Machine Learning and Ising Model

Xiangdong Zeng (supervised by Ling-Yan Hung)

June 8, 2018

Department of Physics, Fudan University

Table of contents

Part 1. Ising model

Ising model

- "Binary" spins (+1 or −1) arranged in a lattice
- Simplest model for (anti)-ferromagnetism and phase transition
- Hamiltonian:

$$H(\{\sigma_i\}) = -\sum_{\langle ij\rangle} J_{ij}\sigma_i\sigma_j - \mu\sum_i B_i\sigma_i = -J\sum_{\langle ij\rangle}\sigma_i\sigma_j - \mu B\sum_i \sigma_i$$

- Probability of configuration $\{\sigma_i\}$:

$$P(\{\sigma_i\}) = \frac{e^{-\beta H(\{\sigma_i\})}}{Z_N}, \quad Z_N = \sum_{\{\sigma_i\}} e^{-\beta H(\{\sigma_i\})}$$

Critical behaviors and RG

- Universality: $\xi \to \infty$ at critical point
- Partition function is scale-invariant
  - Partition function:
$$Z_N = \sum_{\{\sigma_i\}} \mathrm{e}^{-\beta H(\{\sigma_i\}, \boldsymbol{K})}$$
  - Hamiltonian:
$$H(\{\sigma_i\}, \boldsymbol{K}) = -\sum_i K_i^{(1)} \sigma_i - \sum_{\langle ij \rangle} K_{ij}^{(2)} \sigma_i \sigma_j - \sum_{\langle ijk \rangle} K_{ijk}^{(3)} \sigma_i \sigma_j \sigma_k - \cdots$$
  - Scale transformation:
$$N' = L^{-d} N, \quad \xi' = L^{-1} \xi$$
  - Summation over Kadanoff's block $\{\sigma_i'\}$:
$$Z_{N'} = \sum_{\{\sigma_i'\}} \mathrm{e}^{-\beta H^{\mathrm{RG}}(\{\sigma_i'\}, \boldsymbol{K}')} = \sum_{\{\sigma_i'\}} \sum_{\{\sigma_i\}} \mathrm{e}^{-\beta \left[H(\{\sigma_i\}, \boldsymbol{K}) - \mathbf{T}_\lambda(\{\sigma_i\}, \{\sigma_i'\}) \right]}, \quad \boldsymbol{K}' = \mathbf{R}_L(\boldsymbol{K})$$

Part 2. Machine learning

Basic ideas

- Extract features from data, build models, make predictions or decisions with the help of computers
- Give machine the ability to “learn”
- Categories:
  - Supervised learning: regression, classification, etc
  - Unsupervised learning: clustering, dimensionality reduction, etc
- Main steps:
  1. Build a reasonable model based on data’s feature;
  2. Give the loss function;
  3. Train on the dataset (training set), find the parameters to minimize the loss function;
  4. Validate the model and parameters, fine tuning (validation set);
  5. Test the trained model on *new* dataset (test set).

Linear classifier (logistic regression)

- Hypothesis:

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{x})$$

- Activation function:

$$\sigma(z) = \frac{1}{1 + \mathrm{e}^{-z}}$$

- Loss function:

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{m} \log\left\{1 + \exp\left[-y^{(i)} h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right)\right]\right\}$$

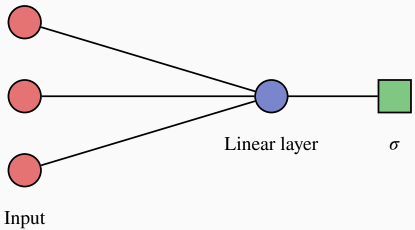


Figure 1: Network structure of the linear classifier

Neural network

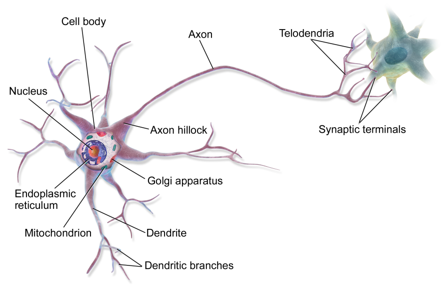


Figure 2: A multipolar neuron [1]

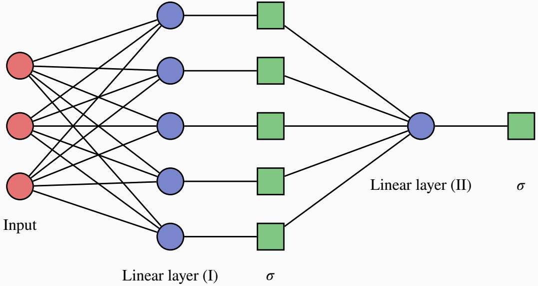


Figure 3: A two-layer neural network (perceptron)

[1]Source: `https://en.wikipedia.org/wiki/File:Blausen_0657_MultipolarNeuron.png`

Training results (linear model)

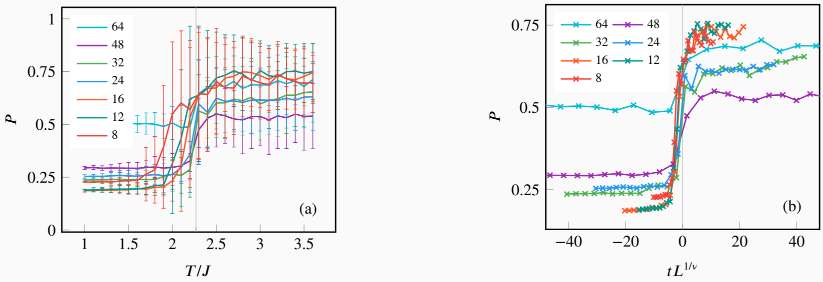


Figure 4: Predict results on Ising lattice. (a) Original; (b) after scale transformation.

Training results (neural network)

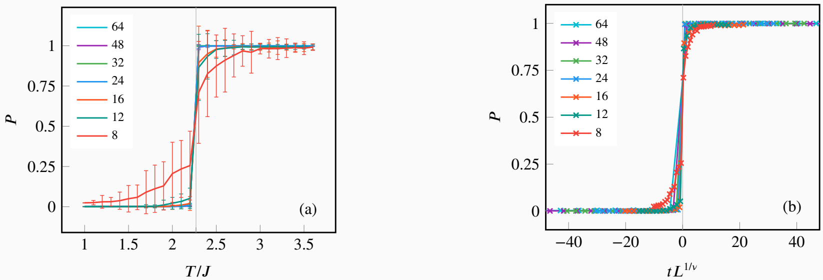


Figure 5: Predict results on Ising lattice. (a) Original; (b) after scale transformation.

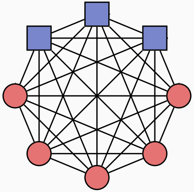
Part 3. Machine learning and RG

Energy-based model

- Boltzmann machine
  - Energy function:

$$\begin{aligned} E(\boldsymbol{s}) &= E(\{s_i\}) \\ &= -\sum_{i<j} W_{ij} s_i s_j - \sum_i \theta_i s_i \end{aligned}$$

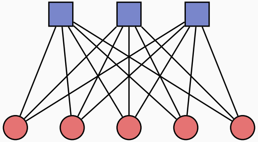
  - Structure:



- Restricted Boltzmann machine (RBM)
  - Energy function:

$$\begin{aligned} E(\boldsymbol{v}, \boldsymbol{h}) &= -\boldsymbol{v}^\top \boldsymbol{W} \boldsymbol{h} - \boldsymbol{b}^\top \boldsymbol{v} - \boldsymbol{c}^\top \boldsymbol{h} \\ &= -\sum_{i,j} W_{ij} v_i h_j - \sum_i b_i v_i - \sum_j c_j h_j \end{aligned}$$

  - Structure:



RBM trained on MNIST

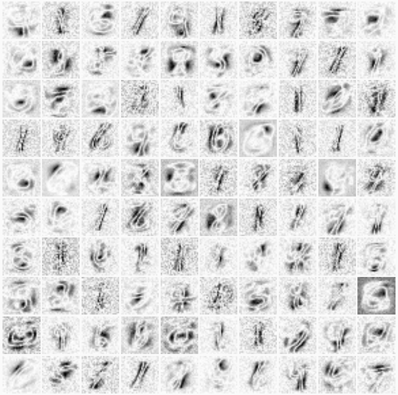


Figure 6: Weight matrix (reshaped)

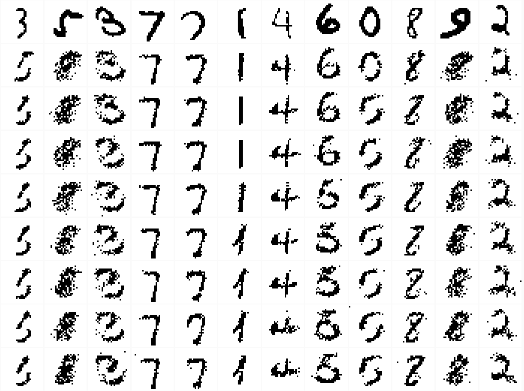


Figure 7: Reconstructed images

RBM trained on Ising lattice

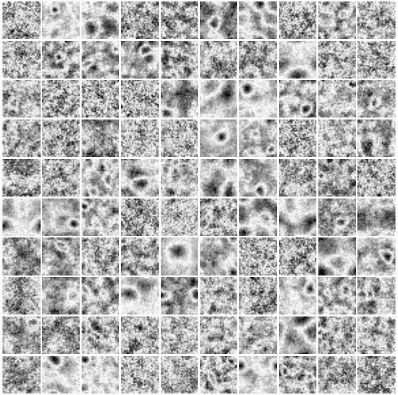


Figure 8: Weight matrix (reshaped)

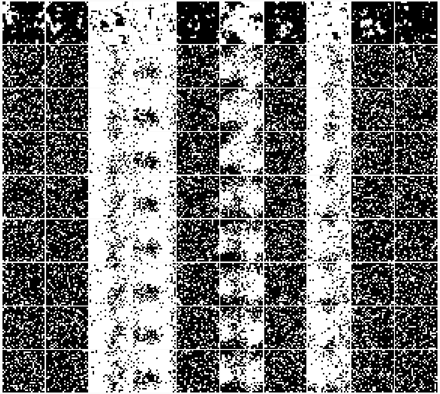


Figure 9: Reconstructed images

Exact mapping between RBM and RG

- Partition function under coarse graining:

$$Z_{N'} = \sum_{\{\sigma_i'\}} \mathrm{e}^{-\beta H^{\mathrm{RG}}(\{\sigma_i'\}, \boldsymbol{K}')} = \sum_{\{\sigma_i'\}} \sum_{\{\sigma_i\}} \mathrm{e}^{-\beta \left[H(\{\sigma_i\}, \boldsymbol{K}) - \mathbf{T}_\lambda(\{\sigma_i\}, \{\sigma_i'\}) \right]}$$

- Express in RBM:

$$\mathrm{e}^{-H^{\mathrm{RG}}(\boldsymbol{h})} = \sum_{\boldsymbol{v}} \mathrm{e}^{\mathbf{T}(\boldsymbol{v}, \boldsymbol{h}) - H(\boldsymbol{v})}$$

- Let $\mathbf{T}(\boldsymbol{v}, \boldsymbol{h}) = H(\boldsymbol{v}) - E(\boldsymbol{v}, \boldsymbol{h})$, then

$$\frac{1}{Z} \mathrm{e}^{-H^{\mathrm{RG}}(\boldsymbol{h})} = \frac{1}{Z} \sum_{\boldsymbol{v}} \mathrm{e}^{-E(\boldsymbol{v}, \boldsymbol{h})} = P(\boldsymbol{h}) = \frac{1}{Z} \mathrm{e}^{-H^{\mathrm{RBM}}(\boldsymbol{h})} \implies H^{\mathrm{RG}}(\boldsymbol{h}) = H^{\mathrm{RBM}}(\boldsymbol{h})$$

Convolution and wavelet transform

- Convolution:

$$(f * g)(t) = \int_{-\infty}^{+\infty} f(\tau)\, g(t - \tau)\, \mathrm{d}\tau$$

- A kind of “weighted average”
- Expand function with *daughter wavelet*:

$$\psi_{\boldsymbol{a},s}(\boldsymbol{x}) = \frac{1}{s^{d/2}} \psi\left(\frac{\boldsymbol{x} - \boldsymbol{a}}{s}\right)$$

- Wavelet transform:

$$W_f(\boldsymbol{a}, s) = \int \mathrm{d}^d x\, f(\boldsymbol{x}) \psi^\dagger_{\boldsymbol{a},s}(\boldsymbol{x})$$

AdS/CFT and wavelet transform

- Reconstruct fields in the bulk from boundary operators:

$$\phi^i(\boldsymbol{x}, z) = \int \mathrm{d}^d y\, K_i(\boldsymbol{x}, z|\boldsymbol{y}) O^i(\boldsymbol{y}) + \sum_{j,k} \frac{\lambda^i_{jk}}{N} \int \mathrm{d}^d x'\, \mathrm{d}z'\, G_i(\boldsymbol{x}, z|\boldsymbol{x}', z') \int \mathrm{d}^d y_1\, K_j(\boldsymbol{x}', z'|\boldsymbol{y}_1) O^j(\boldsymbol{y}_1) \int \mathrm{d}^d y_2\, K_k(\boldsymbol{x}', z'|\boldsymbol{y}_2) O^k(\boldsymbol{y}_2) + \cdots$$

- Boundary–bulk kernel:

$$K_i(\boldsymbol{x}, z|\boldsymbol{y}) = \left[\frac{z}{z^2 - \|\boldsymbol{x} - \boldsymbol{y}\|^2}\right]^{d-\Delta_i} \Theta\left(z^2 - \|\boldsymbol{x} - \boldsymbol{y}\|^2\right)$$

- Mother and daughter wavelets:

$$\psi_\Delta(\boldsymbol{x}) = \left(\frac{1}{1 - \|\boldsymbol{x}\|^2}\right)^{d-\Delta} \Theta\left(1 - \|\boldsymbol{x}\|^2\right), \quad \psi_{\boldsymbol{a},s}(\boldsymbol{x}) = \frac{1}{z^{d/2}} \psi\left(\frac{\boldsymbol{y} - \boldsymbol{x}}{z}\right)$$

- Then

$$K(\boldsymbol{x}, z|\boldsymbol{y}) = z^{\Delta - d/2} \psi_{\boldsymbol{x},z}(\boldsymbol{y})$$

Convolutional RBM (CRBM)

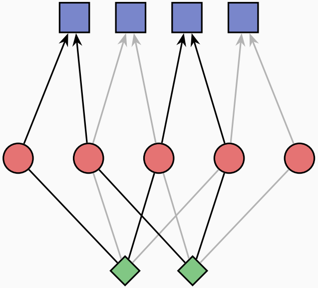
- Linear to convolutional:

$$h_j + c_j \sim \sum_i W_{ij}(v_i + b_i) \to h_{\boldsymbol{j}} + c_{\boldsymbol{j}} \sim \sum_{\boldsymbol{i}} W_{\boldsymbol{i}}(v_{\boldsymbol{i+j}} + b_{\boldsymbol{i+j}})$$

- Energy function:

$$E(\boldsymbol{v}, \boldsymbol{h}) = -\sum_{\boldsymbol{i},\boldsymbol{j}} W_{\boldsymbol{i}} v_{\boldsymbol{i+j}} h_{\boldsymbol{j}} - \sum_{\boldsymbol{i}} b_{\boldsymbol{i}} v_{\boldsymbol{i}} - \sum_{\boldsymbol{j}} c_{\boldsymbol{j}} h_{\boldsymbol{j}}$$

- Our conjecture: filter $\boldsymbol{W}$ in CRBM is the Green's function K in AdS/CFT



Conclusion

Conclusion

- Supervised learning can be used to classify phases of Ising model
- RBM can be considered as an implementation of renormalization
- Convolution and wavelet transform that root in AdS/CFT can be mapped to machine learning (e.g. CRBM)

Thank you!

Exact solutions

- One dimension: no phase transition
- Two dimension
  - Critical temperature:

$$\frac{T_c}{J} = \frac{2}{\ln\left(1+\sqrt{2}\right)} \approx 2.2692$$

  - Heat capacity (logarithmic divergence):

$$\frac{C}{N} \simeq -0.4945 \ln \left|1 - \frac{T}{T_c}\right| + \text{const}$$

  - Spontaneous magnetization:

$$\frac{\bar{M}}{N\mu} = \begin{cases} \left(1 - \sinh^{-4} 2\beta J\right)^{1/8}, & T < T_c \\ 0, & T > T_c \end{cases}$$

Monte Carlo simulation

- Basic idea: average over samples instead of all configurations
- Distribution evolution:

$$
\begin{aligned}
P\big(\{\sigma_i\}, t+1\big) = P\big(\{\sigma_i\}, t\big) &+ \sum_{\{\sigma_i'\}} P\big(\{\sigma_i'\}, t\big) W\big(\{\sigma_i'\} \to \{\sigma_i\}\big) \\
&- \sum_{\{\sigma_i'\}} P\big(\{\sigma_i\}, t\big) W\big(\{\sigma_i\} \to \{\sigma_i'\}\big)
\end{aligned}
$$

- Detailed balance condition:

$$
P_{\mathrm{eq}}(\{\sigma_i\}) W\big(\{\sigma_i\} \to \{\sigma_i'\}\big) = P_{\mathrm{eq}}(\{\sigma_i'\}) W\big(\{\sigma_i'\} \to \{\sigma_i\}\big)
$$

- Metropolis transition rates:

$$
W\big(\{\sigma_i\} \to \{\sigma_i'\}\big) = \begin{cases} 1, & \Delta E \leqslant 0 \\ \mathrm{e}^{-\beta \Delta E}, & \Delta E > 0 \end{cases}
$$

Algorithm

1. Initialize all the spins: set spins to be 0 or 1 *randomly*.
2. Generate a trial state $\{\sigma_i'\}$ that is near the current state $\{\sigma_i\}$
    - Usually we can flip a single spin.
3. Calculate ΔE and accept probability P. If random number $\xi < P$, then flip the corresponding spin.
4. Perform step 2 and 3 for all the lattice (one Monte Carlo step, MCS).
    - To maintain detailed balance, we should choose the spin *randomly*.
    - Flipping one-by-one is also acceptable for efficiency.
5. Repeat step 2–4 for some time, then we can measure the thermodynamic observables and calculate their mean values and standard errors.

Simulation results (1)

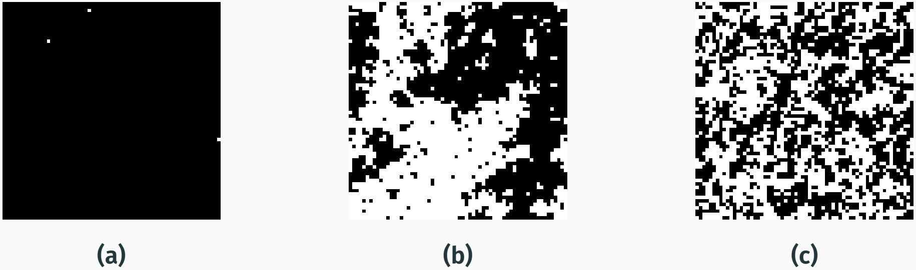


Figure 10: 64×64 Ising lattice at (a) $T_c/J = 1.0$; (b) $T_c/J = 2.3$; (c) $T_c/J = 3.6$.

Simulation results (2)

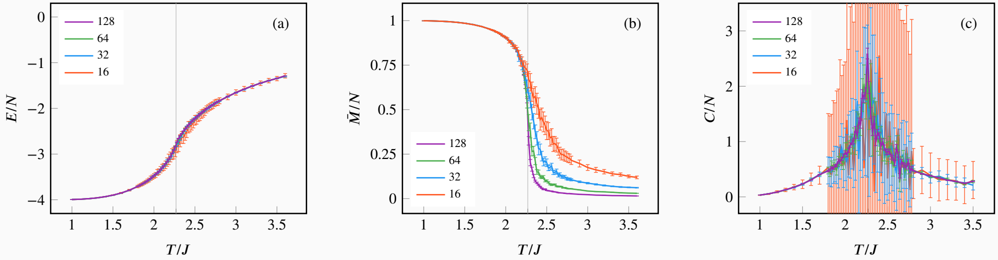


Figure 11: (a) Energy; (b) spontaneous magnetization; (c) heat capacity. Sampling density is increased near T_c. Error bar means one σ.

Linear regression

- Hypothesis:

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{j=0}^{n} \theta_j x_j = \boldsymbol{\theta}^{\mathsf{T}} \boldsymbol{x}$$

- Loss function:

$$L(\boldsymbol{\theta}) = \frac{1}{2} \sum_{i=1}^{m} \left[y^{(i)} - h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) \right]^2$$

- Exact solution:

$$\hat{\boldsymbol{\theta}} = \left(\boldsymbol{X}^{\mathsf{T}} \boldsymbol{X}\right)^{-1} \boldsymbol{X}^{\mathsf{T}} \boldsymbol{y}$$

- Not realistic with large dataset — gradient descent

$$\boldsymbol{\theta} \coloneqq \boldsymbol{\theta} - \alpha \nabla \boldsymbol{\theta}$$

Neural network — details

- Loss function (mean squared error):

$$L = \frac{1}{2}\sum_{i=1}^{m}\left[y^{(i)} - \hat{y}^{(i)}\right]^2$$

- Loss function (cross-entropy):

$$L = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\ln\hat{y}^{(i)} - \left(1 - y^{(i)}\right)\ln\left(1 - \hat{y}^{(i)}\right)\right]^2$$

- Training algorithm: backpropagation

- Regularization
  - Prevent overfitting
  - A kind of "penalty": force the parameters to be sparse
  - Weight decay (L^2 regularization):

$$\frac{\lambda}{2m}\sum_{k}\left\|\boldsymbol{\theta}_k\right\|^2$$

CD-k algorithm (1)

- Loss function:

$$L(\boldsymbol{\theta}) = -\frac{1}{|D|} \sum_{\boldsymbol{x}^{(i)} \in D} \ln P\left(\boldsymbol{x}^{(i)}\right)$$

- Gradient:

$$-\frac{\partial}{\partial \boldsymbol{\theta}} \ln P\left(\boldsymbol{x}^{(i)}\right) = -\frac{\partial}{\partial \boldsymbol{\theta}} \ln P\left(\boldsymbol{v}^{(i)}\right) = \frac{\partial F(\boldsymbol{v}^{(i)})}{\partial \boldsymbol{\theta}} - \sum_{\boldsymbol{v}} P(\boldsymbol{v}) \frac{\partial F(\boldsymbol{v})}{\partial \boldsymbol{\theta}}$$

  where F is the free energy

$$F(\boldsymbol{v}) = -\ln \sum_{\boldsymbol{h}} \mathrm{e}^{-E(\boldsymbol{v}, \boldsymbol{h})}$$

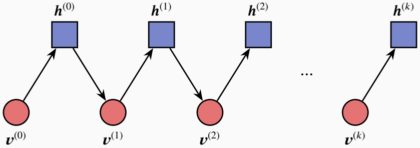
- Positive phase and negative phase (difficult)

CD-k algorithm (2)

- Monte Carlo again:

$$\sum_{\boldsymbol{v}} P(\boldsymbol{v}) \frac{\partial F(\boldsymbol{v})}{\partial \boldsymbol{\theta}} \approx \frac{1}{|N_s|} \sum_{\boldsymbol{v} \in N_s} \frac{\partial F(\boldsymbol{v})}{\partial \boldsymbol{\theta}}$$

- Gibbs sampling:



- Gradient:

$$-\frac{\partial}{\partial \boldsymbol{\theta}} \ln P\left(\boldsymbol{v}^{(i)}\right) \approx \frac{\partial F(\boldsymbol{v}^{(i)})}{\partial \boldsymbol{\theta}} - \frac{\partial F(\boldsymbol{v}'^{(i)})}{\partial \boldsymbol{\theta}}$$

- Update parameters:

$$\begin{cases} \boldsymbol{W} \leftarrow \boldsymbol{W} - \alpha \left(\boldsymbol{v}\boldsymbol{h}^{\mathsf{T}} - \boldsymbol{v}'\boldsymbol{h}'^{\mathsf{T}} \right) \\ \boldsymbol{b} \leftarrow \boldsymbol{b} - \alpha \left(\boldsymbol{v} - \boldsymbol{v}' \right) \\ \boldsymbol{c} \leftarrow \boldsymbol{c} - \alpha \left(\boldsymbol{h} - \boldsymbol{h}' \right) \end{cases}$$

Basic of AdS/CFT

- Anti-de Sitter space:

$$ds^2 = \sum_i dx_i{}^2 - \sum_j dt_j{}^2$$

- Conformal field theory: QFT with conformal symmetry
- Ising model: one of a minimal model in 2D CFT
- Holographic duality:

$$\text{quantum gravity in } M \simeq \text{a QFT in } \partial M$$

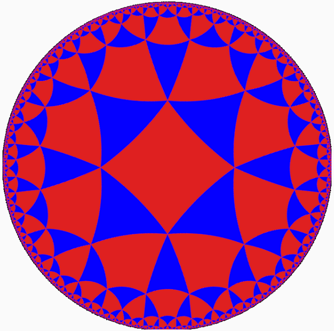


Figure 12: Hyperbolic plane [2]

[2] Source: https://en.wikipedia.org/wiki/File:Uniform_tiling_433-t0.png