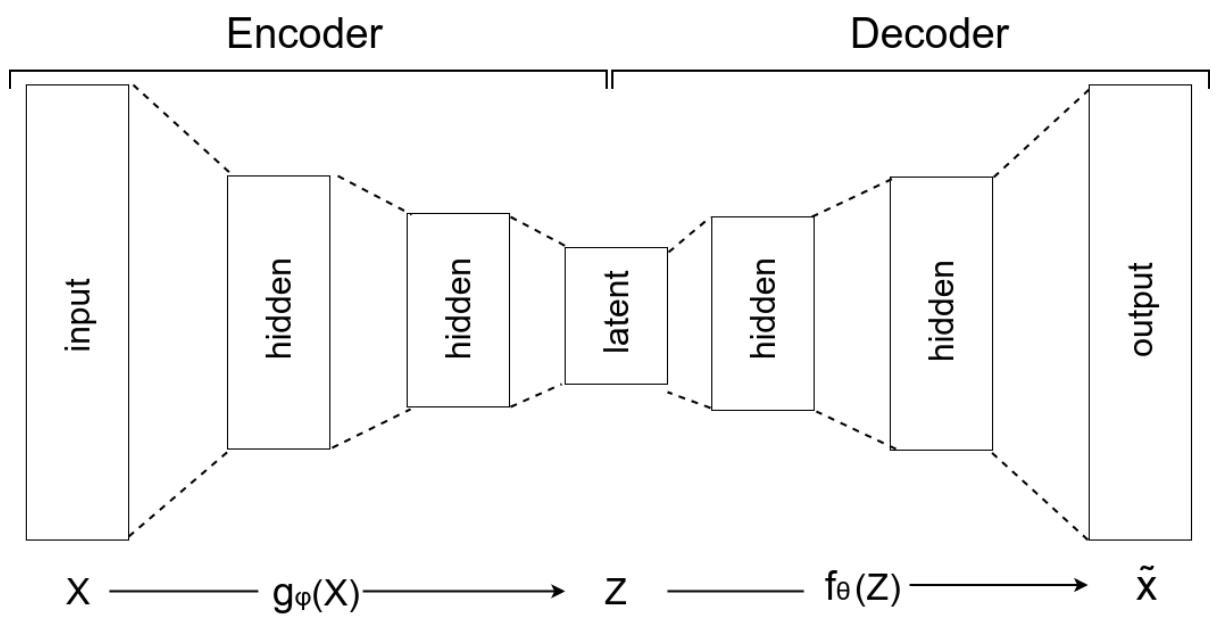


Simulations
$U_\mu(x)$
a
V
Ops & Props
Contractions
Observables
T_c

Identify Phase transitions

* Variational Autoencoder, unsupervised.
* Learn parameters of $X = P(\phi)$ distribution.

[Arxiv:1903.03506]

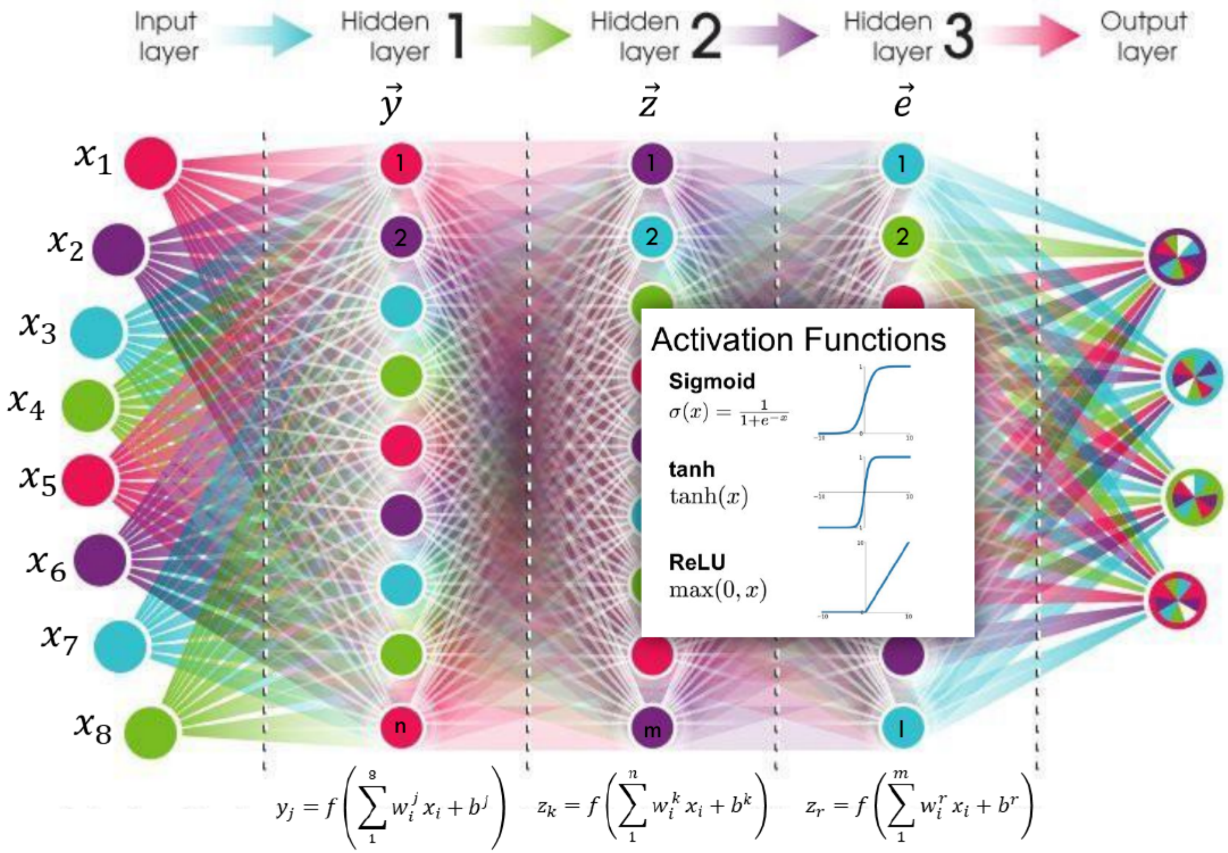


Typical Neural Network

1. Input layer
2. Hidden layer
3. Hidden layer
4. Hidden layer
5. .
6. .
7. .
8. .
9. .

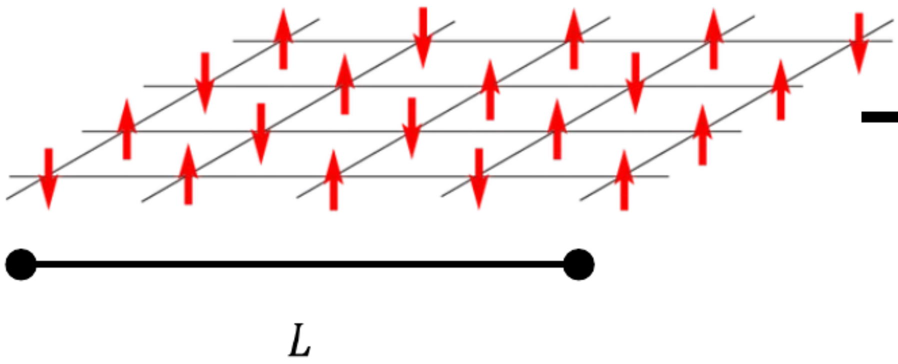
……..

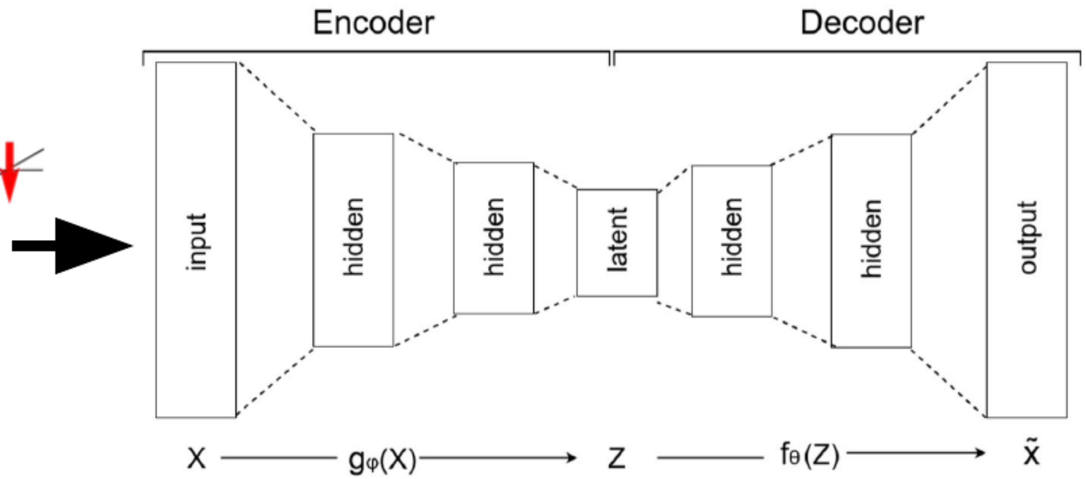
10. Output layer



Identify Phase transitions: Ising Model

$$H = -J \sum_{i,j=(nn)_i}^{N} s_i \, s_j \; - -\mu \, h \sum_{i=1}^{N} s_i$$

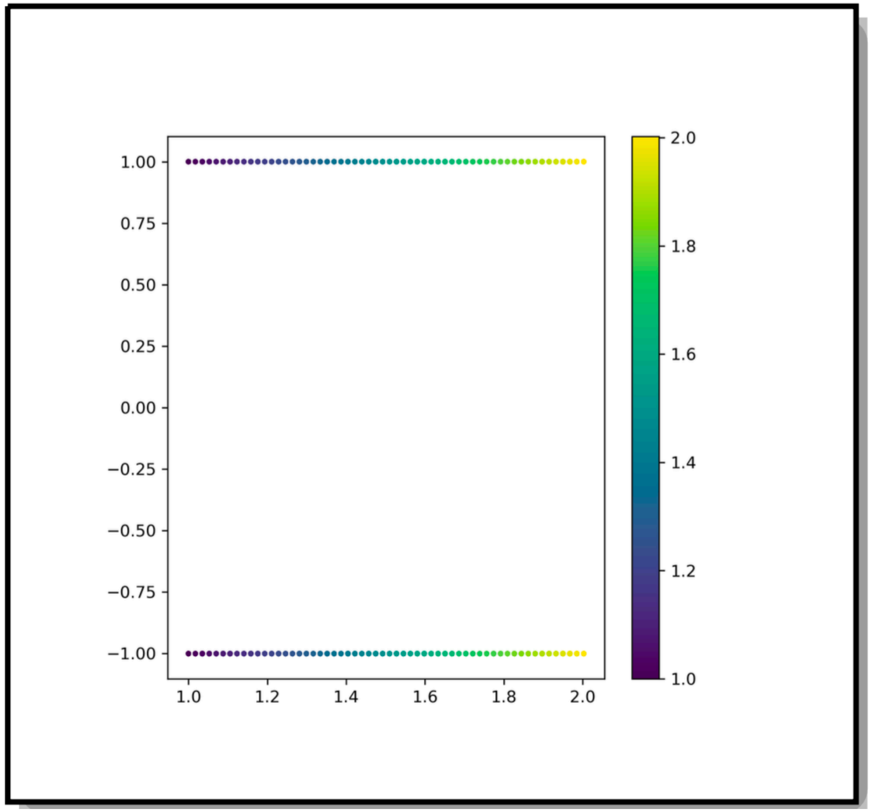




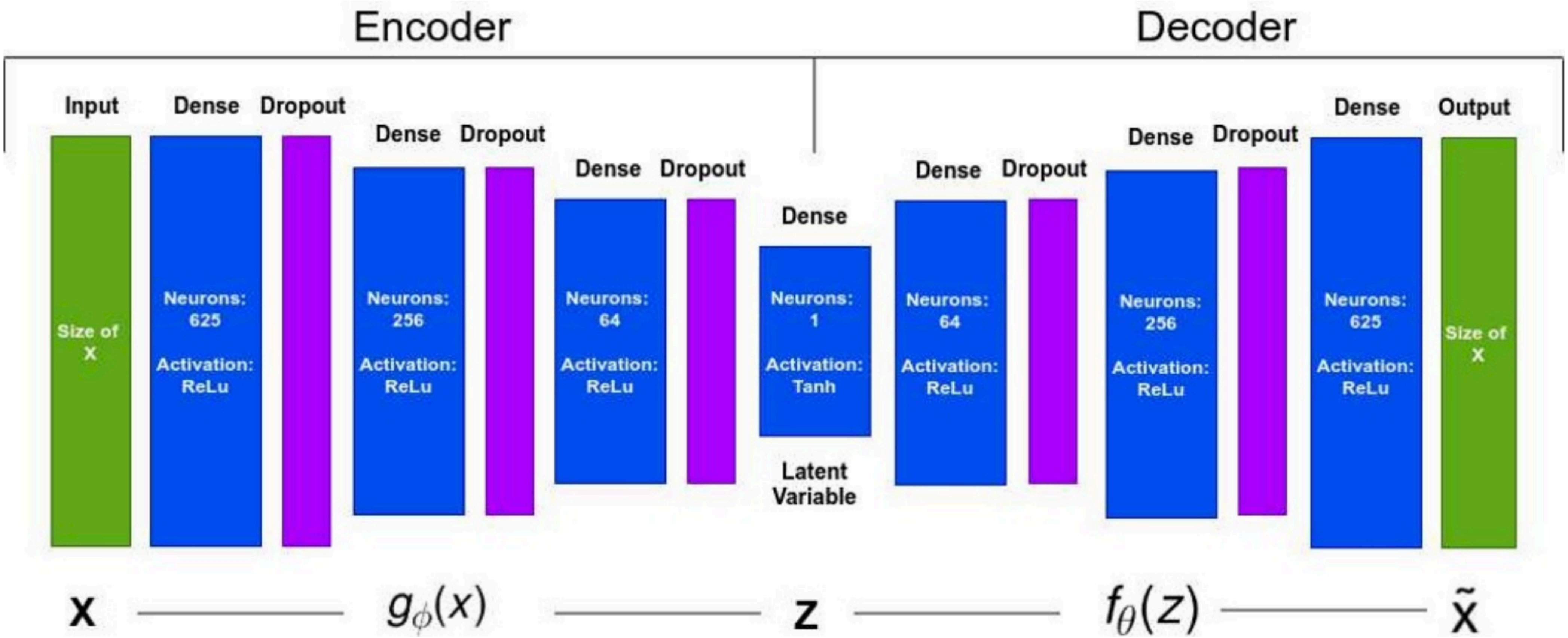
$$\mathrm{MSE}(\theta, \phi) = \frac{1}{n_{\text{data}}} \sum_{i=1} \left(X_i - f_\theta \left(g_\phi \left(X_i \right) \right) \right)^2$$

Can we do away with layers/Activation functions?

- Imagine 0 layers, 1 latent dimension. T = 1, 2.25, 4. Identity Activation function.
- Typical Latent dimension for T=1-2.

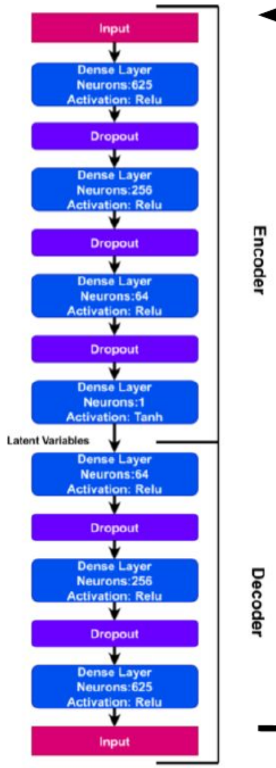


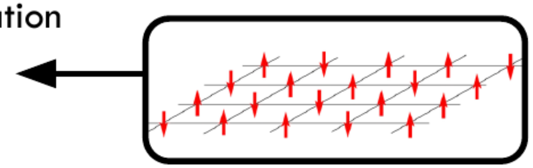
Ising model with Autoencoder



Ising model with Autoencoder

- Eight layers
- Fully connected (Dense)
- **Single** latent dimension
- Through experimentation



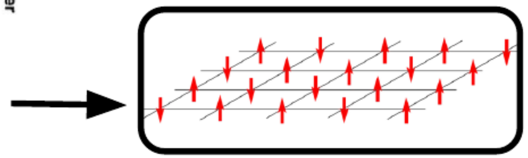


$$\text{relu} : y = max(0, x) = \begin{Bmatrix} x, \text{ if } x > 0 \\ 0 \quad \text{if } x \leq 0 \end{Bmatrix}$$

$$\tanh : y = \frac{1 - e^{-2x}}{1 + e^{-2x}}$$

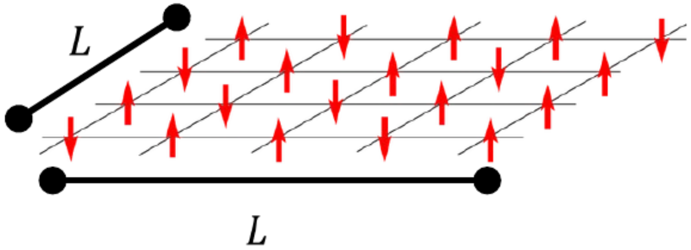
Dropout, reduces overfitting

G. E. Hinton et al, arXiv:1207.0580.



Ising model with Autoencoder

- Each configuration is re-expressed in the form of a vector:

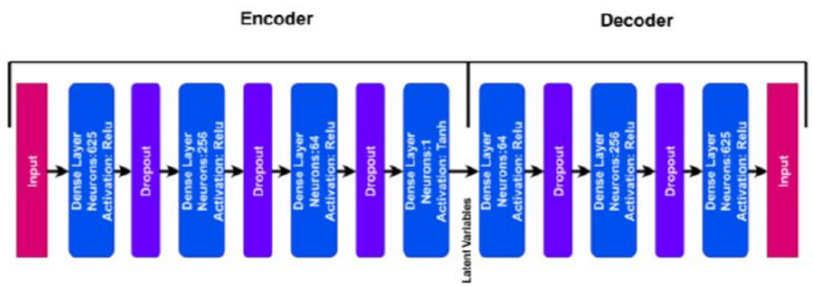


L^2

(-1, 1, -1,1,1, -1,1,-1,1,-1,1,1,-1,1,-1,1,1,1,1,1,-1)

Input into the encoder

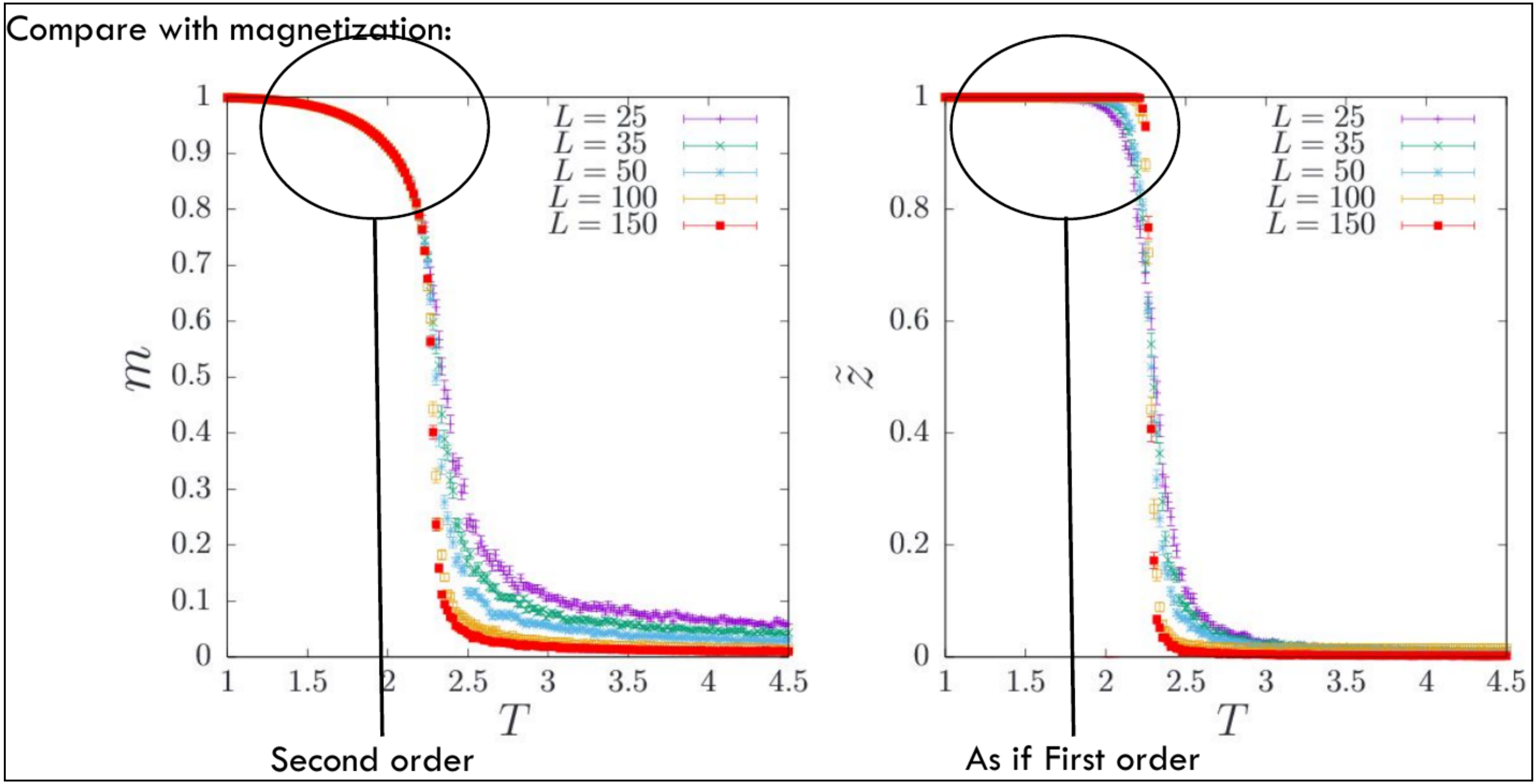
- In other words, each configuration is assigned a number, the latent dimension, which includes all the physically necessary information so that the decoder re-creates the actual configuration



$z_{i_{\mathrm{conf}}}$

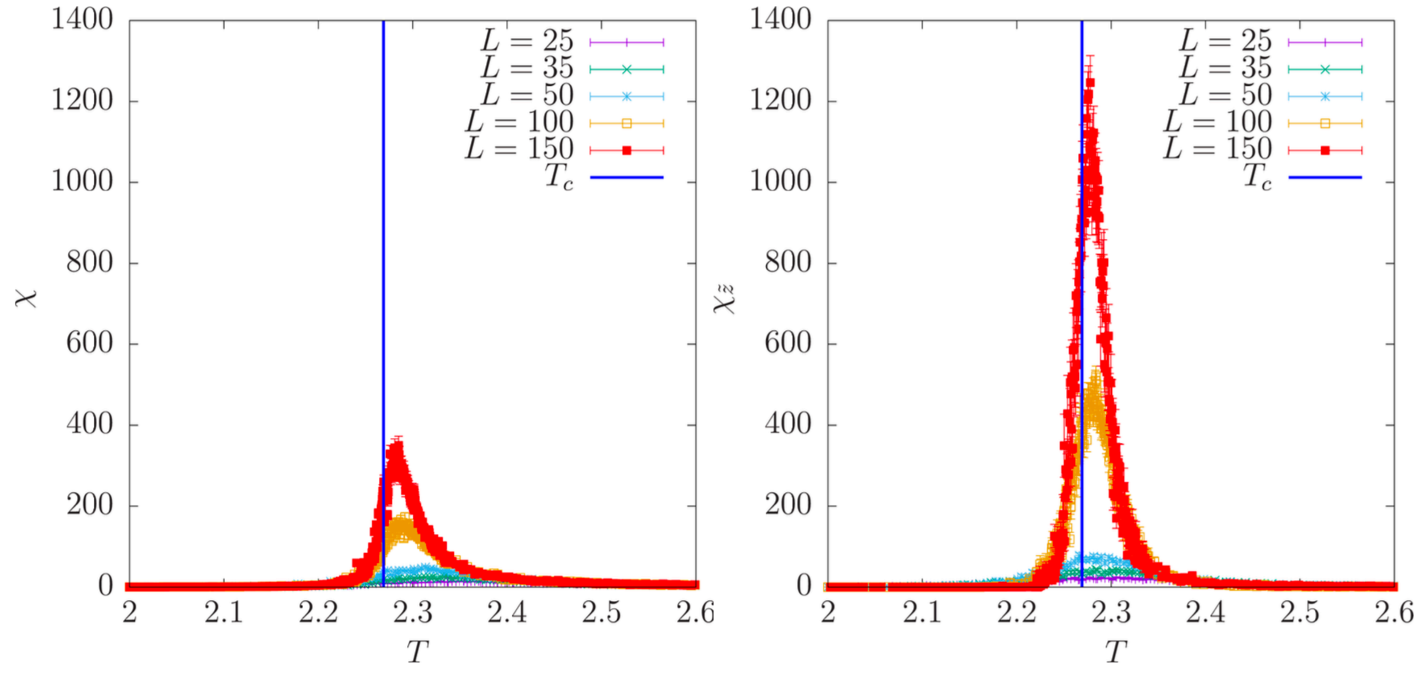
Latent Dimension

Ising model with Autoencoder

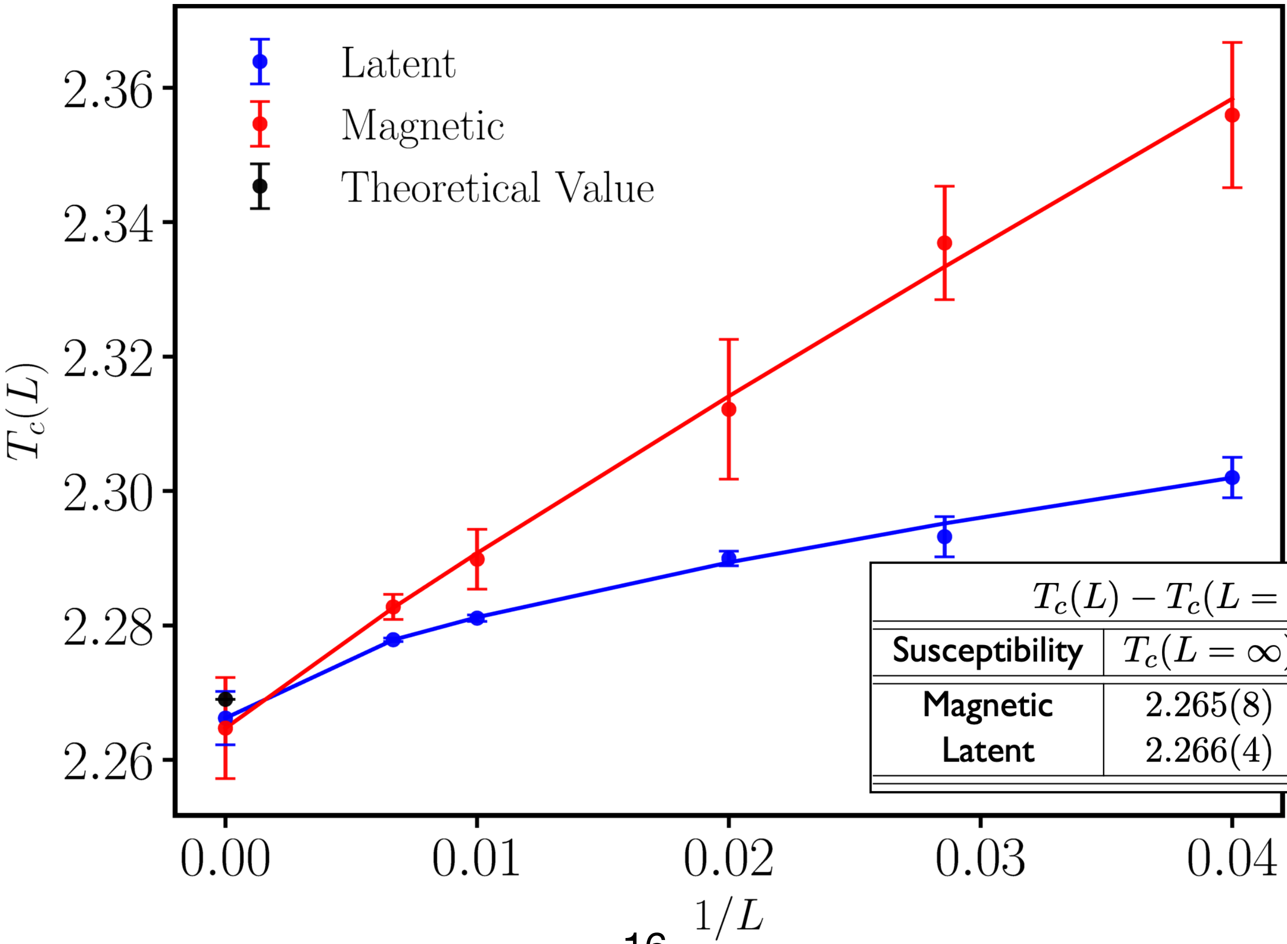


Susceptibility & Latent Susceptibility

$$\chi = \frac{L^2}{T}\left(\langle m^2 \rangle - \langle m \rangle^2\right) \qquad \chi_{\tilde{z}} = \frac{L^2}{T}\left(\langle \tilde{z}^2 \rangle - \langle \tilde{z} \rangle^2\right)$$



Ising model with Autoencoder

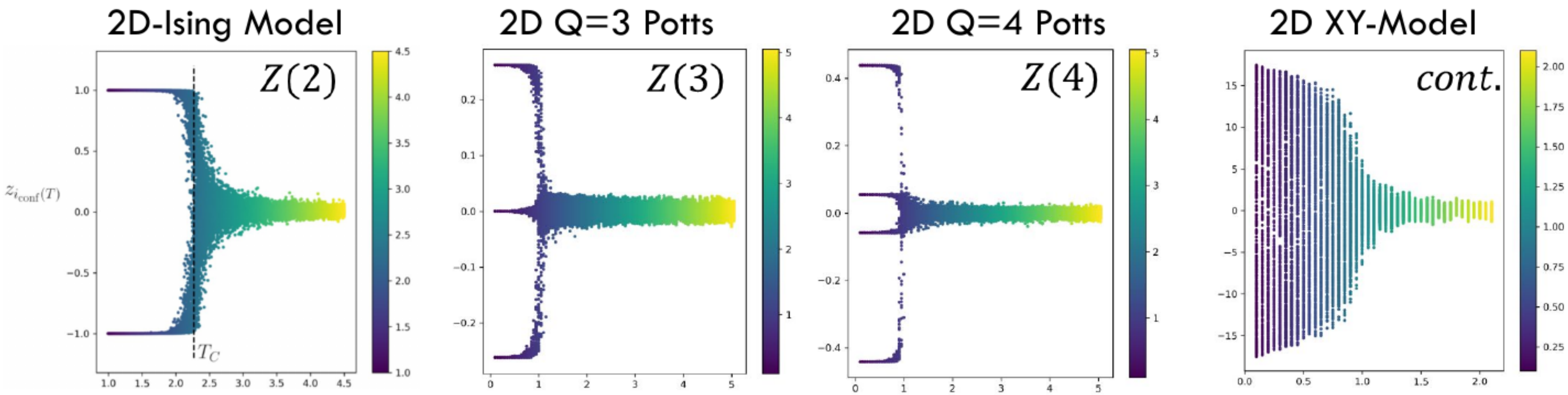


$T_c(L) - T_c(L = \infty) \propto L^{-1/\nu}$

Susceptibility	$T_c(L = \infty)$	ν	$\chi^2/$ dof
Magnetic	2.265(8)	1.08(20)	0.15
Latent	2.266(4)	1.60(14)	0.41

Autoencoder for identifying phase transitions

$$Z(G) = \{z \in G \mid \forall g \in G, zg = gz\}$$



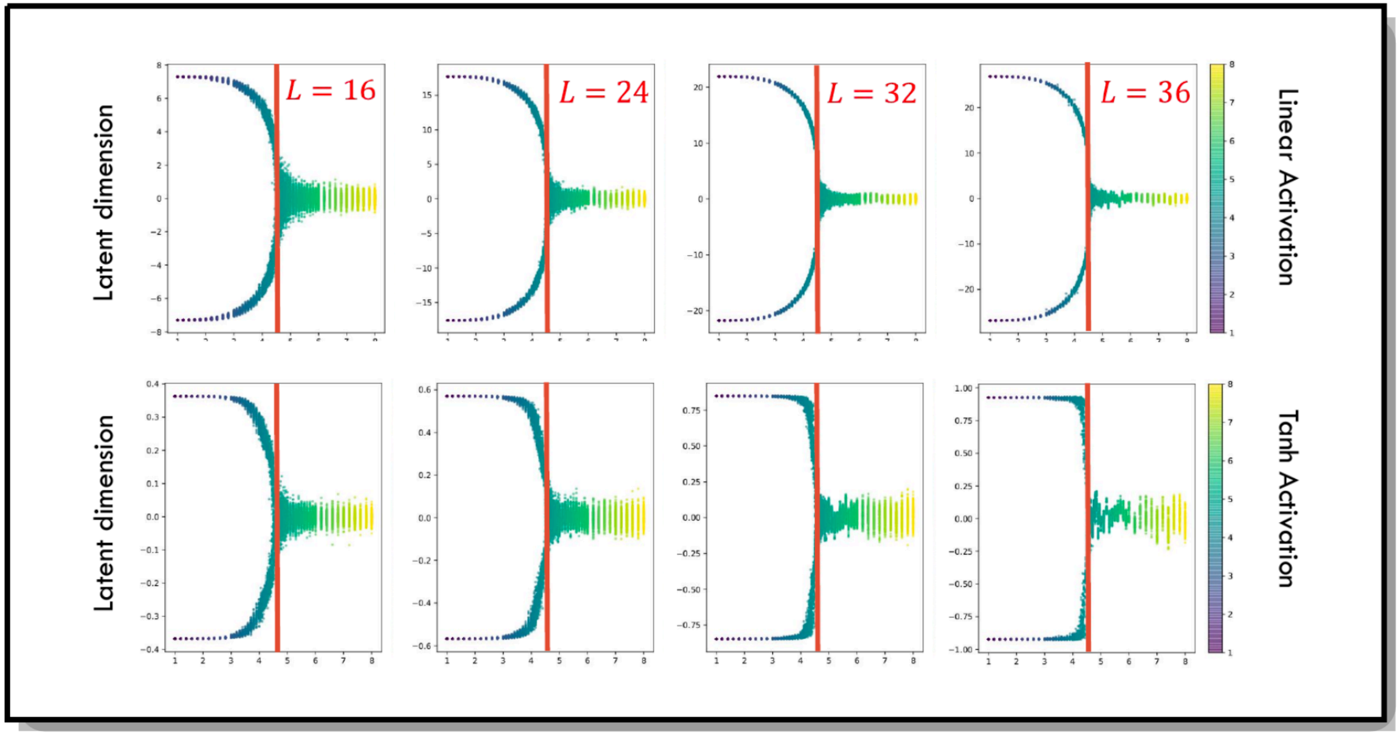
* Drawback

* Doesnt reproduce the correct physics, but can identify critical Temperature.

3D Ising Model

$T_C = 4.511$, Second order.

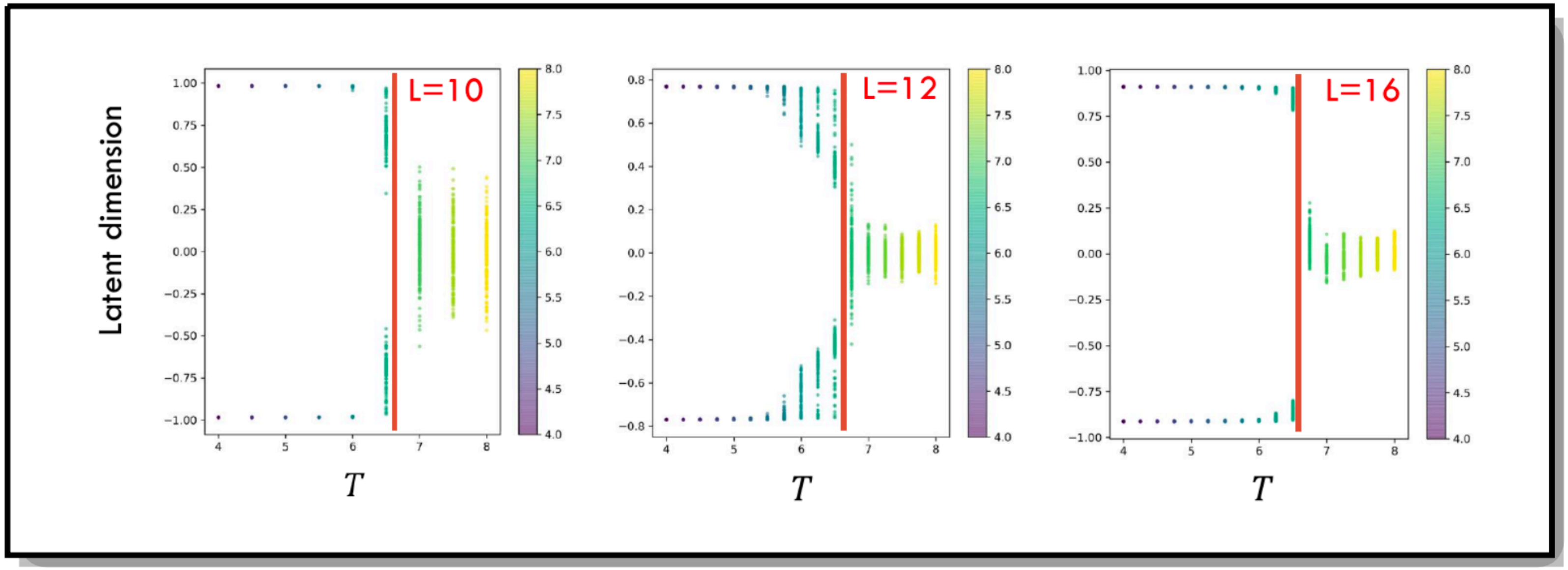
[*Talapov & Blöte 1996*]

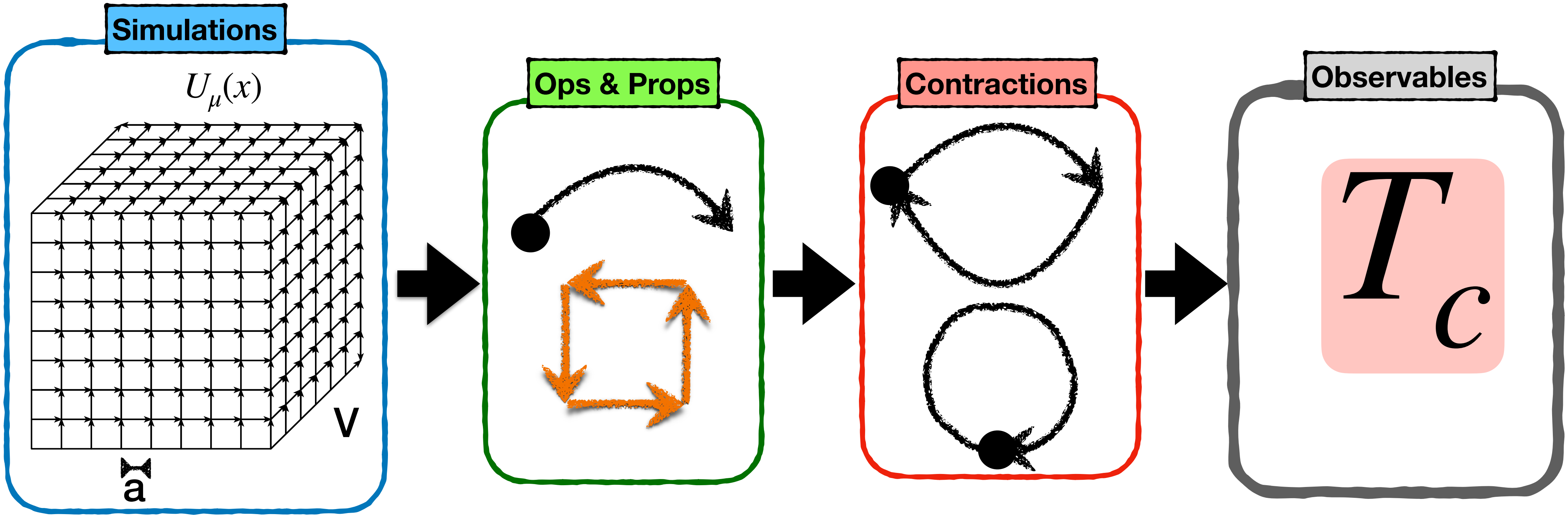


4D Ising Model

$T_C = 6.65.$

[Lundow & Markström 2012]





[Arxiv:1903.03506]

4D SU(3) Gauge theory, Polyakov Loop [Arxiv:2111.05216]

Supervised [Arxiv:1705.05582]

Control Variates

[Arxiv:2307.14950]

- If one define $\tilde{O} \equiv O - f$, where $\langle f \rangle = 0$,

$$\langle \tilde{O} \rangle = \langle O \rangle$$

- Its variance is

$$\mathrm{Var}(\tilde{O}) = \mathrm{Var}(O) + \langle f^2 \rangle - 2\langle Of \rangle$$

- $\mathrm{Var}(\tilde{O}) \leq \mathrm{Var}(O)$ if

$$\langle f^2 \rangle - 2\langle Of \rangle \leq 0$$

- Perfect control variates exist: $f_P = O - \langle O \rangle$

  It can be written as a gradient of a function.

Control Variates

- In general, it is hard to find observables with the expectation value zero.
- It was suggested to use lattice Schwinger-Dyson equation.

$$\int D\phi \frac{\delta}{\delta\phi}\left(g\, \mathrm{e}^{-S(\phi)}\right) = 0$$

- If $g : \mathbb{R}^n \to \mathbb{R}$,

$$f(\phi) = \sum_i \left(\frac{\partial g}{\partial \phi_i} - g \frac{\partial S}{\partial \phi_i} \right)$$

is a control variate with a proper boundary condition.

- Similarly, for $g : \mathbb{R}^n \to \mathbb{R}^n$, $f(\phi) = \nabla \cdot g - g \cdot \nabla S$

Control Variates

- Ansatz for control variates from the knowledge of free theory was suggested

$$g(\phi) = \sum_x a_x \phi_x + \cdots \quad \text{and} \quad f = \sum_i \left(\frac{\partial g}{\partial \phi_i} - g \frac{\partial S}{\partial \phi_i} \right)$$

- Instead of using educated guess, **parametrize g as a neural network**

$$g(\phi) = NN(\phi)$$

Control Variates , Imposing symmetry

- $f(T_x[\phi]) = f(\phi)$ should be imposed.

translation operator

$$\leftrightarrow g(T_y[\phi])_x = g[\phi]_{x+y} \text{ (covariance)}$$

- Define a function $g_0 : \mathbb{R}^n \to \mathbb{R}$

$$g(\phi)_x \equiv g_0(T_x[\phi]) \quad \text{and} \quad f(\phi) = \nabla \cdot g - g \cdot \nabla S$$

- It can be easily shown that $g(\phi)$ is translational covariant.

Control Variates: Minimize variance

- Natural choice of loss function is the variance:

$$L(w) = \langle (O - f)^2 \rangle - \langle O - f \rangle^2$$

neural network parameters

$= \langle O \rangle^2$

- However, if overfitting happens, $O(\phi_i) = f(\phi_i)$ for all training samples i,

very common in ML, dangerous

$$\bar{f} = \frac{1}{N} \sum_{i=1}^{N} f(\phi) = \overline{O} \neq 0$$

- Add tunable parameter μ to avoid overfitting:

not used for estimation

$$L(w, \mu) = \langle (O - f - \mu)^2 \rangle$$

Wilson loops in 2D U(1) lattice gauge theory

- Model:

$$Z = \int \prod_i dU_i \exp\left(-\beta \sum_i \left(1 - \cos(P_i)\right) \right)$$

where $\beta = \frac{2}{g^2}, P_i = U_1 U_2 U_3^\dagger U_4^\dagger$

- Observable: Wilson loop

$$O(A) = \prod_j P_j = \exp(i \sum_j \theta_j^P)$$

2D U(1) lattice gauge theory with open BC & Max.Gauge

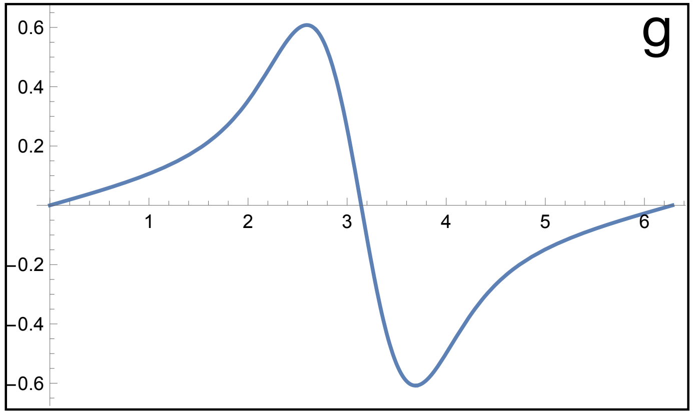
- Integration measure can be written in terms of plaquette variables, and action and observable are separable:

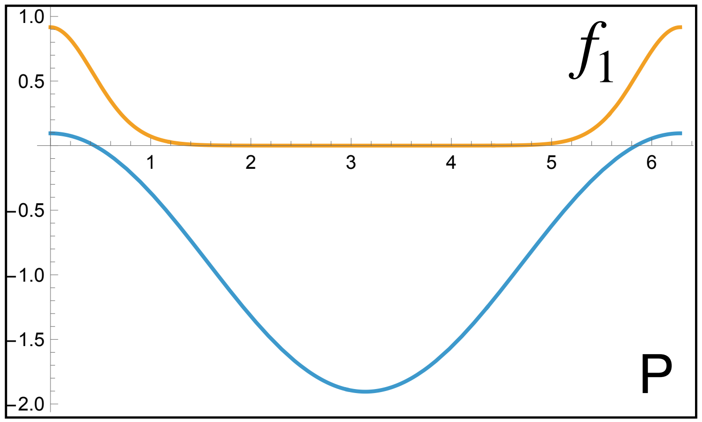
$$Z = \int \prod_i dP_i \exp\left(\beta \sum_i \cos(P_i)\right)$$

- Ansatz:

$$f(\phi) = \exp(i \sum_j \phi_j) - \prod_j \left(\exp(i\phi_j) - f_1(\phi_j)\right)$$

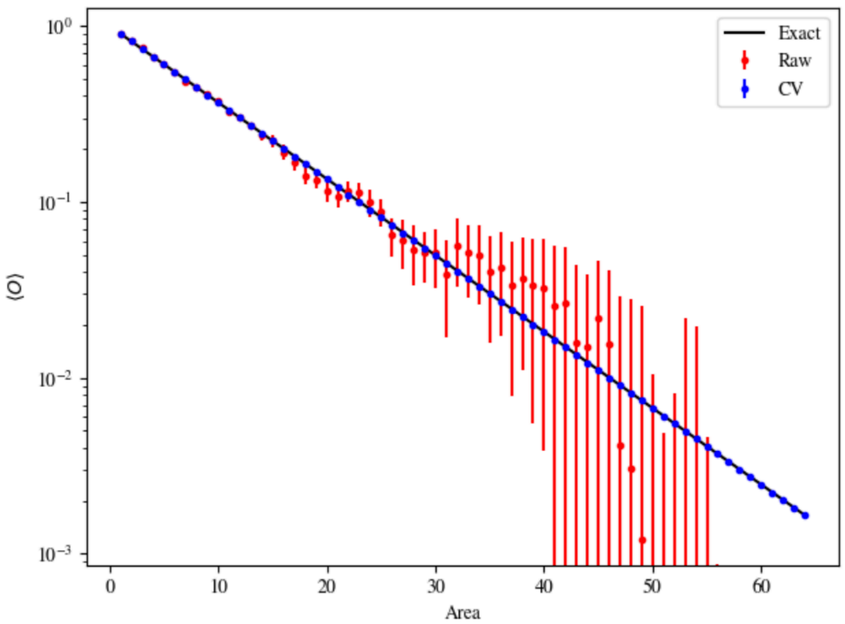
with $f_1(\phi) = \frac{dg}{d\phi} - g\frac{dS}{d\phi} \rightarrow \langle f_1 \rangle = 0 \Rightarrow \langle f \rangle = 0$

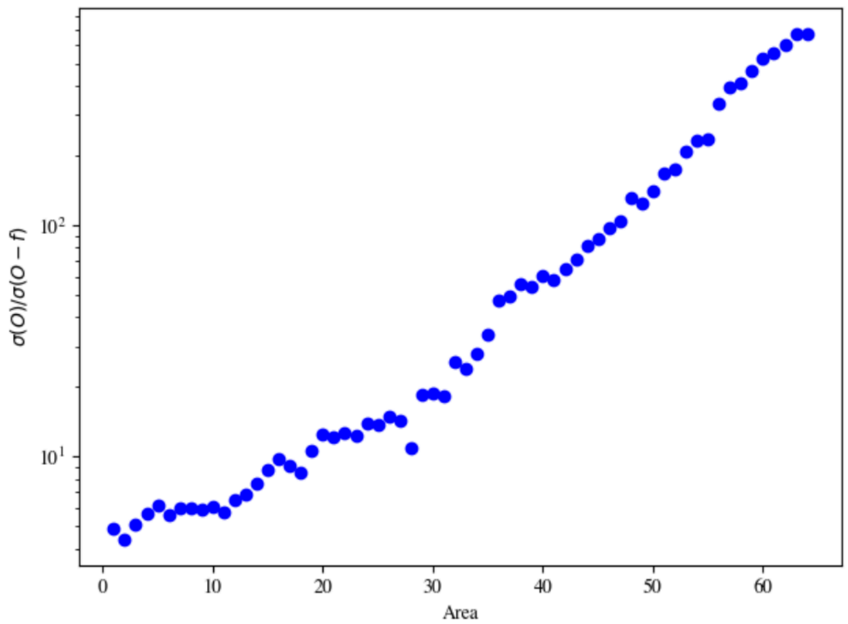




Wilson loops in 2D U(1) lattice gauge theory

- $\langle O(A) \rangle = \mu^A$ where $\mu \equiv \dfrac{I_1(\beta)}{I_0(\beta)}$

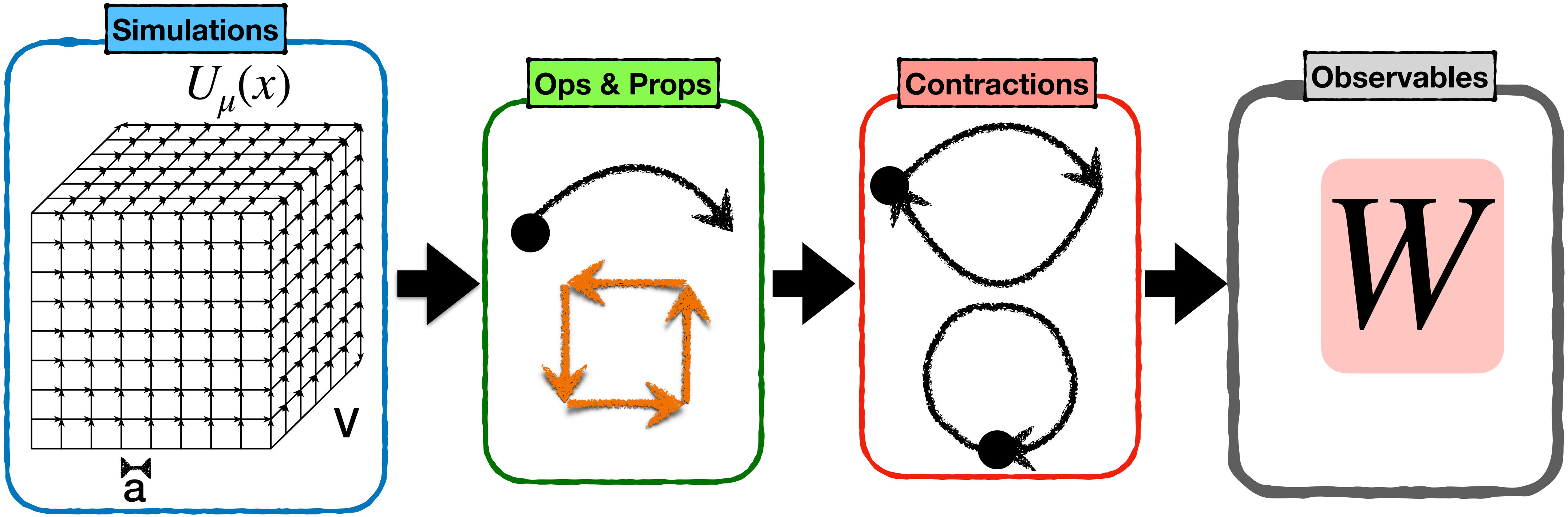




$\beta = 5.555$

Control Variates Outlook

* Control Variates existence guaranteed. Need more investigation.
* Volume scaling yet to be figured out. Should be cheaper than ensemble methods.
* Symmetries need to be encoded.



Contour deformation arxiv:2003.05914

Outlook

* Machine learning techniques are getting proof of principle verification.
* Need dedicated packages for lattice gauge theories.
* In the propagator measurements, significant input needed for preconditioners.
* Quantitative studies of phase transitions based on a synergistic relation between machine learning and statistical mechanics: explore new systems.
* ML targeted hardware helps lattice calculations.
* And if you want to learn automatic differentiation (heart of all ML). Check https://github.com/srijitpaul/iKAN

Safety