大自然的计算：从伊辛模型到生成模型

Ill. Niklas Elmehed © Nobel Prize Outreach

John J. Hopfield

Ill. Niklas Elmehed © Nobel Prize Outreach

Geoffrey E. Hinton

Pan Zhang
ITP,CAS

The Nobel Prize in Physics 2024

The Royal Swedish Academy of Sciences has decided to award the Nobel Prize in Physics 2024 to

John J. Hopfield
Princeton University, NJ, USA

Geoffrey E. Hinton
University of Toronto, Canada

"for foundational discoveries and inventions that enable machine learning with artificial neural networks"

They trained artificial neural networks using physics

This year's two Nobel Laureates in Physics have used tools from physics to develop methods that are the foundation of today's powerful machine learning. John Hopfield created an associative memory that can store and reconstruct images and other types of patterns in data. Geoffrey Hinton invented a method that can autonomously find properties in data, and so perform tasks such as identifying specific elements in pictures.

When we talk about artificial intelligence, we often mean machine learning using artificial neural networks. This technology was originally inspired by the structure of the brain. In an artificial neural network, the brain's neurons are represented by nodes that have different values. These nodes influence each other through connections that can be likened to synapses and which can be made stronger or weaker. The network is *trained*, for example by developing stronger connections between nodes with simultaneously high values. This year's laureates have conducted important work with artificial neural networks from the 1980s onward.

John Hopfield invented a network that uses a method for saving and recreating patterns. We can imagine the nodes as pixels. The *Hopfield network* utilises physics that describes a material's characteristics due to its atomic spin – a property that makes each atom a tiny magnet. The network as a whole is described in a manner equivalent to the energy in the spin system found in physics, and is trained by finding values for the connections between the nodes so that the saved images have low energy. When the Hopfield network is fed a distorted or incomplete image, it methodically works through the nodes and updates their values so the network's energy falls. The network thus works stepwise to find the saved image that is most like the imperfect one it was fed with.

Hopfield network

Geoffrey Hinton used the Hopfield network as the foundation for a new network that uses a different method: the *Boltzmann machine*. This can learn to recognise characteristic elements in a given type of data. Hinton used tools from statistical physics, the science of systems built from many similar components. The machine is trained by feeding it examples that are very likely to arise when the machine is run. The Boltzmann machine can be used to classify images or create new examples of the type of pattern on which it was trained. Hinton has built upon this work, helping initiate the current explosive development of machine learning.

Boltzmann machine

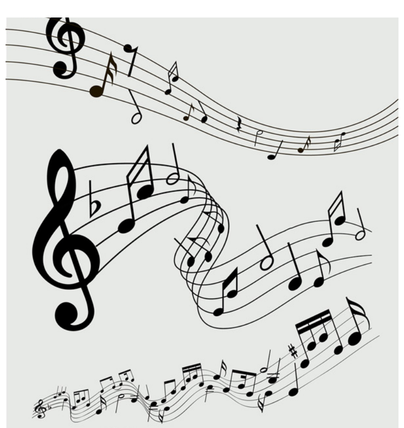
"The laureates' work has already been of the greatest benefit. In physics we use artificial neural networks in a vast range of areas, such as developing new materials with specific properties," says Ellen Moons, Chair of the Nobel Committee for Physics.

John J. Hopfield, born 1933 in Chicago, IL, USA. PhD 1958 from Cornell University, Ithaca, NY, USA. Professor at Princeton University, NJ, USA.

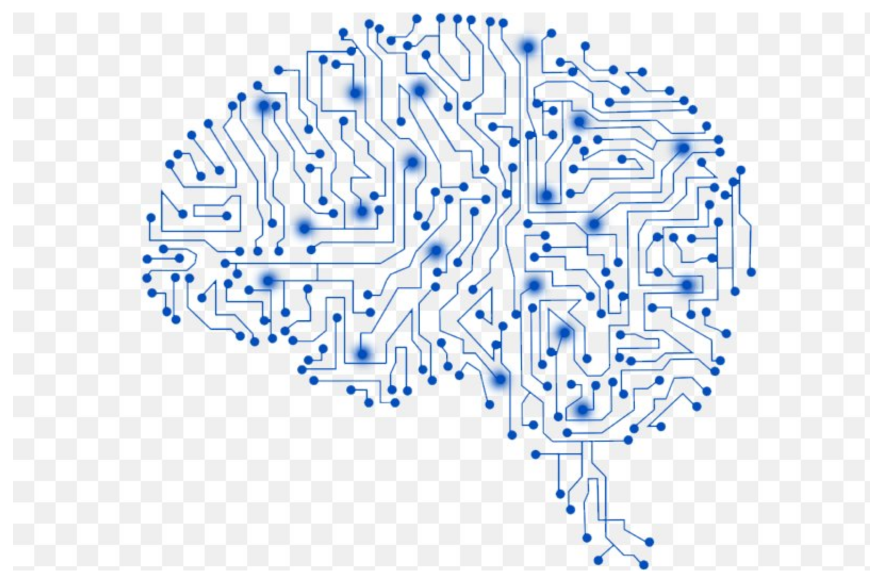
Geoffrey E. Hinton, born 1947 in London, UK. PhD 1978 from The University of Edinburgh, UK. Professor at University of Toronto, Canada.

无监督学习（生成学习）

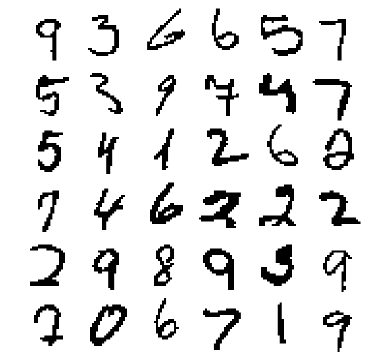
床前明月光，
疑是地上霜。
举头望明月，
低头思故乡。



训练



生成
（采样）



千山鸟飞绝，
万径人踪灭。
孤舟蓑笠翁，
独钓寒江雪。

$P(X)$

↑↑↑↓↑↓↓↓↓↓↑↑

↑↑↑↓↑↓↓↓↓↓↑↑

The Forward-Forward Algorithm: Some Preliminary Investigations

Geoffrey Hinton
Google Brain
geoffhinton@google.com

2022年NeurIPS
Hinton 75岁

Abstract

The aim of this paper is to introduce a new learning procedure for neural networks
and to demonstrate that it works well enough on a few small problems to be worth

In the early 1980s there were two promising learning procedures for deep neural networks. One was backpropagation and the other was Boltzmann Machines (Hinton and Sejnowski, 1986) which performed unsupervised contrastive learning. A Boltzmann Machine is a network of stochastic binary

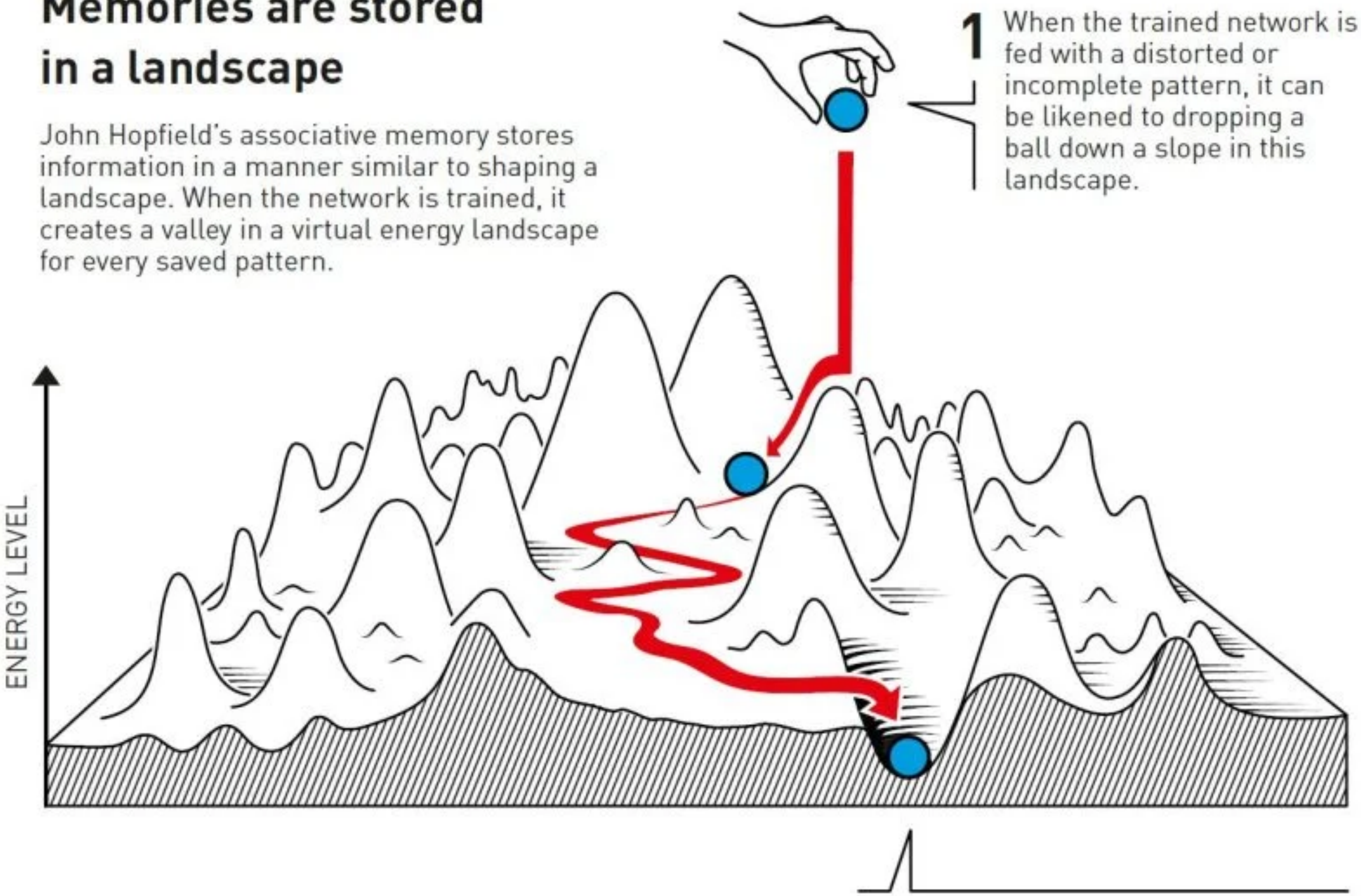
The Boltzmann machine can be seen as a combination of two ideas:

1. Learn by minimizing the free energy on real data and maximizing the free energy on negative data generated by the network itself.
2. Use the Hopfield energy as the energy function and use repeated stochastic updates to sample global configurations from the Boltzmann distribution defined by the energy function.

来自物理的启发

Memories are stored in a landscape

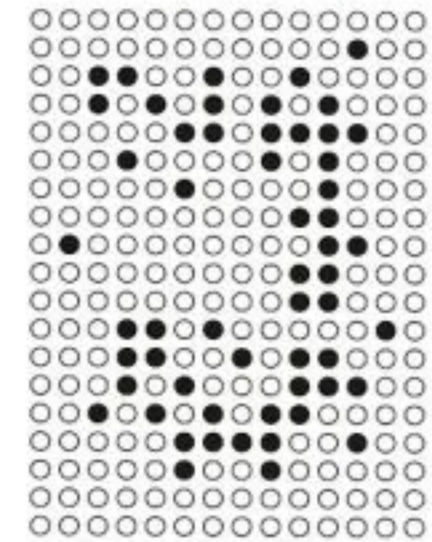
John Hopfield's associative memory stores information in a manner similar to shaping a landscape. When the network is trained, it creates a valley in a virtual energy landscape for every saved pattern.



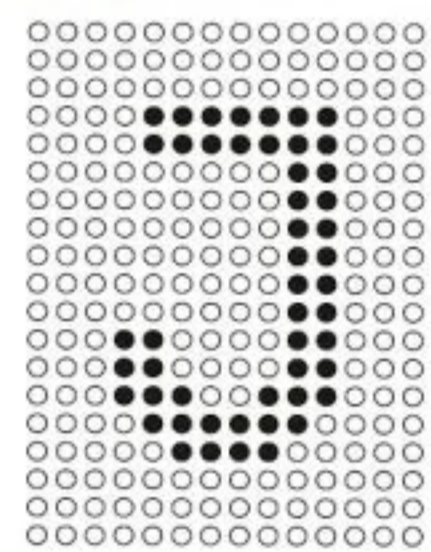
1 When the trained network is fed with a distorted or incomplete pattern, it can be likened to dropping a ball down a slope in this landscape.

2 The ball rolls until it reaches a place where it is surrounded by uphills. In the same way, the network makes its way towards lower energy and finds the closest saved pattern.

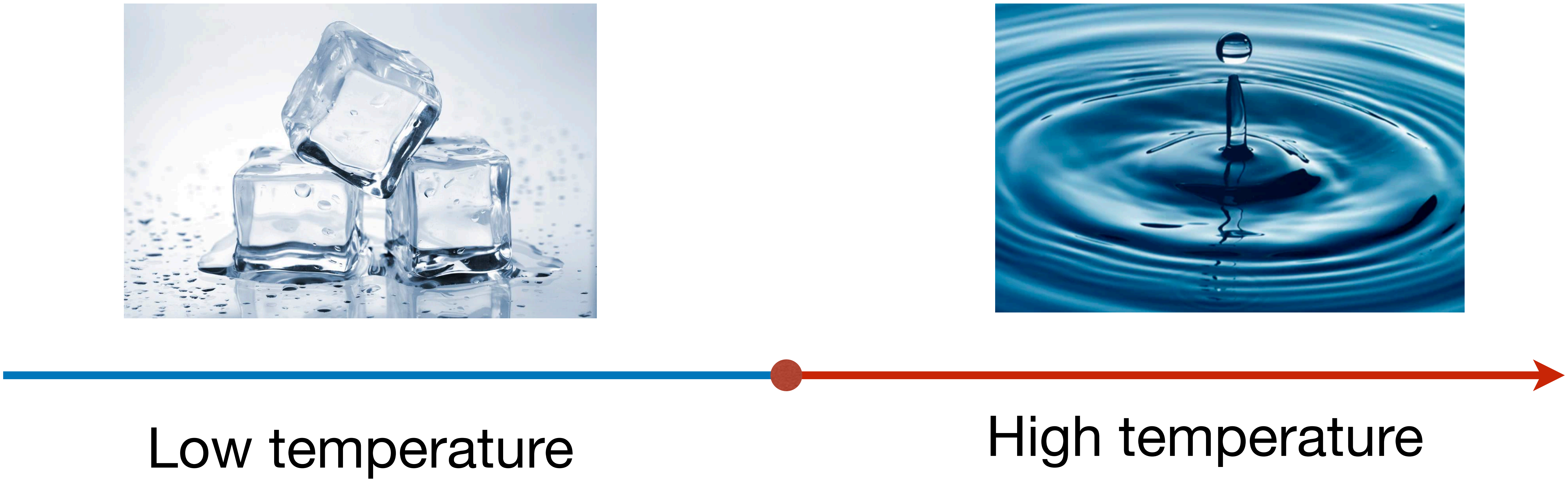
INPUT PATTERN



SAVED PATTERN



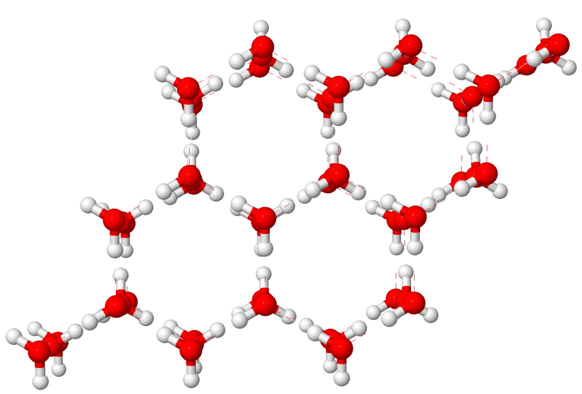
大自然的分布与采样

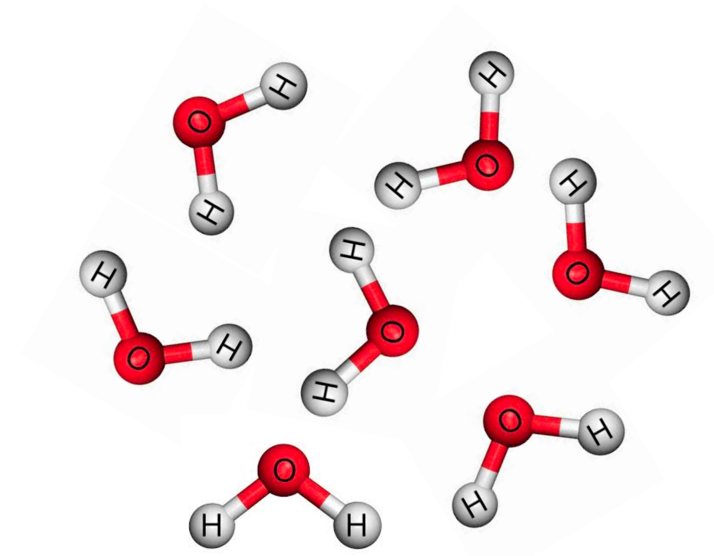


大自然的分布与采样

Low temperature

High temperature



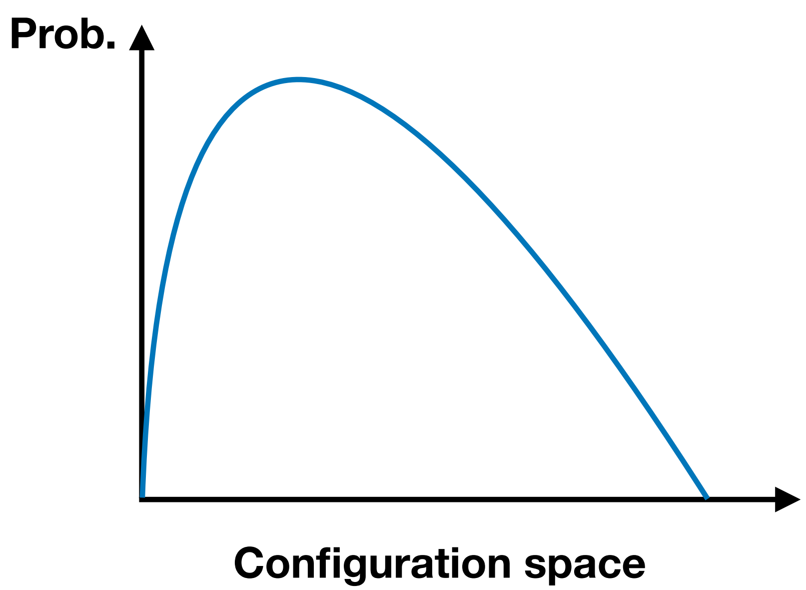


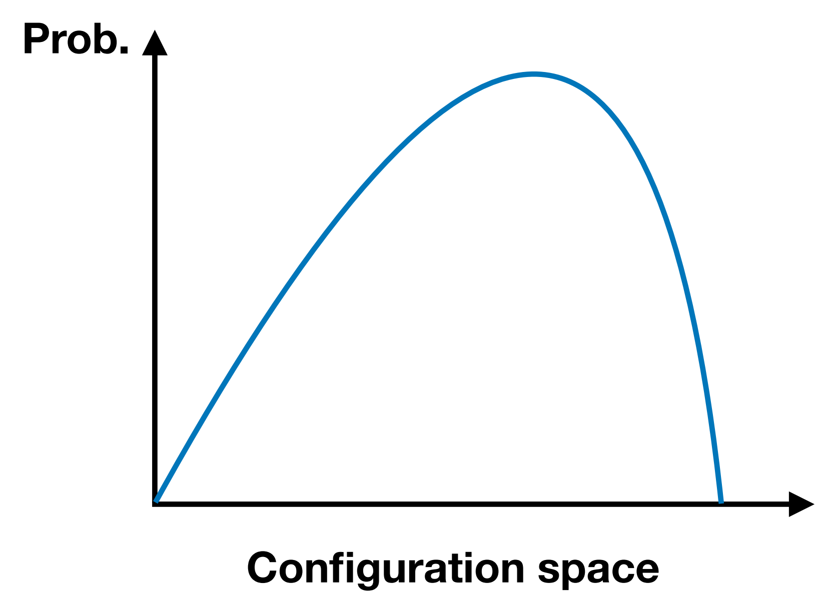
molecules are the same

大自然的分布与采样

Low temperature

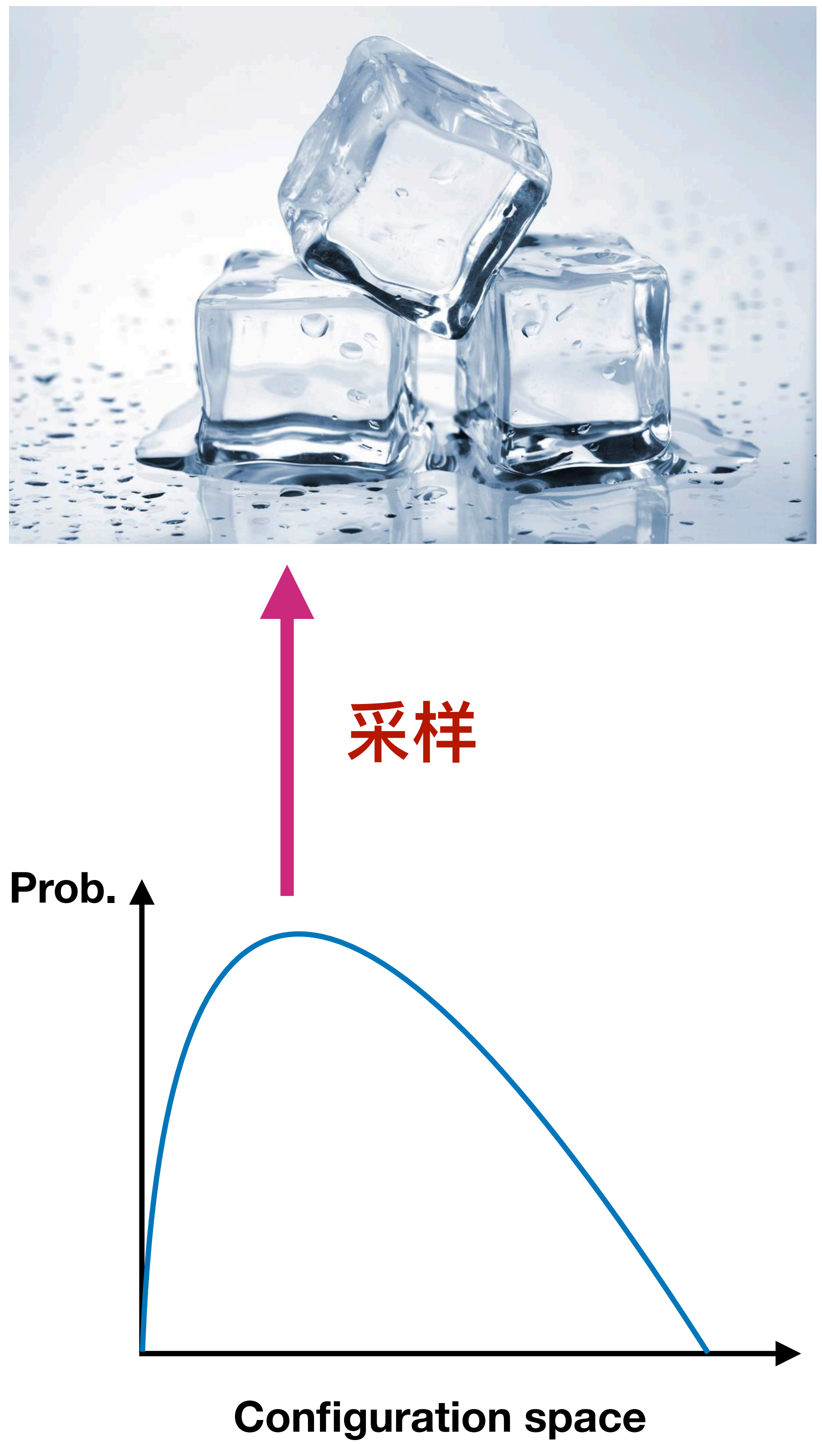
High temperature

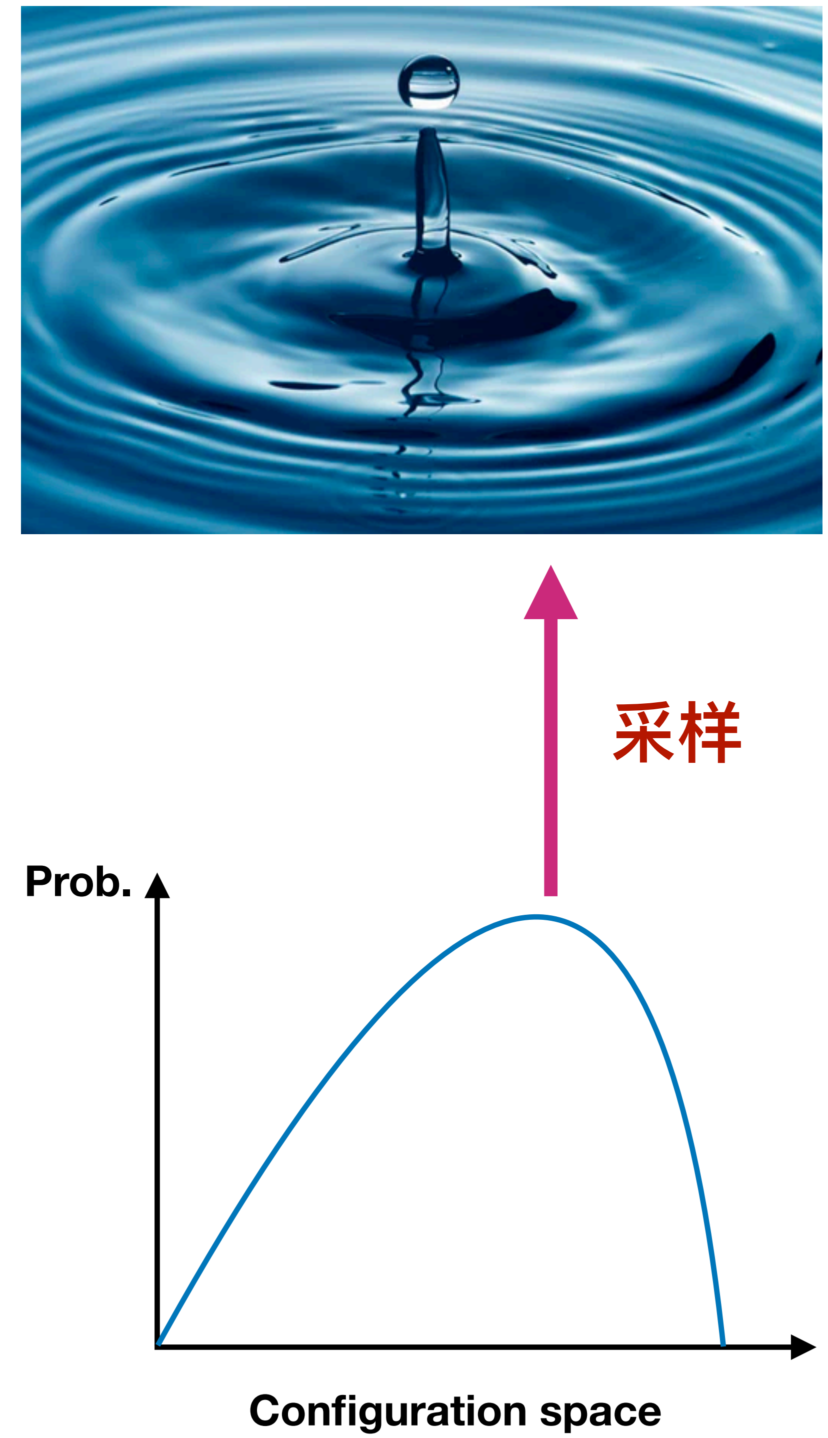




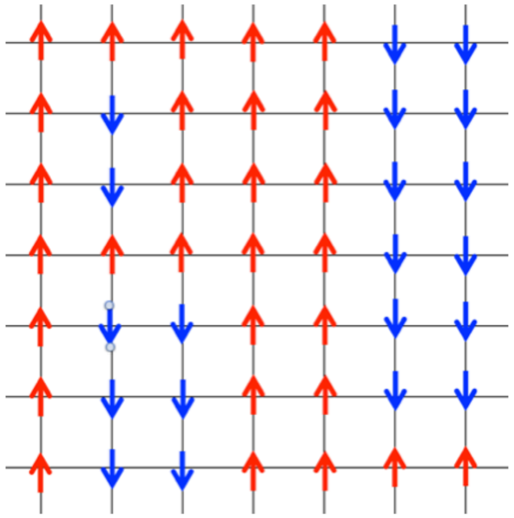
molecules are the same, but *distributions* are different

大自然的分布与采样





统计物理与机器学习



Joint distribution of
micro-configurations

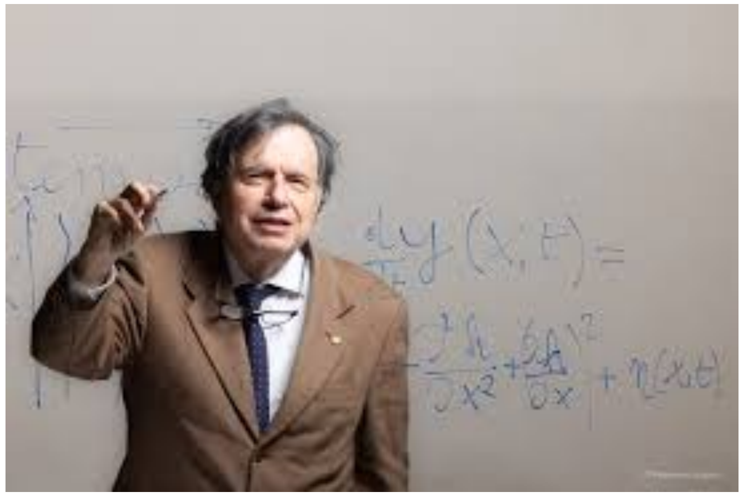
$$P(\sigma) = \frac{1}{Z}\exp(-\beta E(\sigma))$$

Joint distribution of
data variables

$$P(\mathrm{Data})$$

Exponential-large space
Efficient methods
Computational power

AN INTRODUCTION TO LEARNING AND GENERALISATION



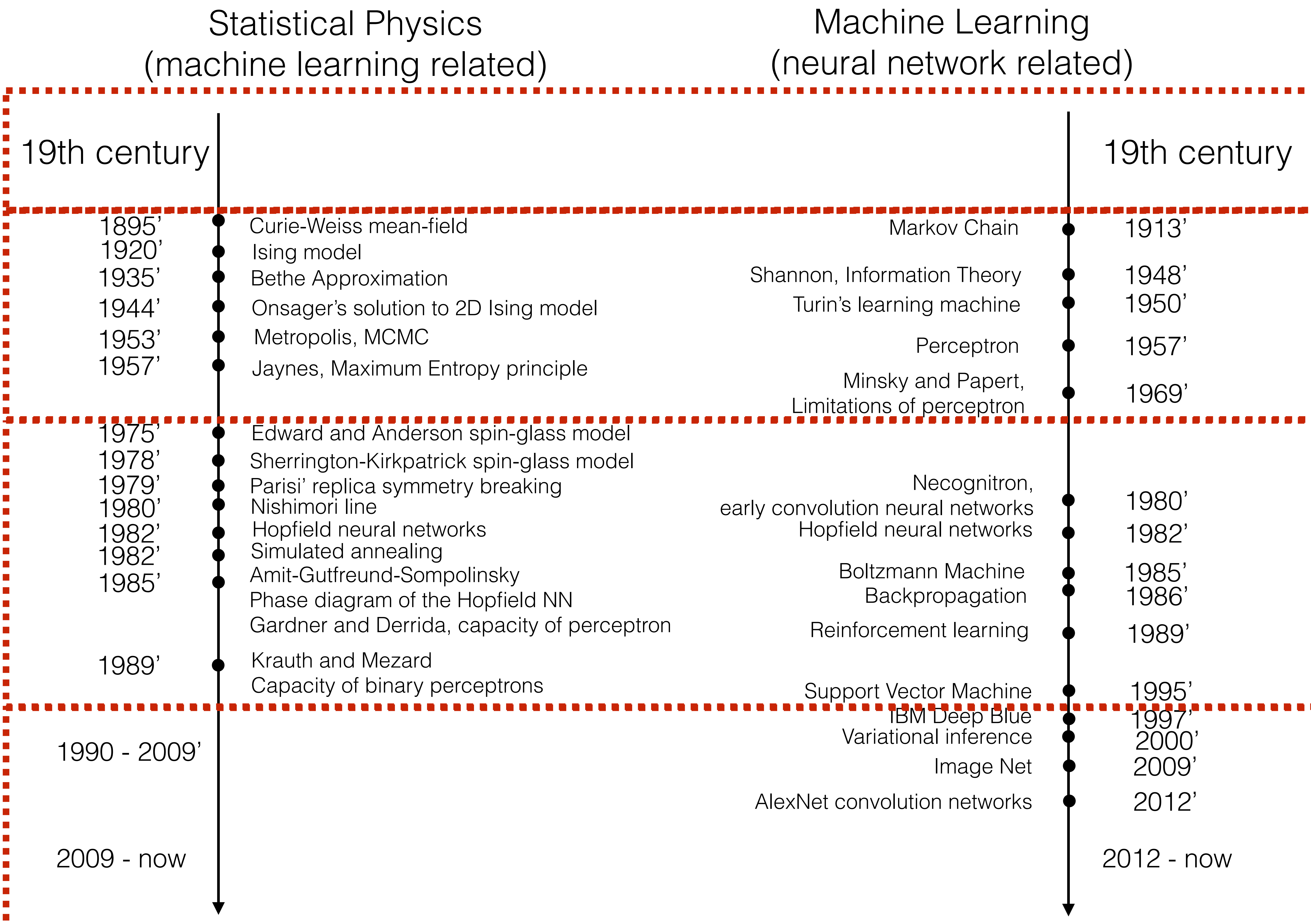
Giorgio Parisi
Dipartimento di Fisica
Piazzale delle Scienze
Roma Italy 00185

ABSTRACT. In this lecture I will present some basic ideas on how computers may learn rules from examples and how generalisation may be achieved. The general prospective is presented. Some comments are also done on the definition of intelligence.

Learning – Generalisation – Intelligence

Giorgio Parisi 1992'

Boltzmann Medal, Lars Onsager Medal, Dirac Medal, Nobel Prize


Statistical Physics
(machine learning related)
19th century
1895' Curie-Weiss mean-field
1920' Ising model
1935' Bethe Approximation
1944' Onsager's solution to 2D Ising model
1953' Metropolis, MCMC
1957' Jaynes, Maximum Entropy principle
1975' Edward and Anderson spin-glass model
1978' Sherrington-Kirkpatrick spin-glass model
1979' Parisi' replica symmetry breaking
1980' Nishimori line
1982' Hopfield neural networks
1982' Simulated annealing
1985' Amit-Gutfreund-Sompolinsky
Phase diagram of the Hopfield NN
Gardner and Derrida, capacity of perceptron
1989' Krauth and Mezard
Capacity of binary perceptrons
1990 - 2009'
2009 - now
Machine Learning
(neural network related)
19th century
Markov Chain 1913'
Shannon, Information Theory 1948'
Turin's learning machine 1950'
Perceptron 1957'
Minsky and Papert,
Limitations of perceptron 1969'
Necognitron,
early convolution neural networks 1980'
Hopfield neural networks 1982'
Boltzmann Machine 1985'
Backpropagation 1986'
Reinforcement learning 1989'
Support Vector Machine 1995'
IBM Deep Blue 1997'
Variational inference 2000'
Image Net 2009'
AlexNet convolution networks 2012'
2012 - now

4.4. NEURAL NETWORKS

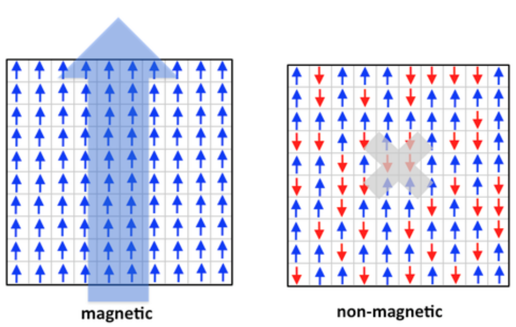
One wide ranging development, in the statistical physics of neural networks, has been the so-called Gardner approach, namely a statistical analysis in parameter space, i.e. the space of interactions (e.g. synaptic weights). It has been called the inverse problem of statistical mechanics, because in ordinary statistical mechanics the interactions are given and the statistical analysis is done in variable space (e.g. the space of neural activities). At this point,

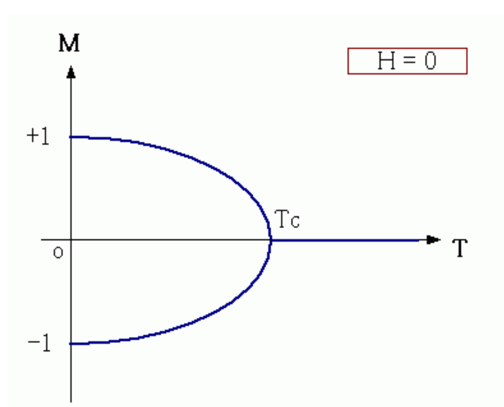
Gerard Toulouse 1992'

Langevin Prize, Holweck Prize

The Ising model
(*Ising, 1924*)

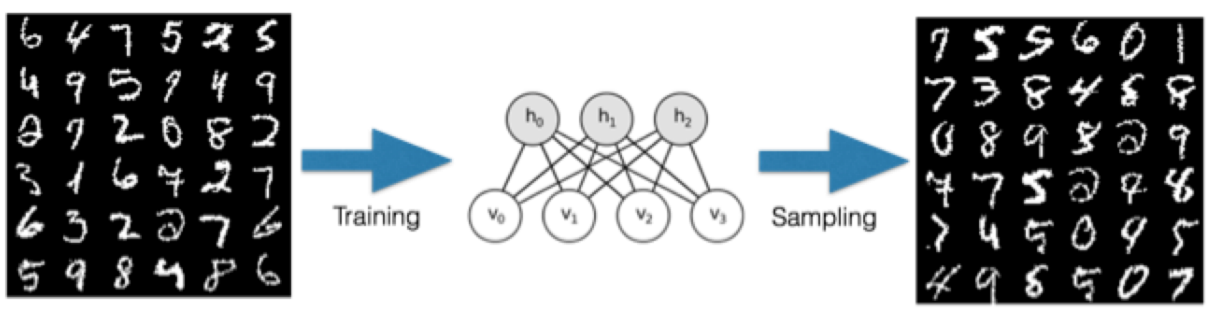
Ordinary Statistical Mechanics



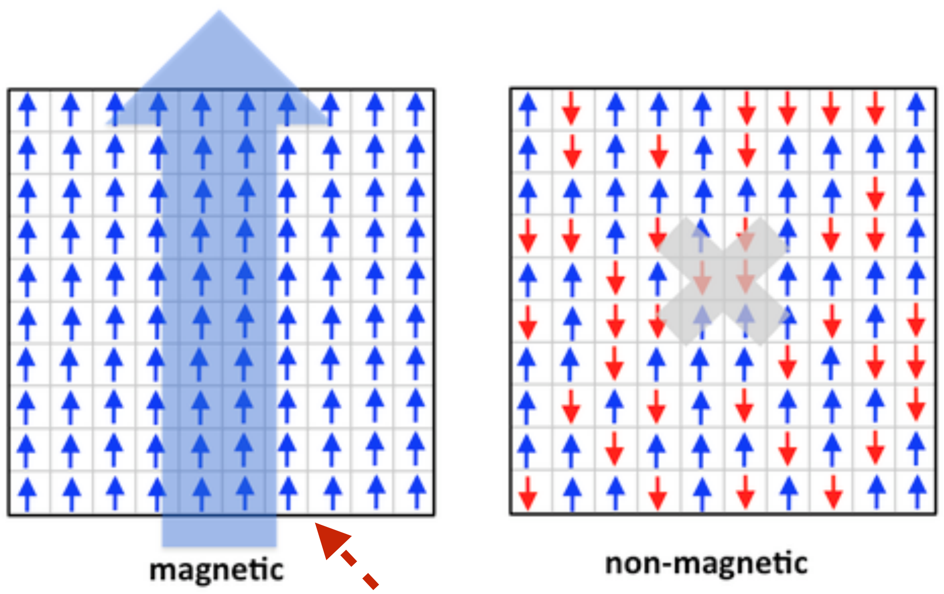


Restricted Boltzmann Machine
(*Ackley, **Hinton**, Sejnowski, 1985*)

Inverse problem of statistical mechanics

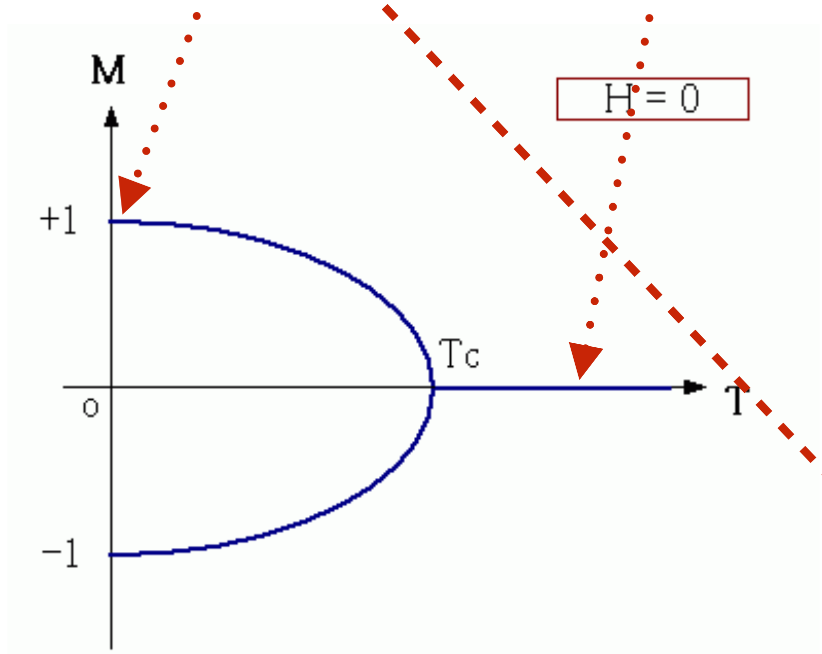


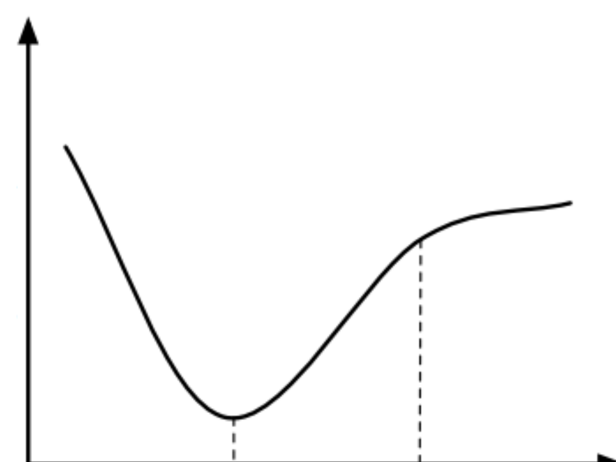
伊辛模型 (Ising, 1924)



$S = \{+1, -1\}^n$ ↑↑↑↓↑↓↓↓↓↓↑↑

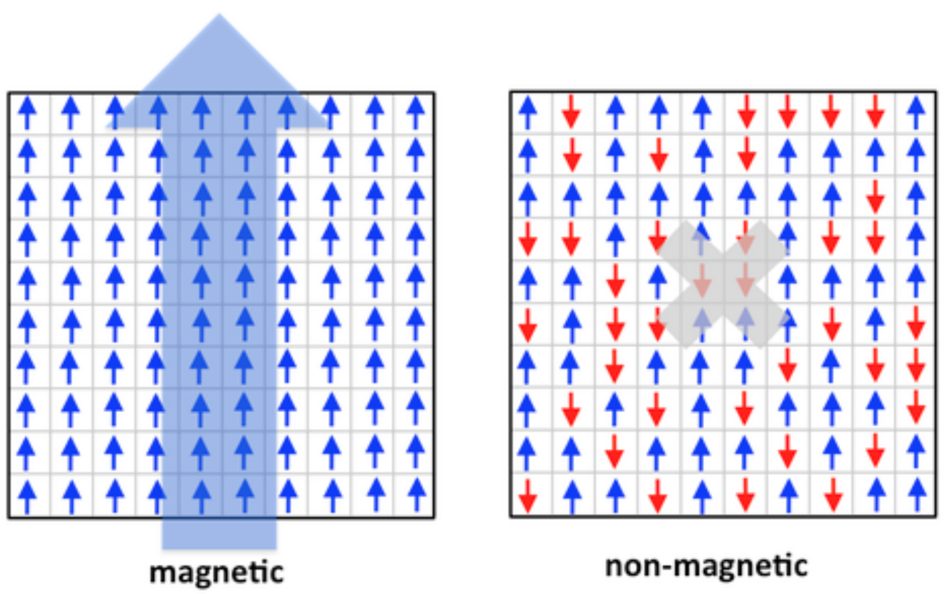
$$P(S) = \frac{1}{Z} e^{\beta \sum_{(ij)} J_{ij} S_i S_j} \qquad J_{ij} = 1$$





- 高温：记不住任何数据
- 低温：记住一个数据（全黑或者全白）！

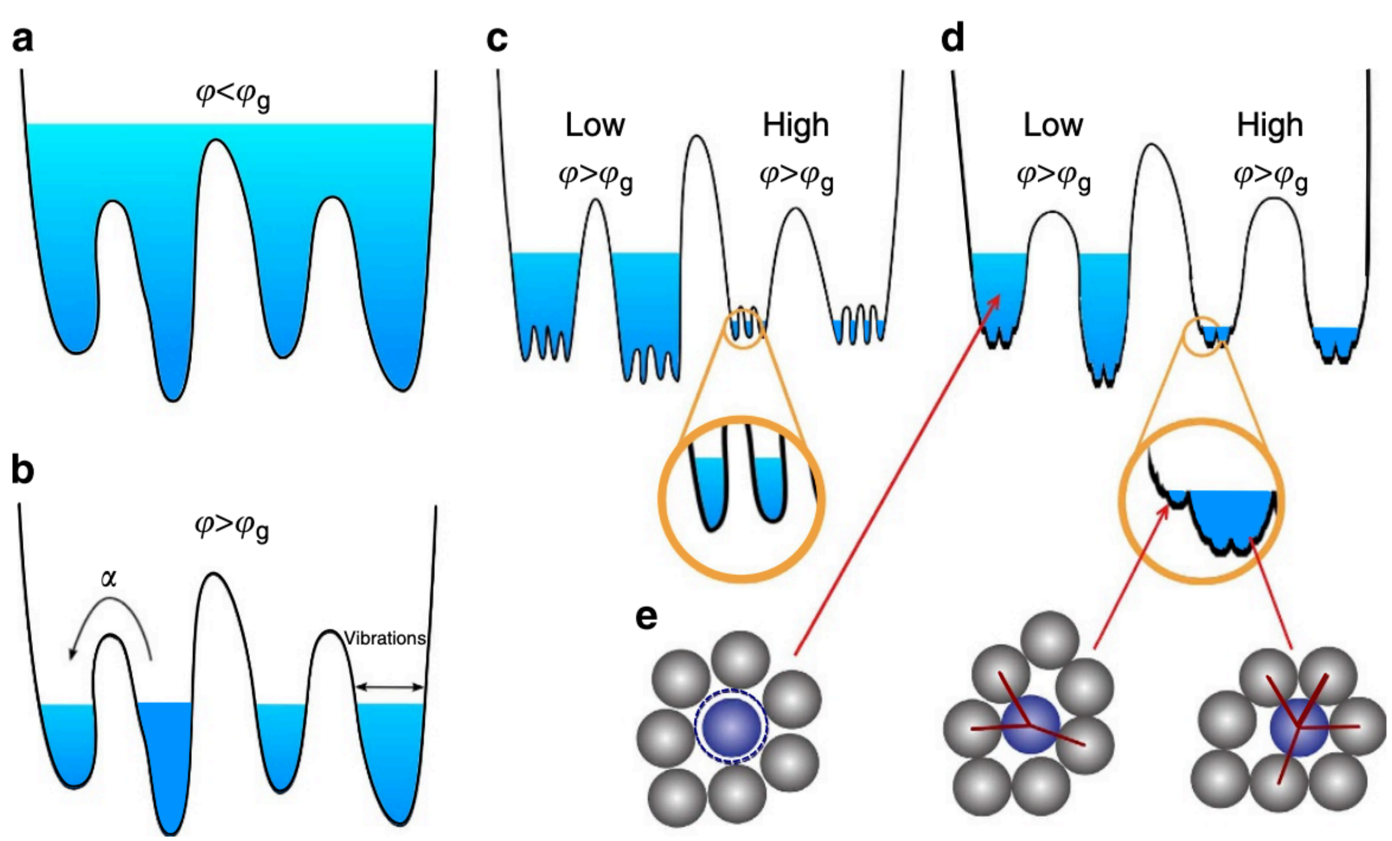
自旋玻璃模型（忘掉所有的数据）



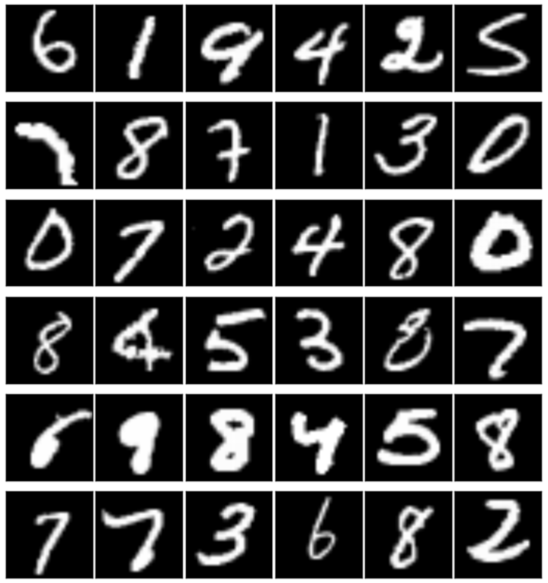
$$S = \{+1, -1\}^n \quad \uparrow\uparrow\uparrow\downarrow\uparrow\downarrow\downarrow\downarrow\downarrow\downarrow\uparrow\uparrow$$

$$P(S) = \frac{1}{Z} e^{\beta \sum_{(ij)} J_{ij} S_i S_j} \qquad J_{ij} \sim \mathcal{N}(0, 1/n)$$

Sherrington-Kirkpatrick 1975’
Parisi (full RSB solution) 1979’



Hopfield model: 记住多个数据 (Hopfield, 1982)



Associative memory (Hopfield, 1982, Amari 1977, Little 1974)

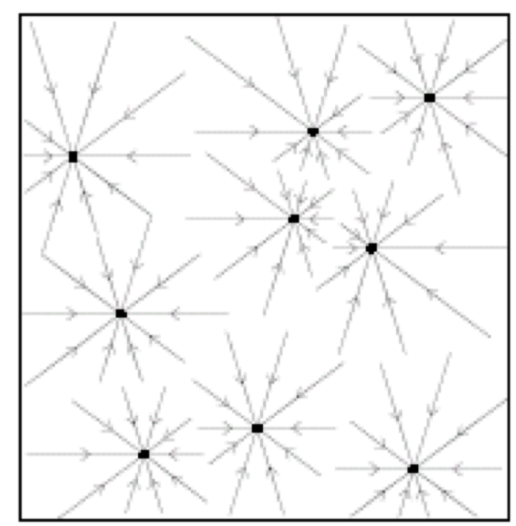
$$P(S) = \frac{1}{Z} e^{\beta \sum_{(ij)} J_{ij} S_i S_j} \qquad \left\{ \xi_i^\mu \right\} \in \{+1, -1\}^{\alpha n \times n}$$

$$J_{ij} = \frac{1}{\alpha n} \sum_{\mu=1}^{\alpha n} \xi_i^\mu \xi_j^\mu$$

Hebb's learning rule
Hebb 1949

网络更新规则: Glauber dynamics (*Glauber* 1963)

$$P(S_i) = \frac{e^{\sum_{j \neq i} J_{ij} S_i S_j}}{2 \cosh(\sum_{j \neq i} J_{ij} S_i S_j)} \propto e^{\beta \frac{1}{\alpha n} \sum_{j \neq i} \sum_\mu \xi_i^\mu \xi_j^\mu S_i S_j}$$



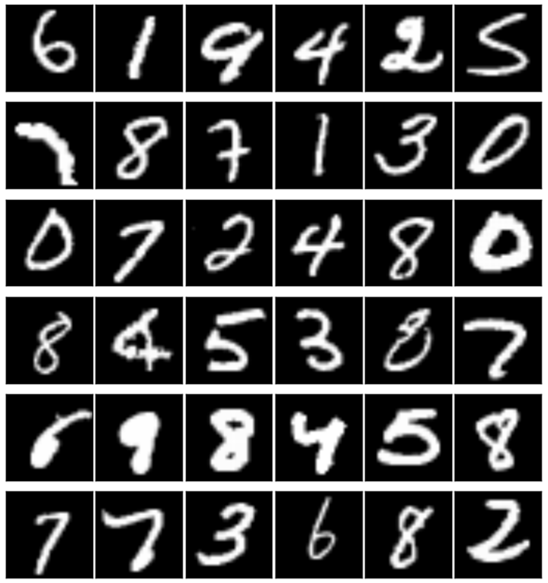
如果网络状态是第一个数据，$S_i = \xi_i^1$

$$P(S_i) \propto e^{\frac{1}{\alpha n} \sum_{j \neq i} \xi_i^1 \xi_j^1 S_i S_j + \sum_{\mu \neq 1} \sum_{j \neq i} \xi_i^\mu \xi_j^\mu S_i S_j}$$

$$= e^{\frac{1}{\alpha n} \sum_{j \neq i} 1 + \sum_{\mu \neq 1} \sum_{j \neq i} \xi_i^\mu \xi_j^\mu S_i S_j}$$

数据被存储为不动点

Hopfield model: 记住多个数据 (Hopfield, 1982)



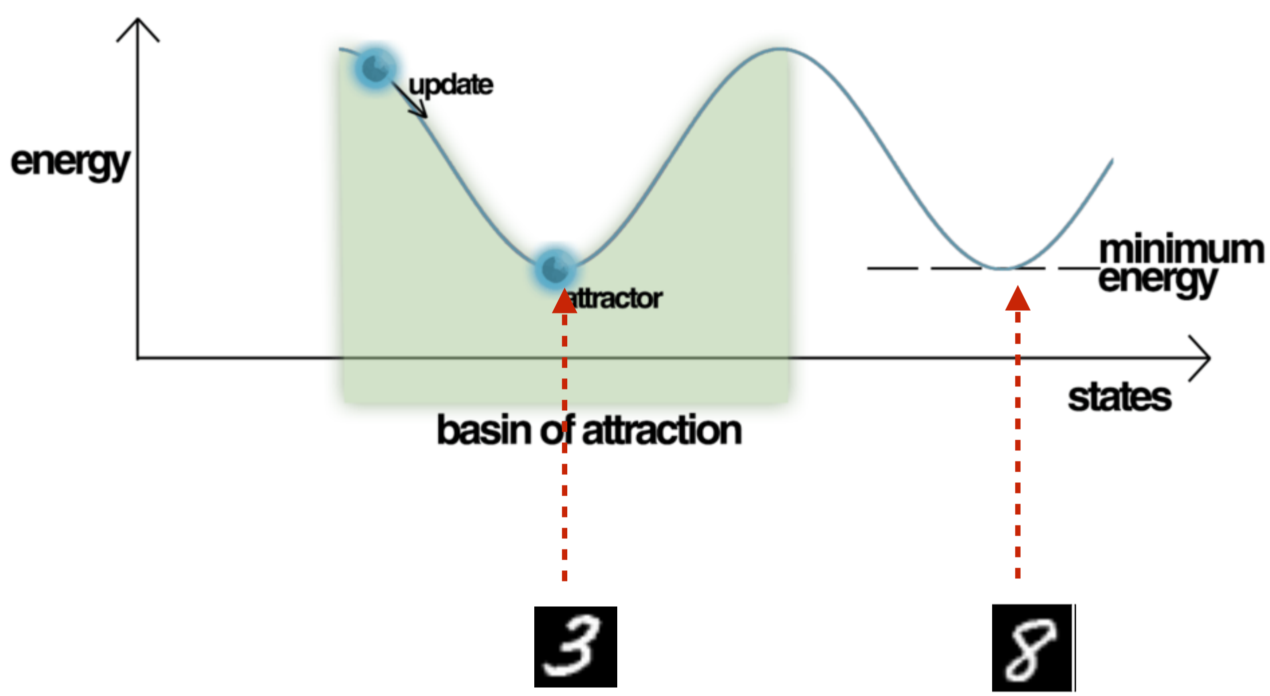
Associative memory (Hopfield, 1982, Amari 1977, Little 1974)

$$P(S) = \frac{1}{Z} e^{\beta \sum_{(ij)} J_{ij} S_i S_j} \qquad \left\{ \xi_i^\mu \right\} \in \{+1, -1\}^{\alpha n \times n}$$

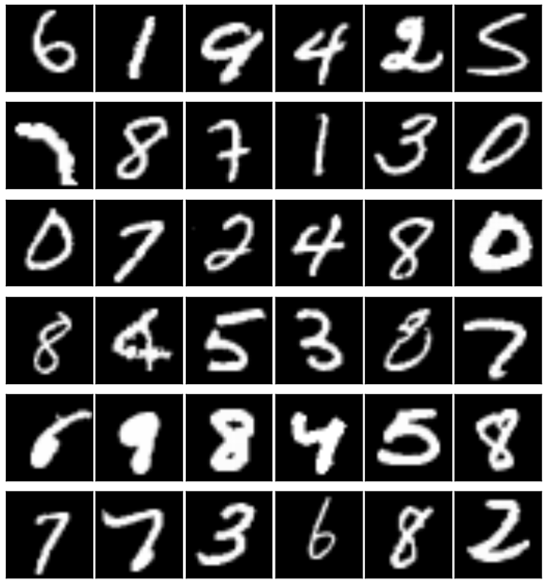
$$J_{ij} = \frac{1}{\alpha n} \sum_{\mu=1}^{\alpha n} \xi_i^\mu \xi_j^\mu$$

Hebb's learning rule

Hebb 1949



Hopfield model的统计物理理论



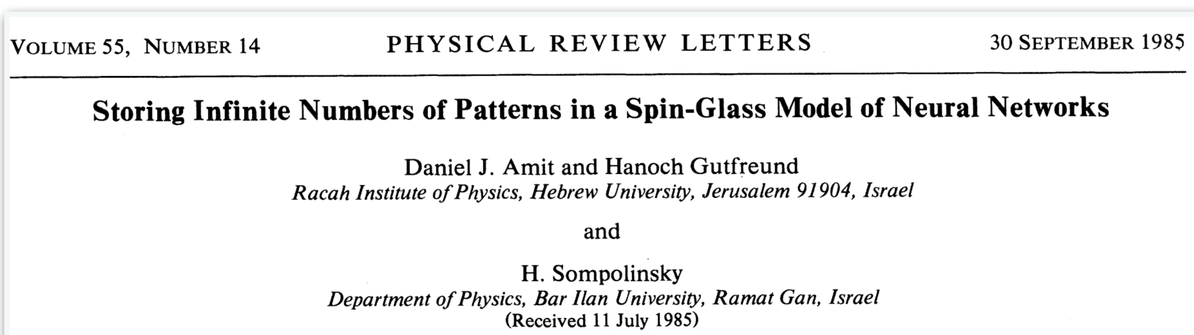
Associative memory (Hopfield, 1982, Amari 1977, Little 1974)

$$P(S) = \frac{1}{Z} e^{\beta \sum_{(ij)} J_{ij} S_i S_j}$$

$$\left\{ \xi_i^\mu \right\} \in \{+1, -1\}^{\alpha n \times n}$$

$$J_{ij} = \frac{1}{\alpha n} \sum_{\mu=1}^{\alpha n} \xi_i^\mu \xi_j^\mu$$

Hebb's learning rule
Hebb 1949



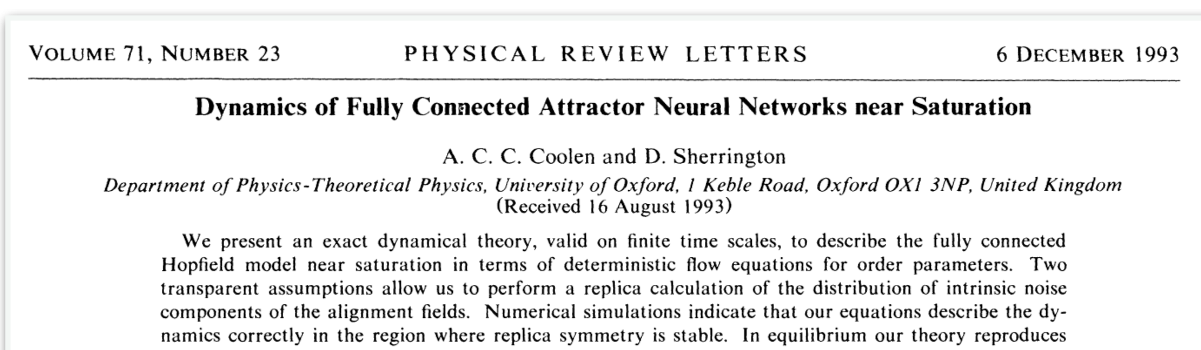
Volume 55, Number 14 PHYSICAL REVIEW LETTERS 30 September 1985

Storing Infinite Numbers of Patterns in a Spin-Glass Model of Neural Networks

Daniel J. Amit and Hanoch Gutfreund
Racah Institute of Physics, Hebrew University, Jerusalem 91904, Israel

and

H. Sompolinsky
Department of Physics, Bar Ilan University, Ramat Gan, Israel
(Received 11 July 1985)

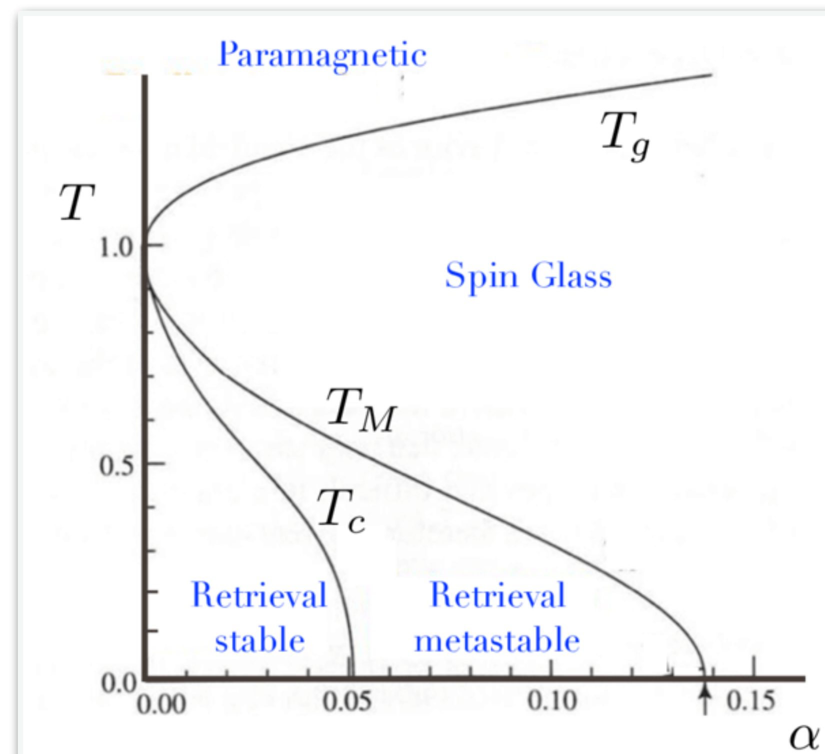


Volume 71, Number 23 PHYSICAL REVIEW LETTERS 6 December 1993

Dynamics of Fully Connected Attractor Neural Networks near Saturation

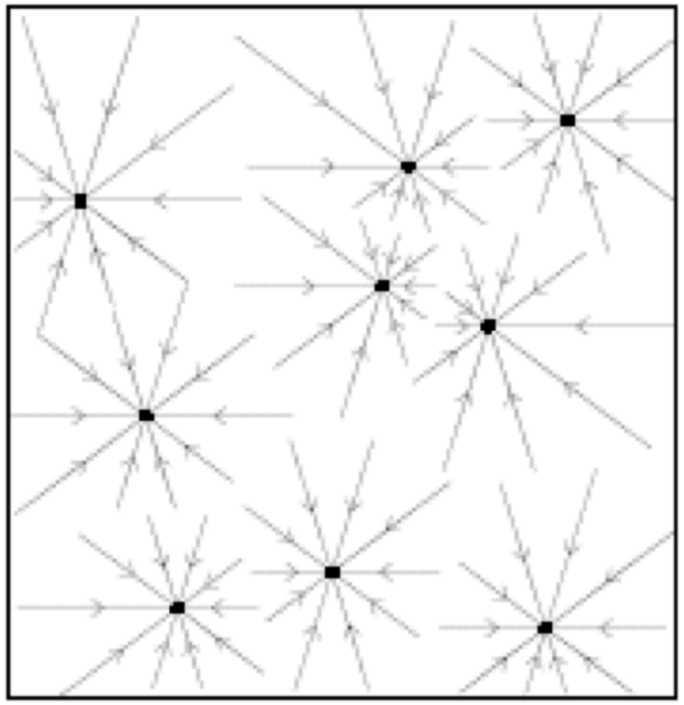
A. C. C. Coolen and D. Sherrington
Department of Physics-Theoretical Physics, University of Oxford, 1 Keble Road, Oxford OX1 3NP, United Kingdom
(Received 16 August 1993)

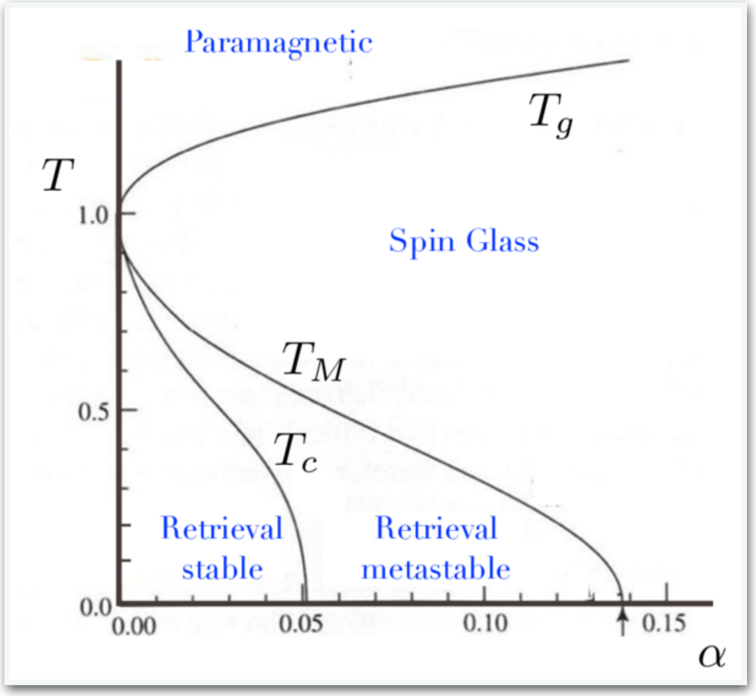
We present an exact dynamical theory, valid on finite time scales, to describe the fully connected Hopfield model near saturation in terms of deterministic flow equations for order parameters. Two transparent assumptions allow us to perform a replica calculation of the distribution of intrinsic noise components of the alignment fields. Numerical simulations indicate that our equations describe the dynamics correctly in the region where replica symmetry is stable. In equilibrium our theory reproduces



局限性：数据需要是正交的，线性capacity

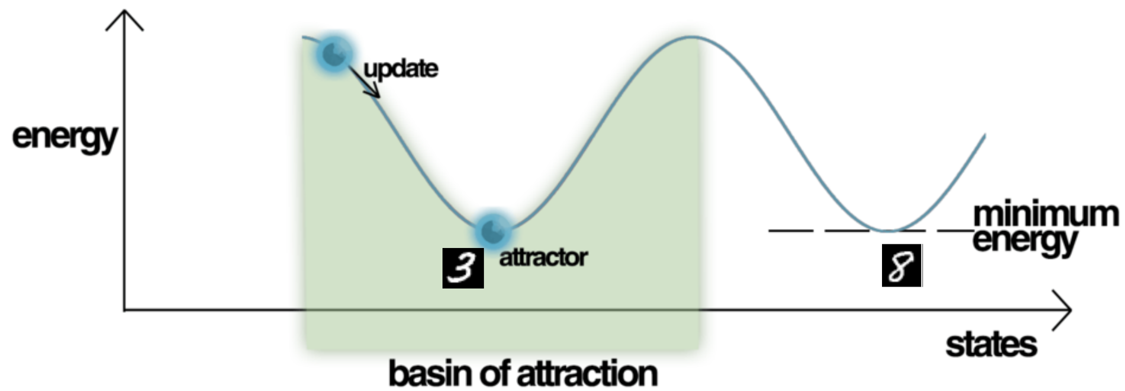
Phase diagram
Amit, Gutfreund, Sompolinsky 1985

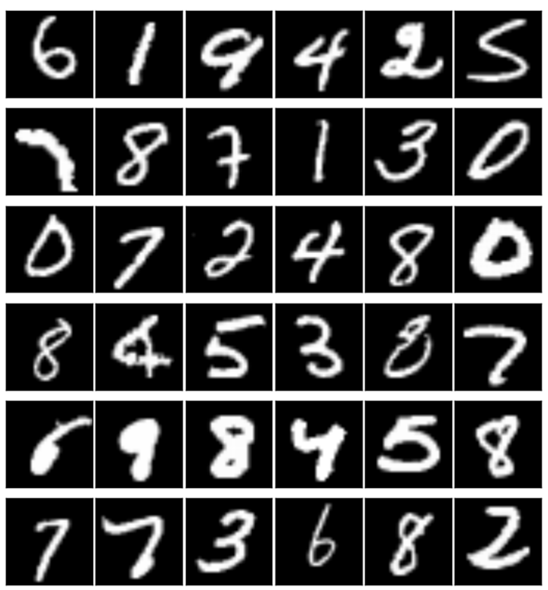




Paramagnetic
T_g
T
1.0
Spin Glass
T_M
0.5
T_c
Retrieval
stable
Retrieval
metastable
0.0
0.00
0.05
0.10
0.15
α

Inverse-Ising model: 根据数据学习权重

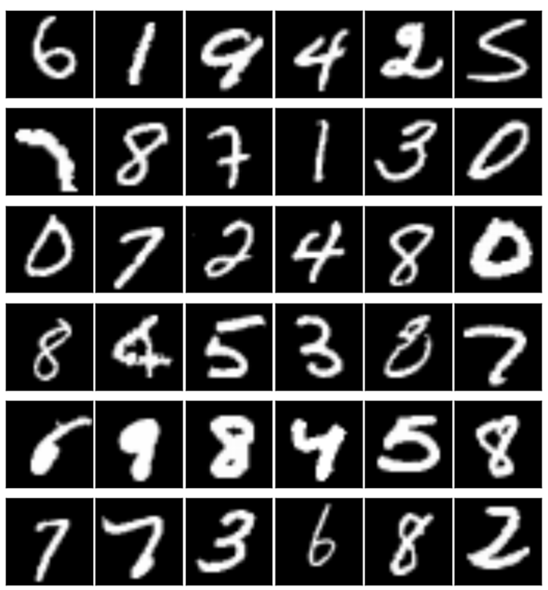




改变 J_{ij}

Ising Model

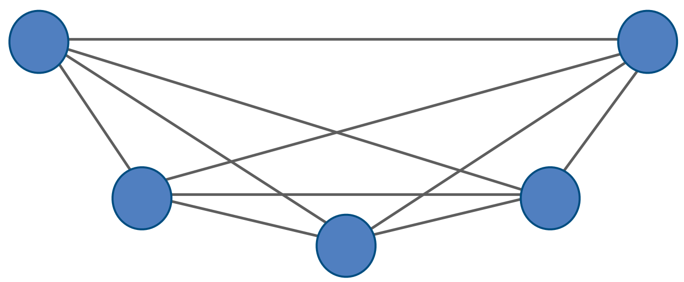
采样



$$P(S) = \frac{1}{Z} e^{\beta \sum_{(ij)} J_{ij} S_i S_j}$$

Inverse-Ising model: 根据数据学习权重

$$P(S) = \frac{1}{Z} e^{\beta \sum_{(ij)} J_{ij} S_i S_j}$$



- 全连接自旋玻璃模型

➡ 需要学习所有的J_{ij}

- 给定数据一阶矩和二阶矩的最大熵模型

$$P(s) = \arg\max H(P), \text{ s.t. } \langle s_i \rangle_P = m_i, \langle s_i s_j \rangle_P = \langle S_i S_j \rangle_{\text{data}}$$

- 指数函数族

➡ Sufficient Statistics是一阶矩和二阶矩

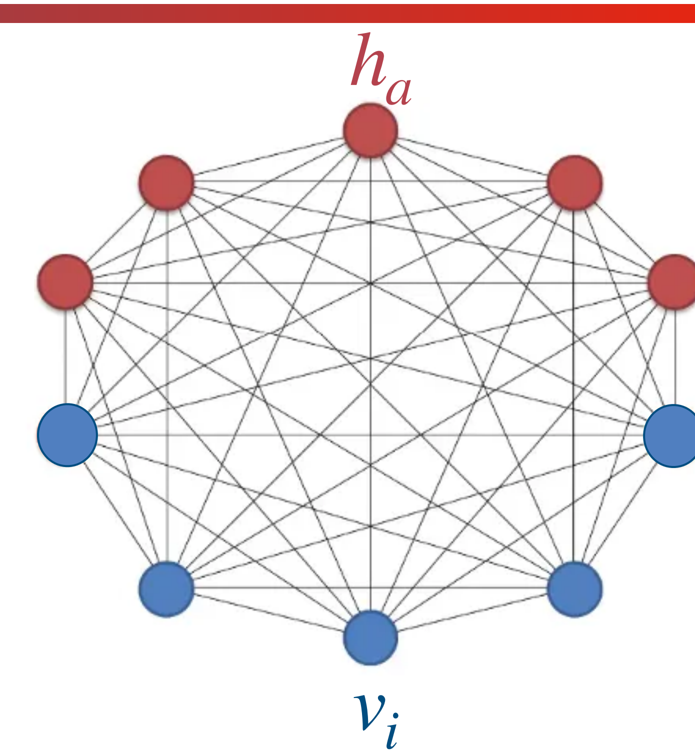
➡ $$\frac{\partial \log P}{\partial J_{ij}} = \langle S_i S_j \rangle_{\text{data}} - \langle S_i S_j \rangle_P$$

缺点：参数少，表达能力不够

玻尔兹曼机：通过隐变量增加表述能力

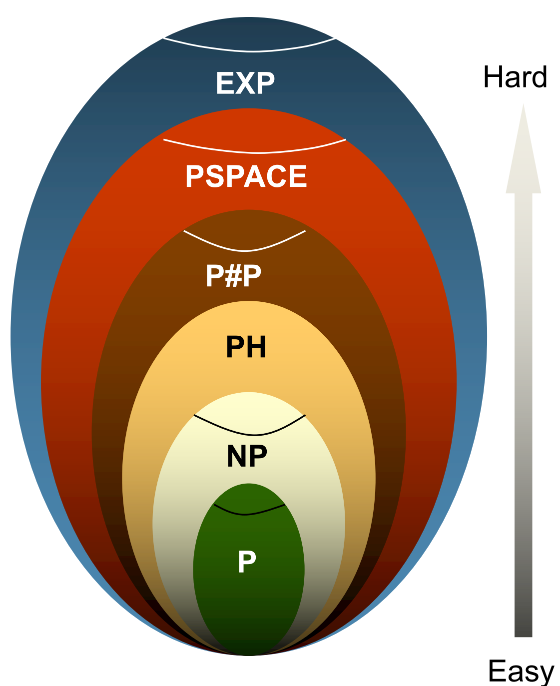
$$P(v,h) = \frac{1}{Z} e^{\sum_{(ij)} J_{ij} v_i v_j + \sum_{ab} J_{ab} h_a h_b + \sum_{ia} W_{ia} v_i h_a}$$

$$P(v) = \frac{1}{Z} \sum_h e^{\sum_{(ij)} J_{ij} v_i v_j + \sum_{ab} J_{ab} h_a h_b + \sum_{ia} W_{ia} v_i h_a}$$



- 玻尔兹曼机是带有隐变量的Inverse Ising model (*Ackley, **Hinton**, Sejnowski, 1985*)
  - ➡ 增加参数数目，模型表述能力
  - ➡ 表达数据中的高阶关联
- 缺点：难以训练

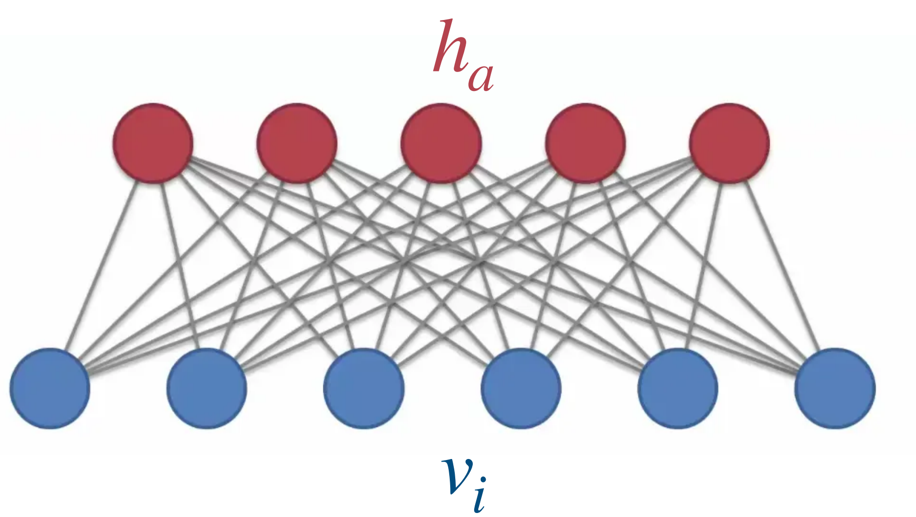
$$-\frac{\partial \log P}{\partial W_{ia}} = \langle v_i h_a \rangle_{\text{data+model}} - \langle v_i h_a \rangle_{\text{model}}$$



受限玻尔兹曼机 (RBM)：有效的训练算法

$$P(v,h) = \frac{1}{Z} e^{\sum_{ia} W_{ia} v_i h_a}$$

$$P(v) = \frac{1}{Z} \sum_h e^{\sum_{ia} W_{ia} v_i h_a}$$



- RBM是二分图上的玻尔兹曼机 (*Hinton, Sejnowski, 1986*)
  - ➡ 只有隐变量和显变量之间的连接
  - ➡ 数据中的高阶关联通过隐变量诱导
  - ➡ Contrastive Divergence算法可有效计算梯度（*Hinton* 2002)

$$\frac{\partial \log P}{\partial W_{ia}} = \langle v_i h_a \rangle_{\text{data+model}} - \langle v_i h_a \rangle_{\text{model}}$$

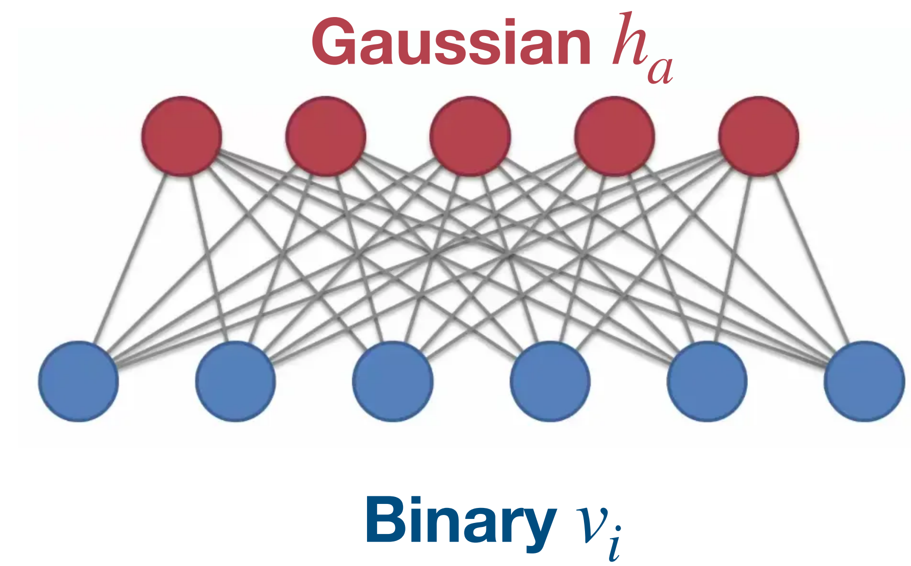
减小数据能量　　增加其他构型能量

奇怪的知识：Hopfield model等价于Gaussian RBM

ELSEVIER
Neural Networks
Volume 34, October 2012, Pages 1-9

On the equivalence of Hopfield networks and Boltzmann Machines

Adriano Barra [a], Alberto Bernacchia [b], Enrica Santucci [c], Pierluigi Contucci [d]



PHYSICAL REVIEW E **95**, 022117 (2017)

Mean-field message-passing equations in the Hopfield model and its generalizations

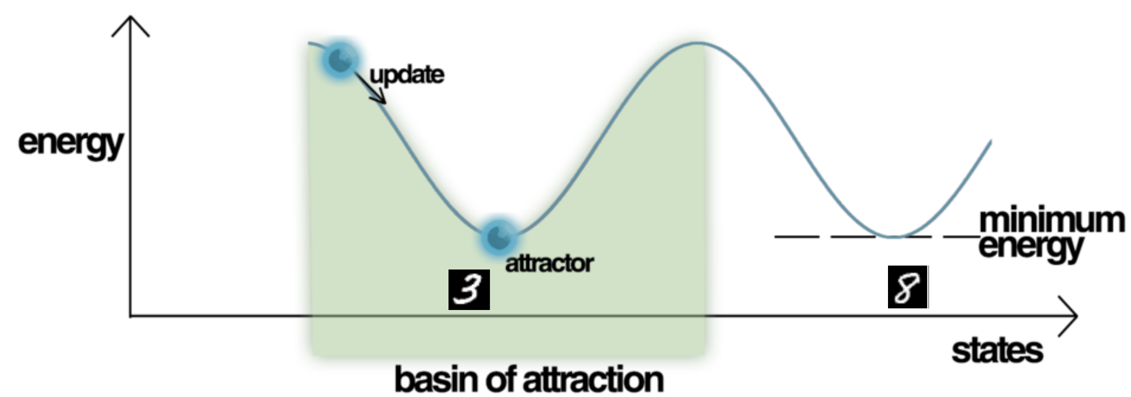
Marc Mézard
Physics Department, Ecole Normale Supérieure, PSL Research University, Paris
(Received 26 August 2016; published 14 February 2017)

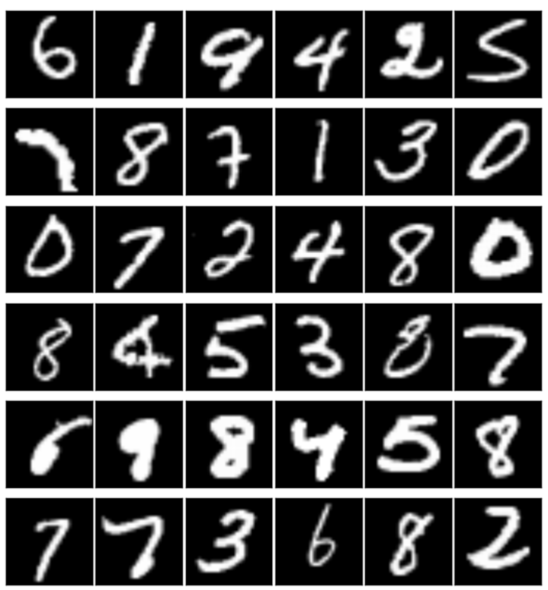
Motivated by recent progress in using restricted Boltzmann machines as preprocessing algorithms for deep neural network, we revisit the mean-field equations [belief-propagation and Thouless-Anderson Palmer (TAP) equations] in the best understood of such machines, namely the Hopfield model of neural networks, and we explicit how they can be used as iterative message-passing algorithms, providing a fast method to compute the local polarizations of neurons. In the "retrieval phase", where neurons polarize in the direction of one memorized pattern, we point out a major difference between the belief propagation and TAP equations: The set of belief propagation equations depends on the pattern which is retrieved, while one can use a unique set of TAP equations.

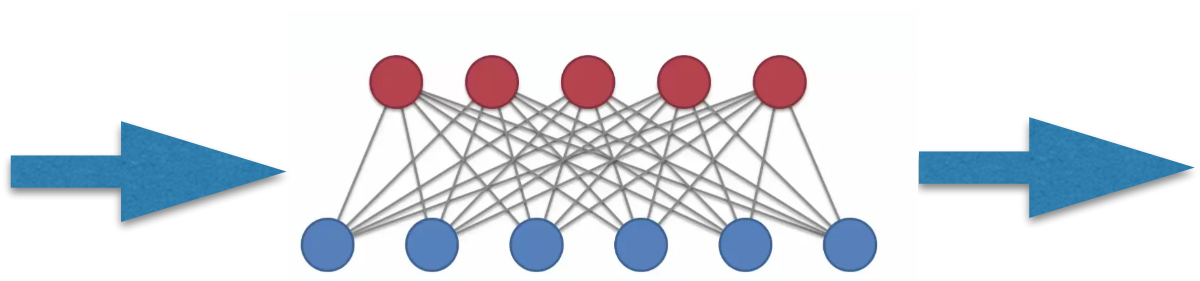
$$Z = \sum_s \int \prod_\mu \frac{d\lambda_\mu}{\sqrt{2\pi/\beta}} \times \exp\left[-\frac{\beta}{2}\sum_\mu \lambda_\mu^2 + \beta \sum_{\mu,i} \frac{\xi_i^\mu}{\sqrt{N}} s_i \lambda_\mu \right].$$

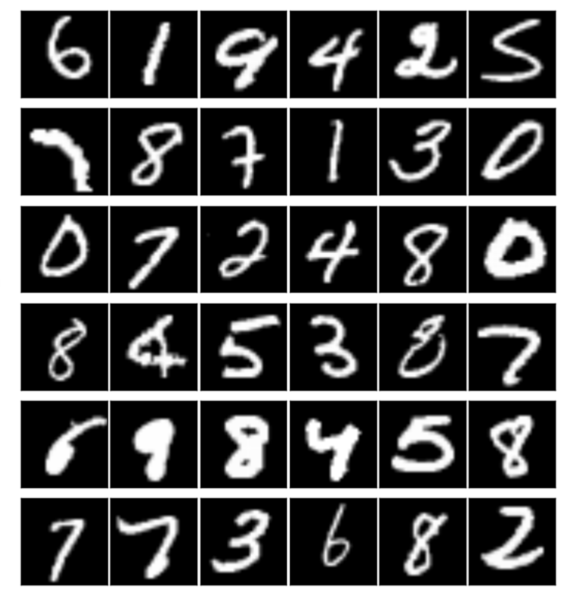
Hubbard Stratonovich 变换

用RBM学习数据分布









学习 W_{ia}

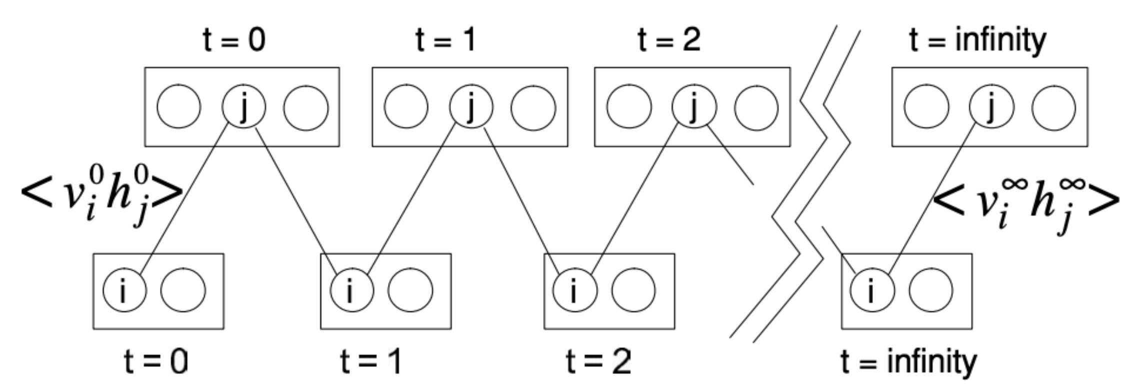
采样

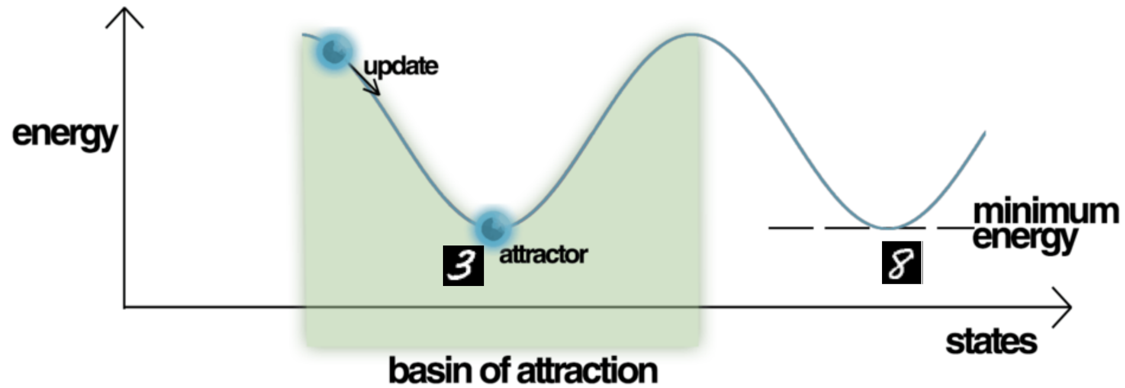
$$P(v) = \frac{1}{Z}\sum_{h} e^{\sum_{ia} W_{ia} \mathbf{v_i} \mathbf{h_a}}$$

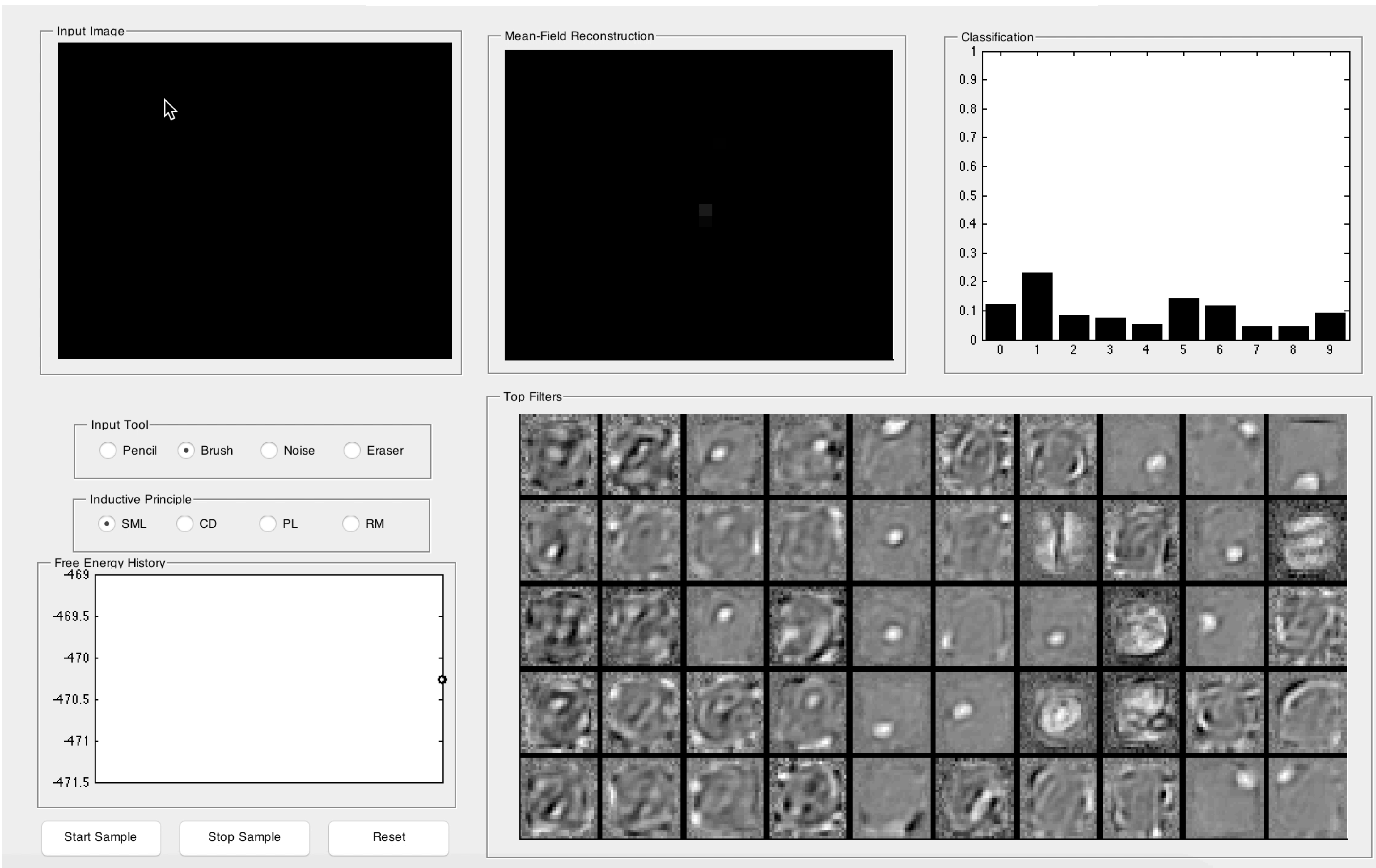
$$\frac{\partial \log P}{\partial W_{ia}} = \langle v_i h_a \rangle_{\text{data+model}} - \langle v_i h_a \rangle_{\text{model}}$$

减小数据能量

增加其他构型能量



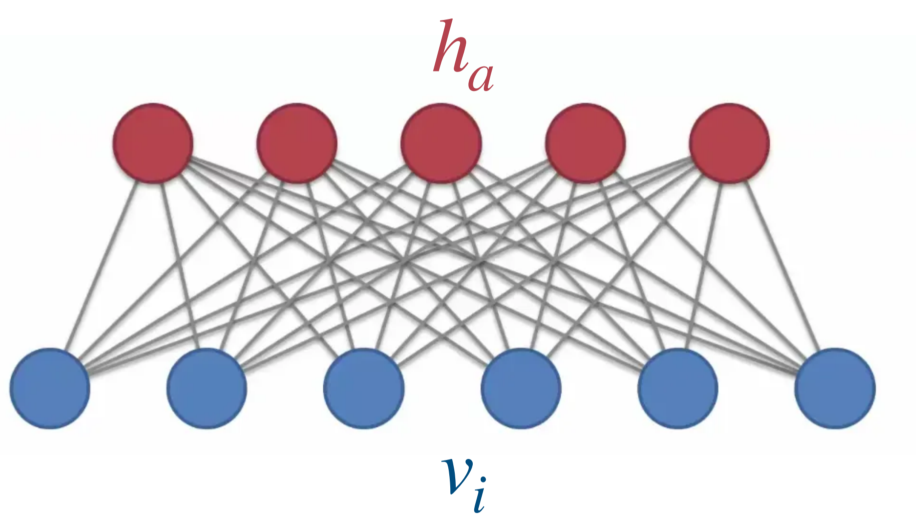

update
energy
attractor
minimum energy
states
basin of attraction


Input Image
Mean-Field Reconstruction
Classification
Top Filters
Input Tool
Pencil
Brush
Noise
Eraser
Inductive Principle
SML
CD
PL
RM
Free Energy History
Start Sample
Stop Sample
Reset

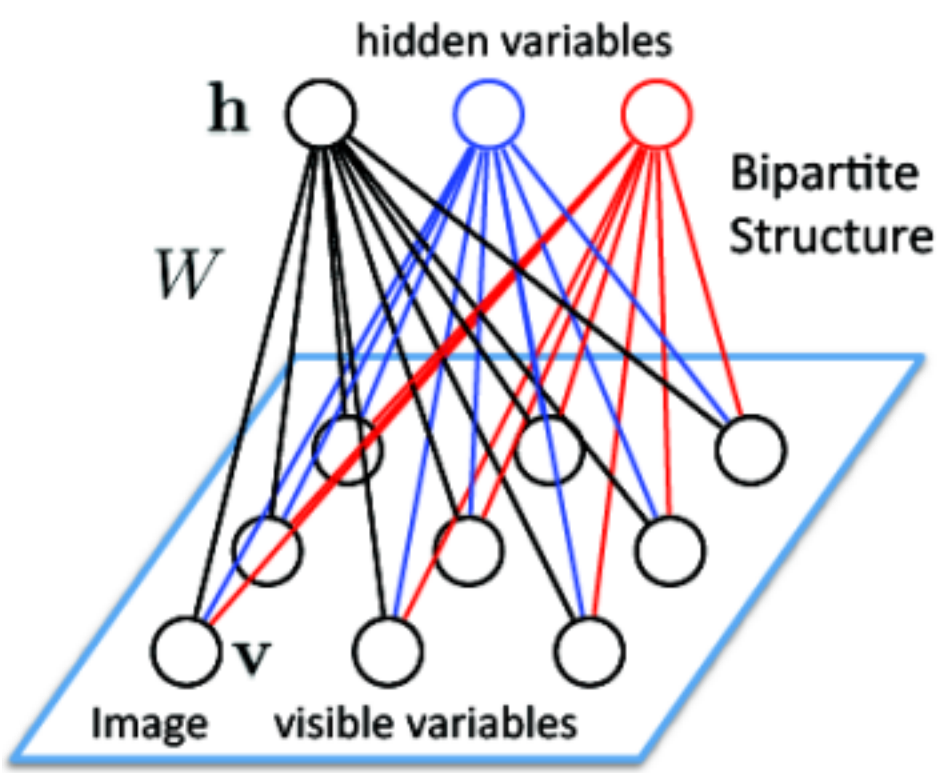
受限玻尔兹曼机 (RBM): 表示学习

$$P(v,h) = \frac{1}{Z} e^{\sum_{ia} W_{ia} v_i h_a}$$

$$P(v) = \frac{1}{Z} \sum_h e^{\sum_{ia} W_{ia} v_i h_a}$$



- RBM隐变量可以给出数据的表示(representation)
  - ➡ 显变量(数据)分布到隐变量分布
  - ➡ 分布维度缩小
  - ➡ 类似物理中的重整化



表示学习、预训练、Fine-tune

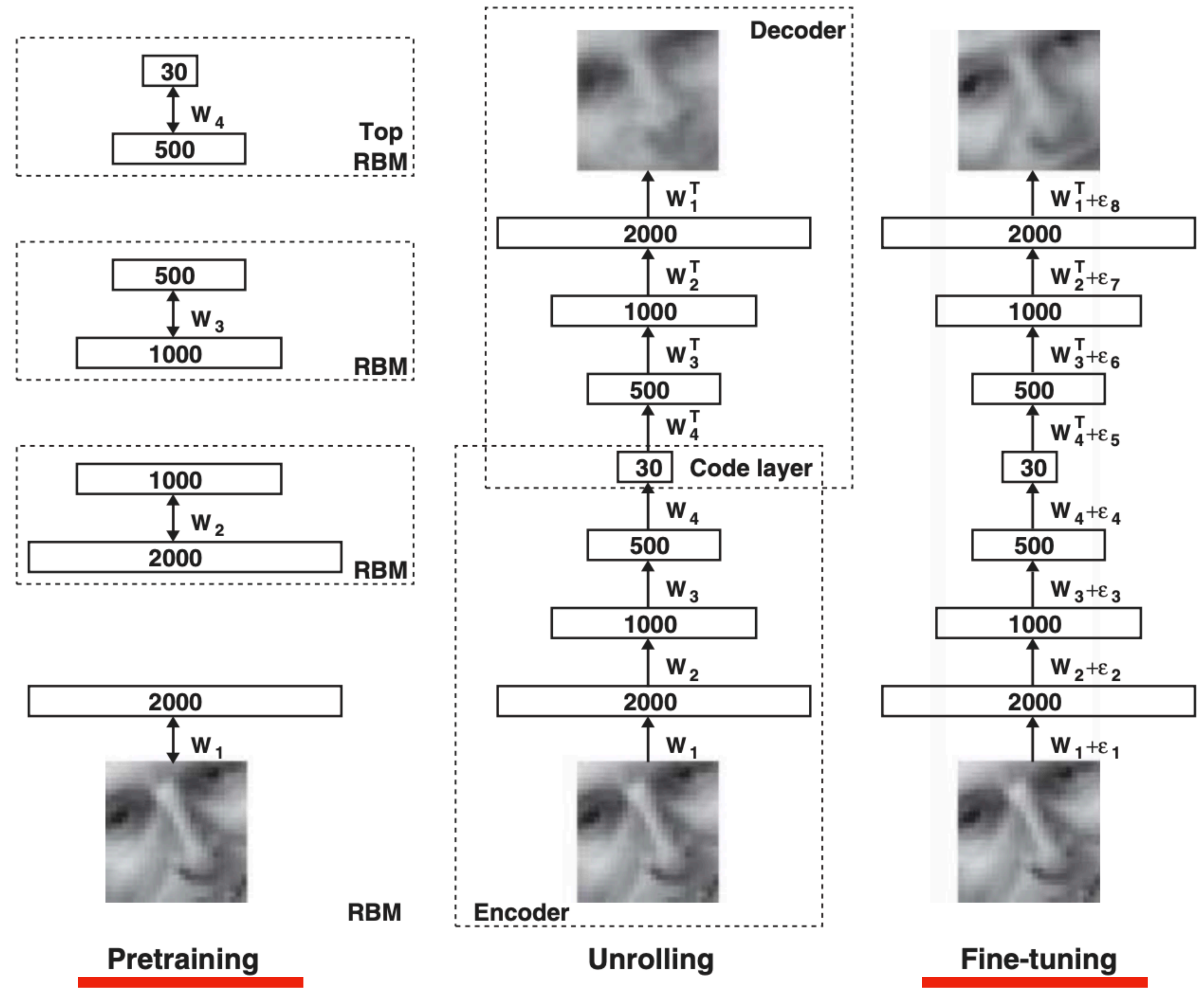


Fig. 1. Pretraining consists of learning a stack of restricted Boltzmann machines (RBMs), each having only one layer of feature detectors. The learned feature activations of one RBM are used as the "data" for training the next RBM in the stack. After the pretraining, the RBMs are "unrolled" to create a deep autoencoder, which is then fine-tuned using backpropagation of error derivatives.

Reducing the Dimensionality of Data with Neural Networks

G. E. Hinton* and R. R. Salakhutdinov

High-dimensional data can be converted to low-dimensional codes by training a multil network with a small central layer to reconstruct high-dimensional input vectors. Gradi can be used for fine-tuning the weights in such "autoencoder" networks, but this works the initial weights are close to a good solution. We describe an effective way of initial weights that allows deep autoencoder networks to learn low-dimensional codes that w better than principal components analysis as a tool to reduce the dimensionality of da

Dimensionality reduction facilitates the classification, visualization, communication, and storage of high-dimensional data. A simple and widely used method is principal components analysis (PCA), which finds the directions of greatest vari data set and represents each data p coordinates along each of these dire describe a nonlinear generalization o uses an adaptive, multilayer "encode

2006 VOL 313 SCIENCE www.sciencemag.org

Learning representations by back-propagating errors

David E. Rumelhart*, Geoffrey E. Hinton† & Ronald J. Williams*

* Institute for Cognitive Science, C-015, University of California, San Diego, La Jolla, California 92093, USA
† Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Philadelphia 15213, USA

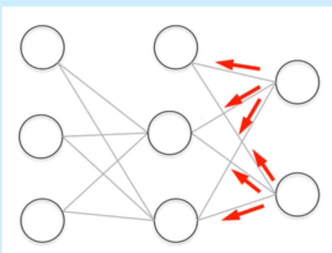
Rumelhart, Hinton, Williams, 1986

"My main contribution was to show how you can use it for learning distributed representations" - Hinton

从Hinton的主要工作看神经网络学习发展

生成学习萌芽

基于RBM的
非监督学习
发展缓慢

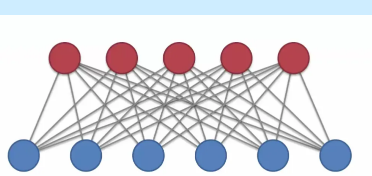


Back-propagation
1986

Deep belief network
Autoencoder
Pre-training
Fine-tune
2006

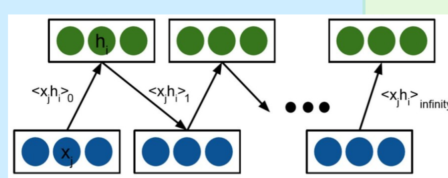
2012

1985
Boltzmann
Machine

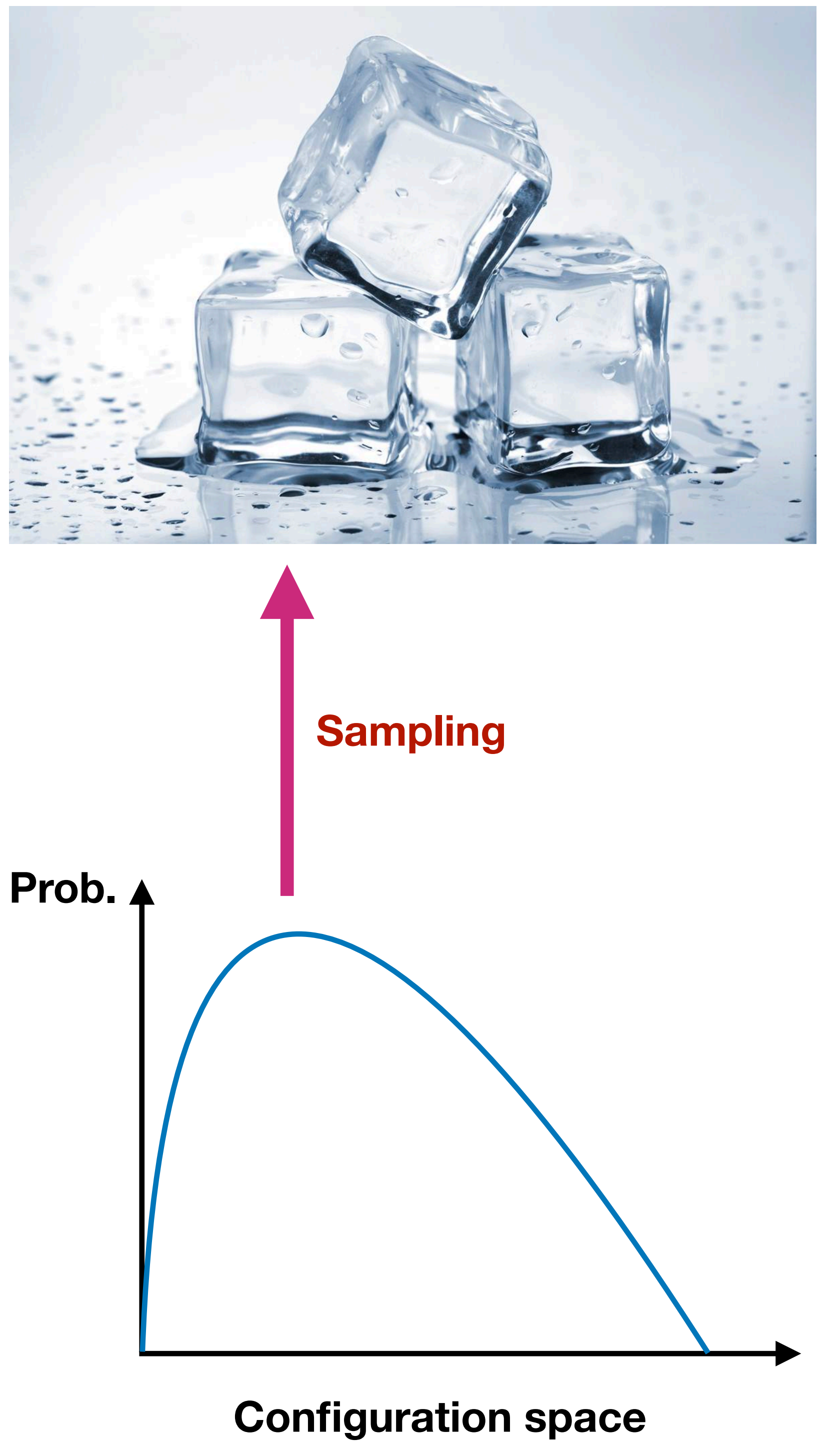


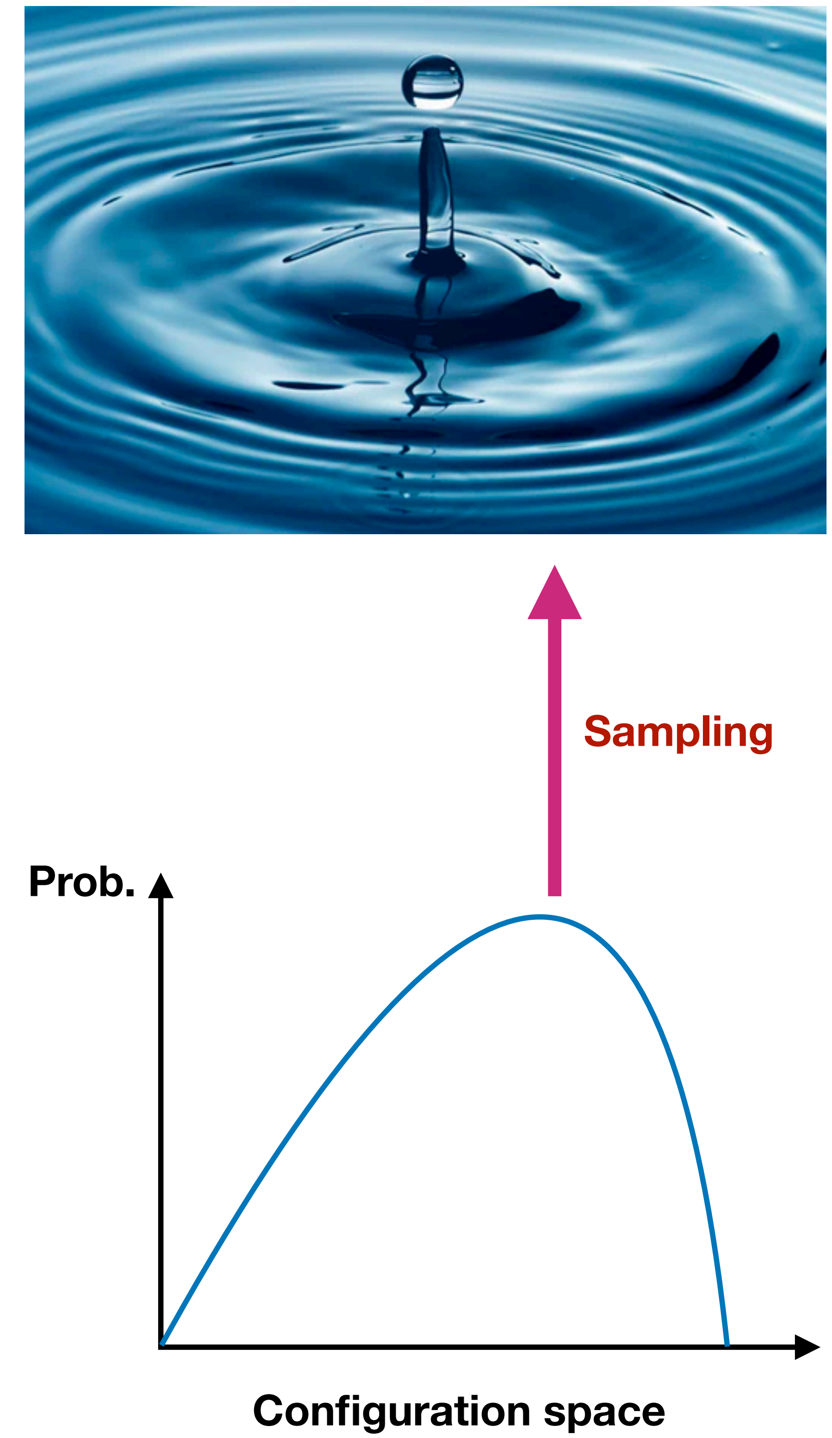
1995
Helmholtz
Machine

2002
Con. Div.
Product of Experts

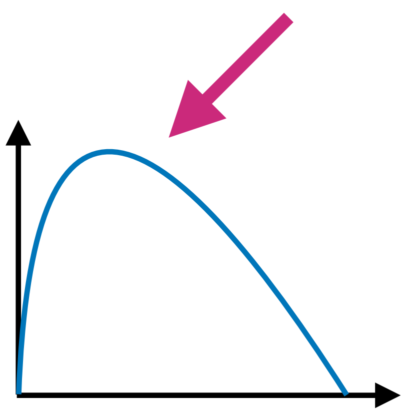


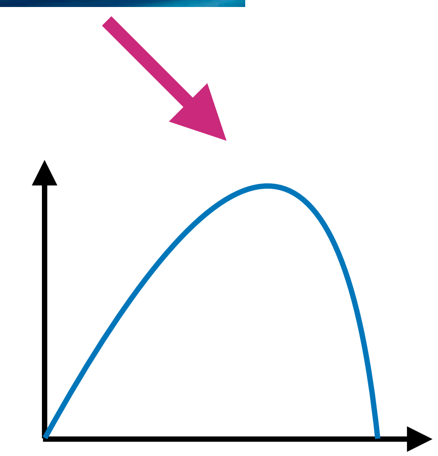
玻尔兹曼分布：样本生成困难,配分函数难以计算





玻尔兹曼分布：样本生成困难,配分函数难以计算

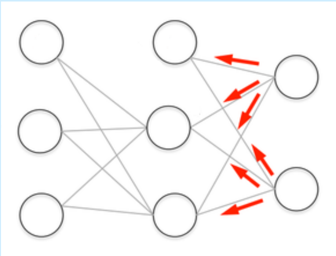




从Hinton的主要工作看神经网络学习发展

生成学习萌芽

酝酿期

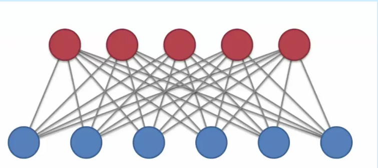


Back-propagation
1986

Deep belief network
Autoencoder
Pre-training
Fine-tune
2006

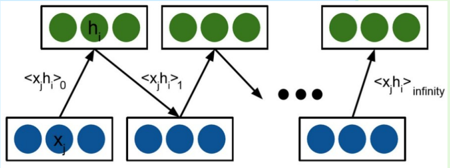
Alex net
2012

1985
Boltzmann
Machine

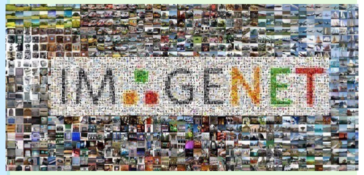


1995
Helmholtz
Machine

2002
Con. Div.
Product of Experts



2009
ImageNet

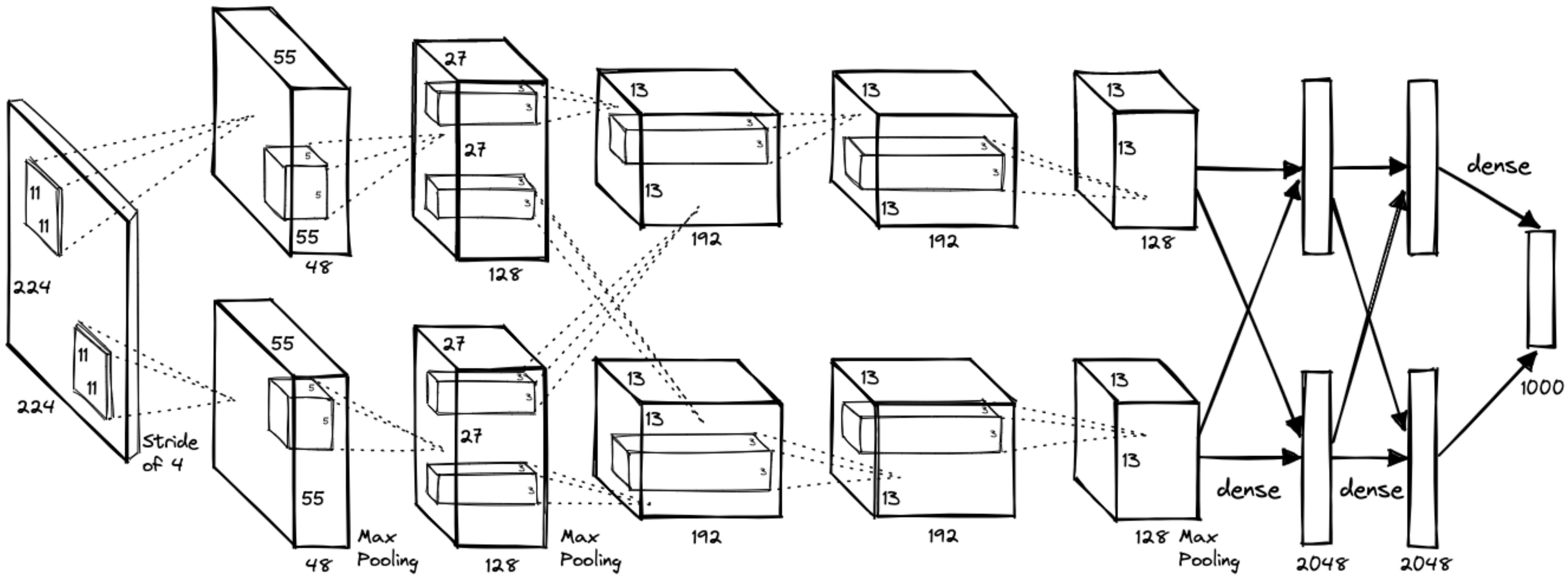


ImageNet

"… while a lot of people are paying attention to models, let's pay attention to data. Data will redefine how we think about models." – *Fei-Fei Li*

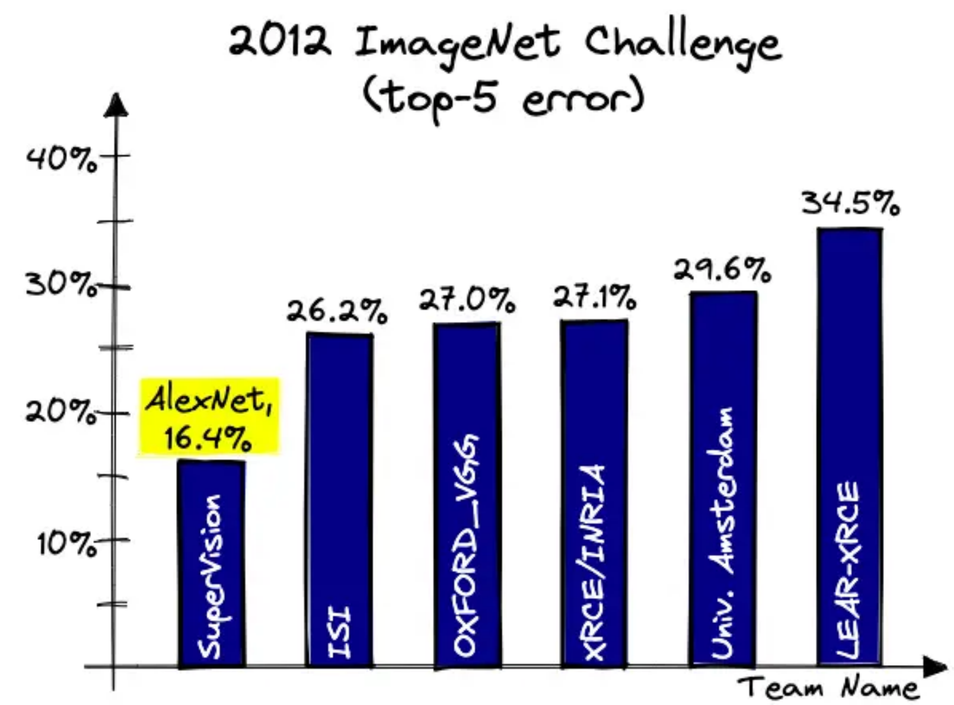
- WordNet + 人工标签 (Amazon众包服务 + 标签错误矫正)
- 2009: 12 million images across 22,000 categories
- 2010: 第一届 ImageNet Challenge

AlexNet: The birth of Deep Learning



Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton, 2012

- Convolution network + SGD + ImageNet
- Trained across 2 GPUs (Nvidia GTX 580)
- ReLu activation function
- Data Augmentation, Dropout

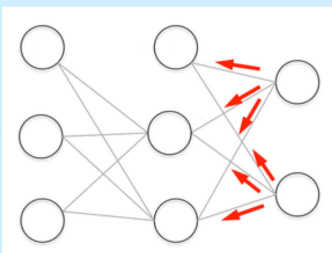


Era of Deep Learning

生成学习萌芽

酝酿期

Deep Learning



Back-propagation
1986

Deep belief network
Autoencoder
Pre-training
Fine-tune
2006

Alex net
2012

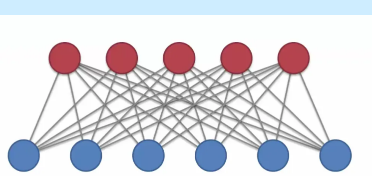
ResNet
2015

Diffusion
Model
2015

Transformer
2017

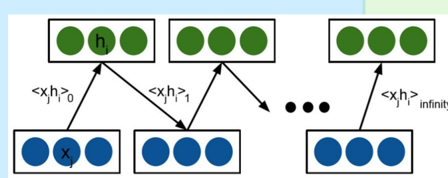
ChatGPT
2022

1985
Boltzmann
Machine



1995
Helmholtz
Machine

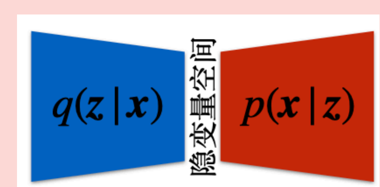
2002
Con. Div.
Product of Experts



2009
ImageNet

2014
GAN
VAE

2015
Flow
models



2016
AlphaGo

2018-2020
AlphaFold

Hinton的insight和信心(坚持)

The Forward-Forward Algorithm: Some Preliminary Investigations

Geoffrey Hinton
Google Brain
geoffhinton@google.com

2022年NeurIPS
Hinton 75岁

Abstract

The aim of this paper is to introduce a new learning procedure for neural networks and to demonstrate that it works well enough on a few small problems to be worth further investigation. The Forward-Forward algorithm replaces the forward and backward passes of backpropagation by two forward passes, one with positive (*i.e.* real) data and the other with negative data which could be generated by the network itself. Each layer has its own objective function which is simply to have high goodness for positive data and low goodness for negative data. The sum of the squared activities in a layer can be used as the goodness but there are many other possibilities, including minus the sum of the squared activities. If the positive and negative passes could be separated in time, the negative passes could be done offline, which would make the learning much simpler in the positive pass and allow video to be pipelined through the network without ever storing activities or stopping to propagate derivatives.

1 What is wrong with backpropagation

The astonishing success of deep learning over the last decade has established the effectiveness of performing stochastic gradient descent with a large number of parameters and a lot of data. The

Hinton的insight和信心(坚持)

Recent Papers

Hinton, G. E. (2022)
The Forward-Forward Algorithm: Some Preliminary Investigations
arXiv:2212.13345
[pdf of final version]
[ffcode.zip matlab code for the supervised version of FF v
[load mnistdata.mat in matlab to create the data]
[README.txt explains what to do to run FF}
Sindy Loewe's translation to python code is available at h

Chen, T., Zhang, R., & Hinton, G. (2022)
Analog bits: Generating discrete data using diffusion moc
arXiv preprint arXiv:2208.04202 [pdf]

Ren, M., Kornblith, S., Liao, R., & Hinton, G. (2022)
Scaling Forward Gradient With Local Losses
arXiv preprint arXiv:2210.03310 [pdf]

MATLAB CODE

.... Matlab for Science paper

....t-SNE software

....trajectory from motor program

....ink from trajectory

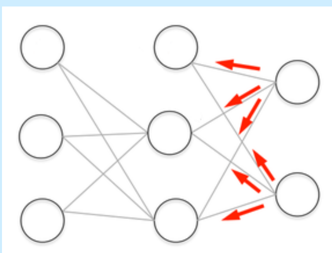
....introduction to python

Neural Networks and Statistical Physics ???

生成学习萌芽

酝酿期

Deep Learning



Back-propagation
1986

Deep belief network
Autoencoder
Pre-training
Fine-tune
2006

Alex net
2012

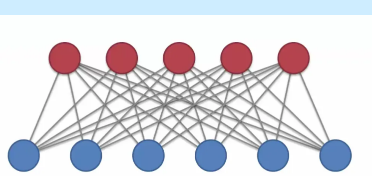
ResNet
2015

Diffusion Model
2015

Transformer
2017

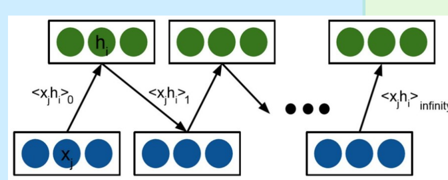
ChatGPT
2022

1985
Boltzmann Machine



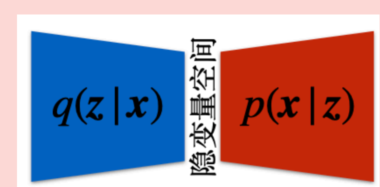
1995
Helmholtz Machine

2002
Con. Div.
Product of Experts



2009
ImageNet

2014
GAN
VAE

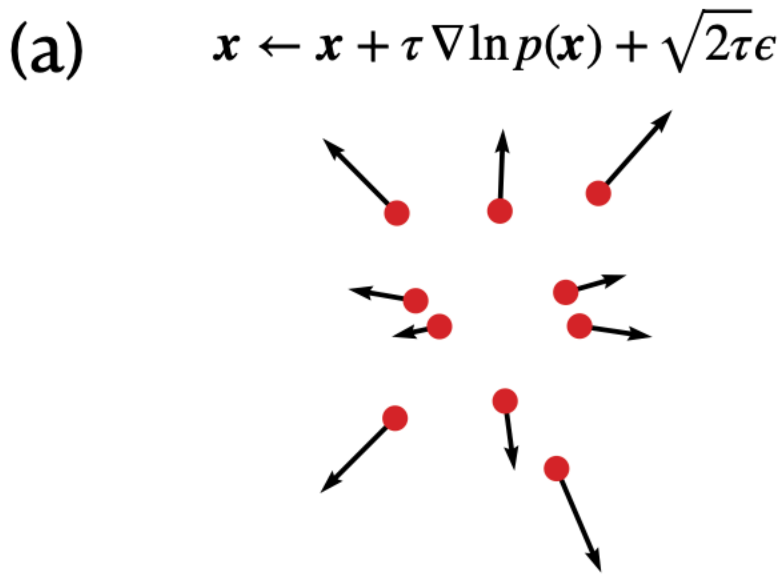


2015
Flow models

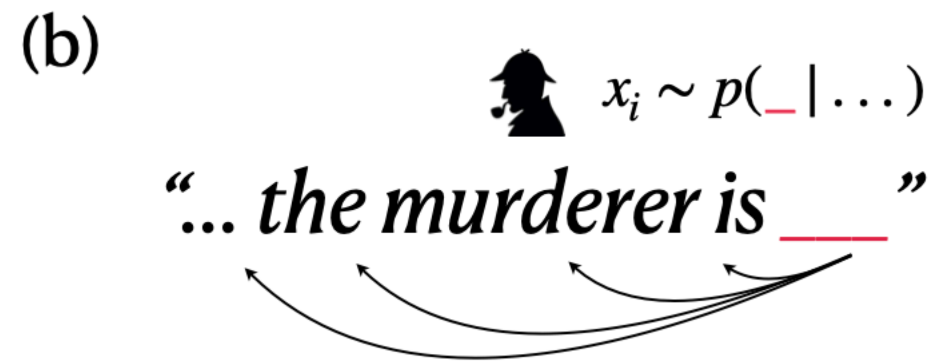
2016
AlphaGo

2018-2020
AlphaFold

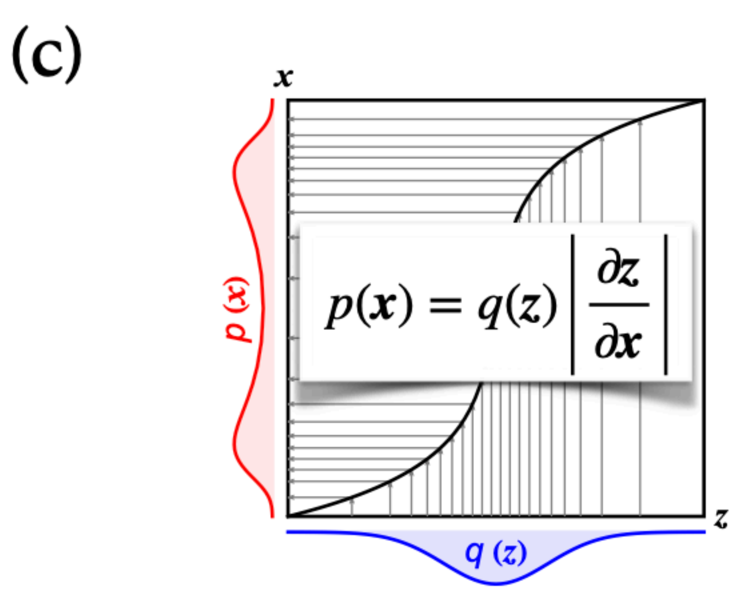
现代生成模型有统计物理基因，但直接关系不强



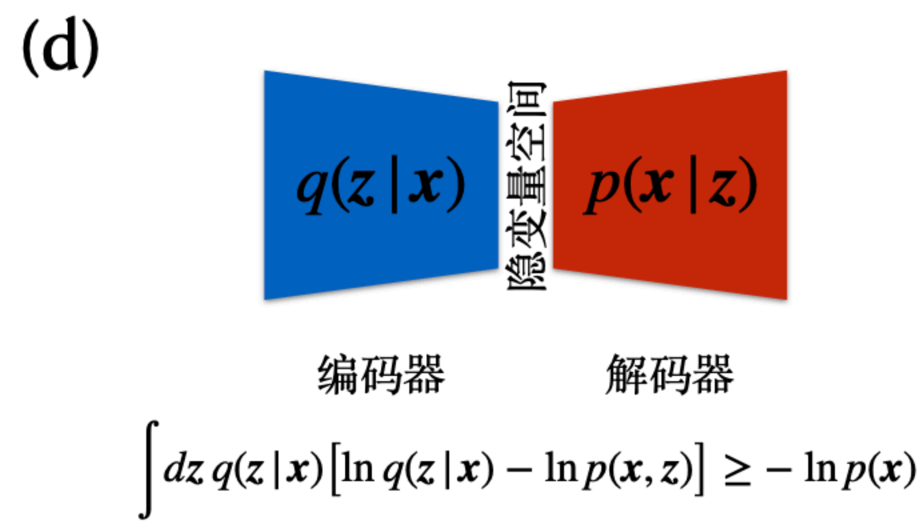
Diffusion models



Autoregressive models



Flow models



Variational autoencoder

王磊，张潘，《写给物理学家的生成模型》，“物理” 2024

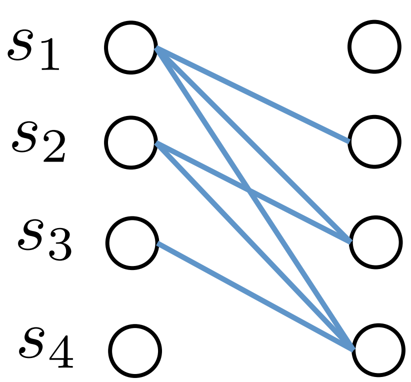
Auto-regressive distribution

- Representing joint distribution using chain rule of conditional probabilities.

$$q(\mathbf{s}) = \prod_i q(s_i|\mathbf{s}_{j<i})$$

$$\begin{aligned} q(s_1, s_2, s_3, s_4) &= q(s_4|s_3, s_2, s_1)q(s_3, s_2, s_1) \\ &= q(s_4|s_3, s_2, s_1)q(s_3|s_2, s_1)q(s_2, s_1) \\ &= q(s_4|s_3, s_2, s_1)q(s_3|s_2, s_1)q(s_2|s_1)q(s_1) \end{aligned}$$

Fully Visible Belief Network [Frey 1998]



$$\hat{s}_1 = \text{sigmoid}(0) = q(s_1 = 1)$$

$$\hat{s}_2 = \text{sigmoid}(w_{12}s_1) = q(s_2 = 1|s_1)$$

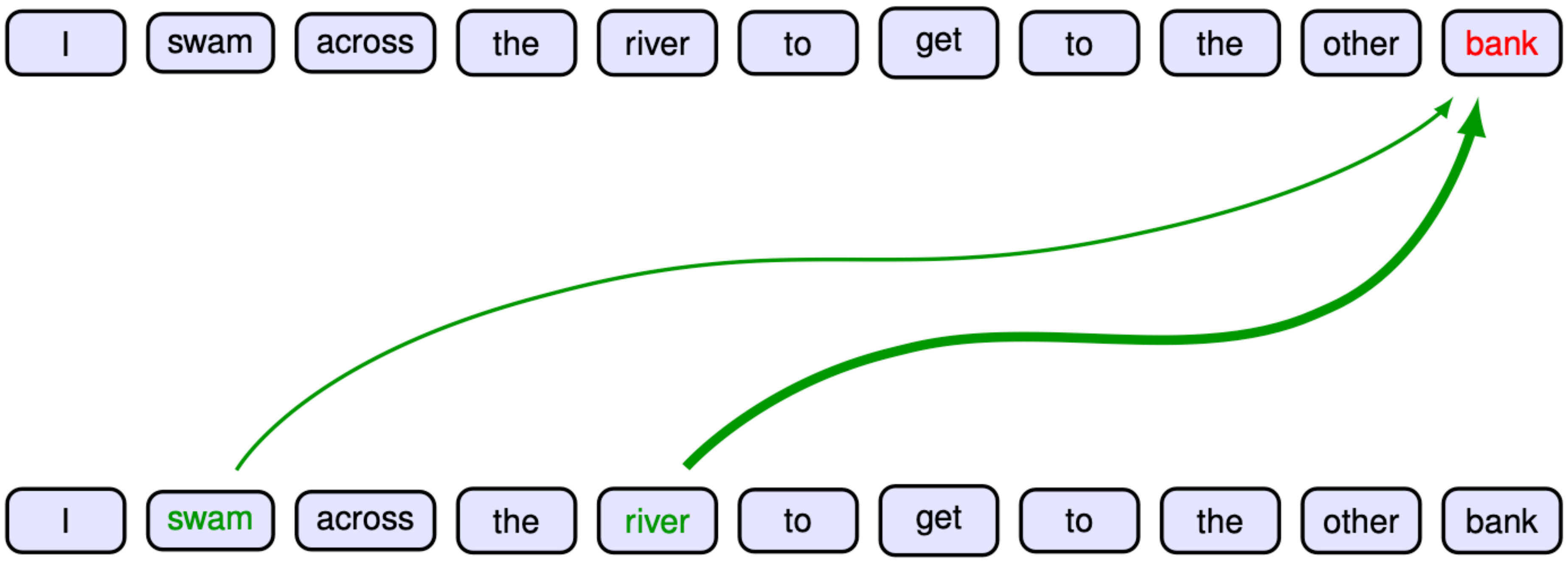
$$\hat{s}_3 = \text{sigmoid}(w_{13}s_1 + w_{23}s_2) = q(s_3 = 1|s_2, s_1)$$

$$\hat{s}_4 = \text{sigmoid}(w_{14}s_1 + w_{24}s_2 + w_{34}s_4) = q(s_4 = 1|s_3, s_2, s_1)$$

$$q(s1, s2, s3, s4) = \hat{s}_1\hat{s}_2\hat{s}_3\hat{s}_4$$

- *conditional probabilities* $\Longrightarrow$ Directed Sampling
  Known as ***ancestral sampling*** [*Bishop 2006*]

Key problem： data-related long-range correlations



I swam across the river to get to the other bank.
I walked across the road to get cash from the bank.

The meaning of bank depends on the words of previous positions.

The positions may vary in various sentences.

Self-attention

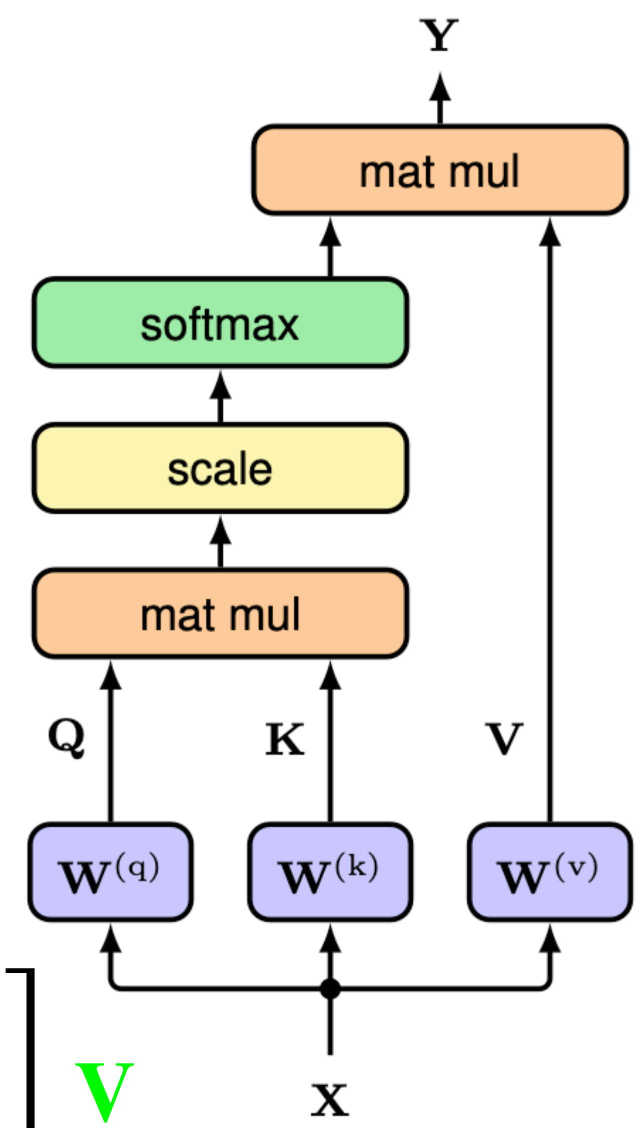
Focus on some features

$$\widetilde{\mathbf{Q}} = \mathbf{X}\mathbf{W}^{(\mathbf{q})}$$

$$\widetilde{\mathbf{K}} = \mathbf{X}\mathbf{W}^{(\mathbf{k})}$$

$$\widetilde{\mathbf{V}} = \mathbf{X}\mathbf{W}^{(\mathbf{v})}$$

$$Y = \mathrm{Softmax}[\mathbf{Q}\mathbf{K}^\top]\mathbf{V}$$

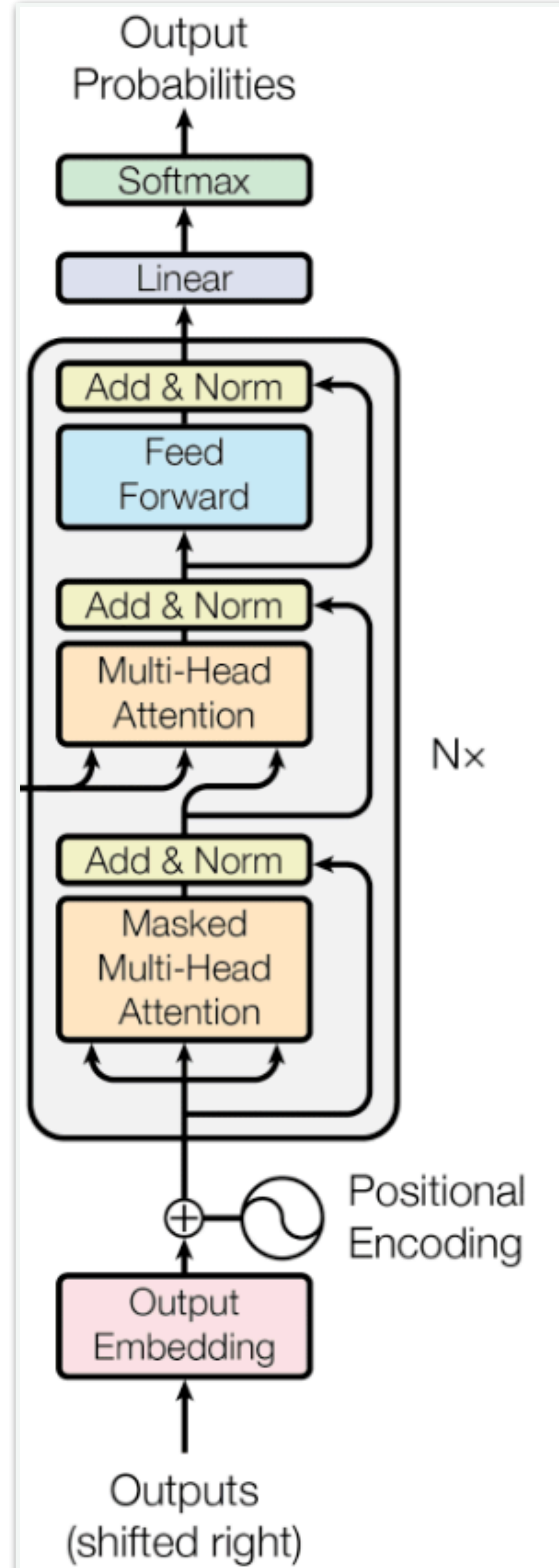


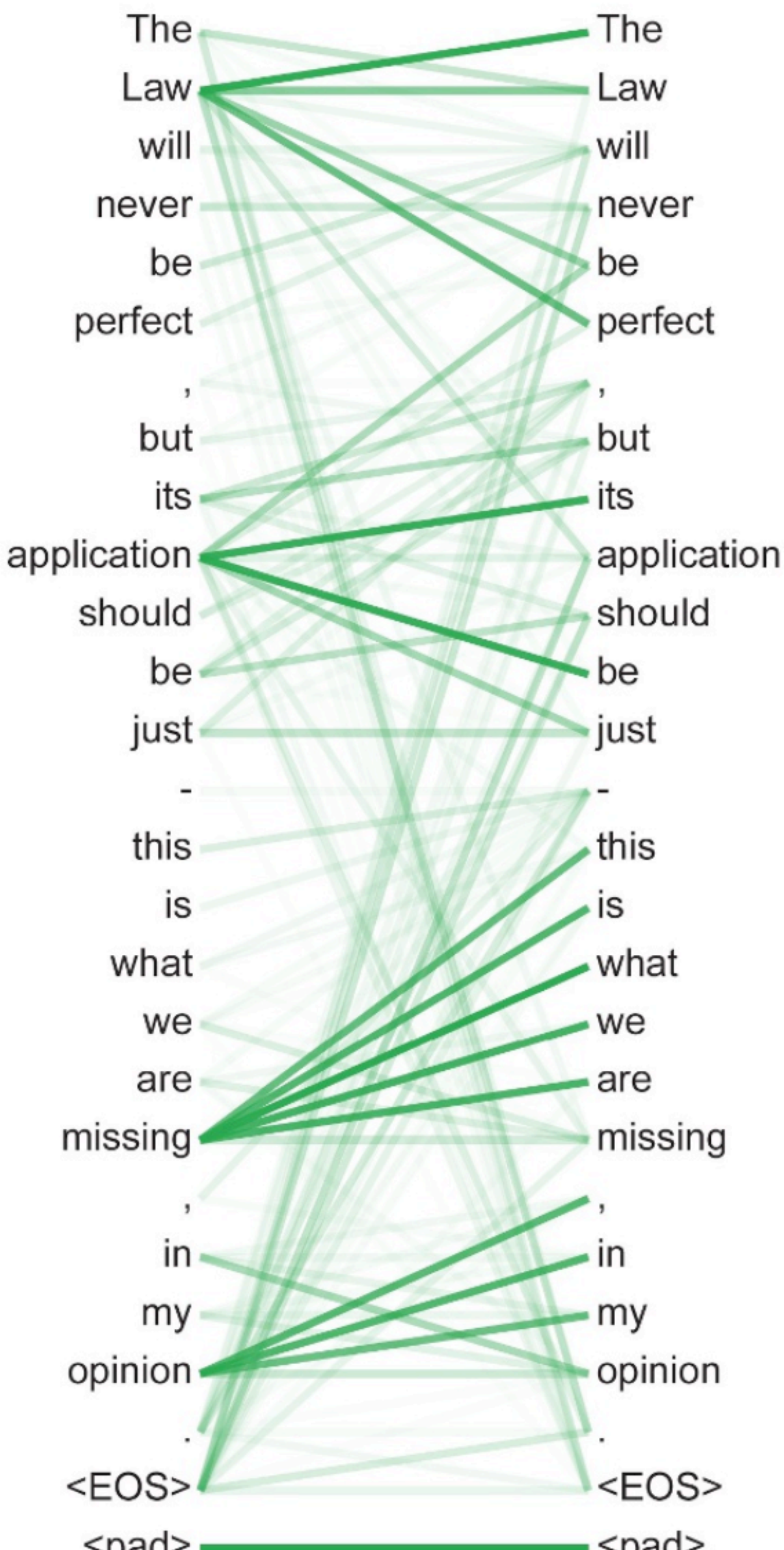
$$\mathbf{Y} = \mathrm{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{Softmax}\left[\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{D_k}}\right]\mathbf{V}$$

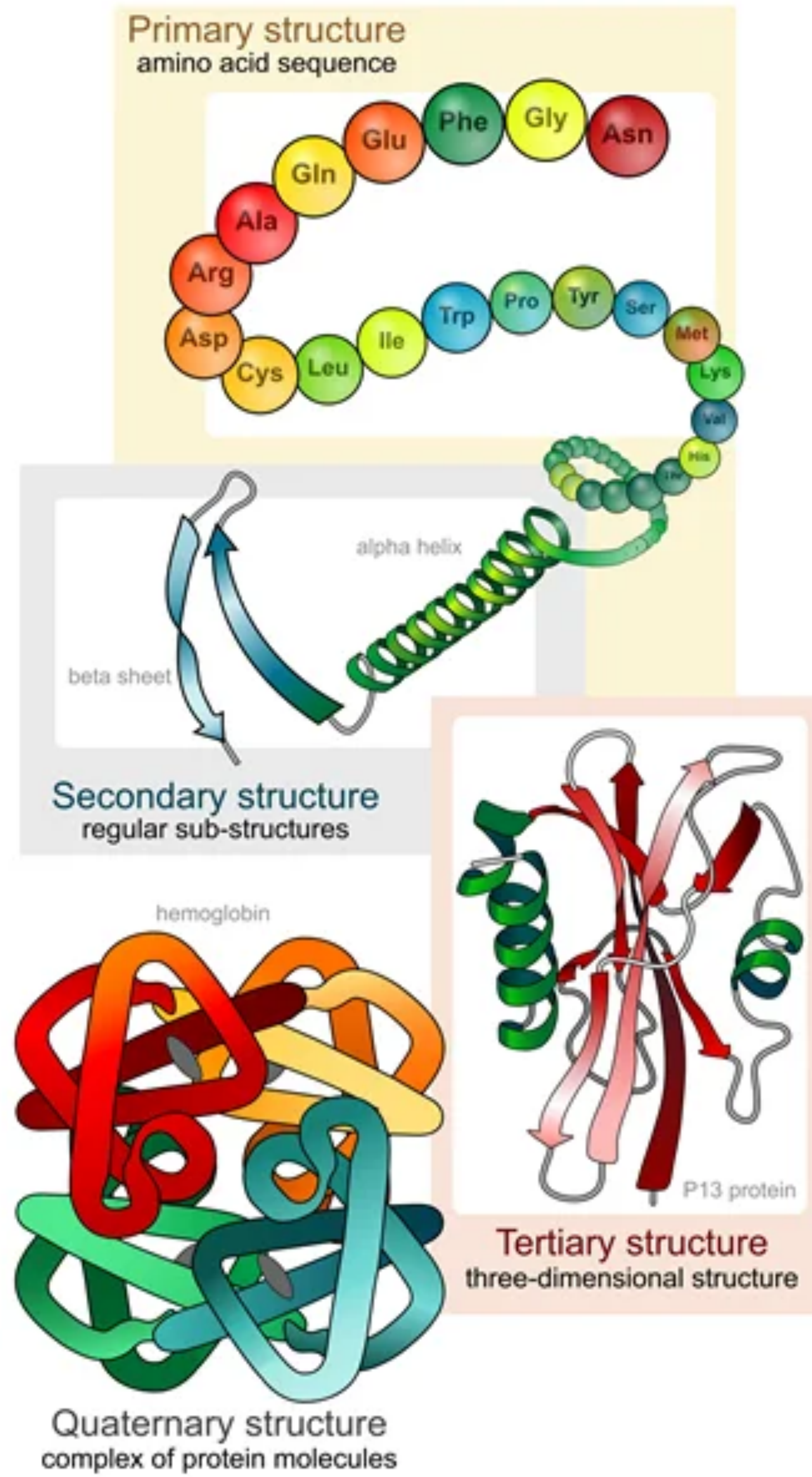
Data-dependent attention coefficients

GPT in 60 Lines of NumPy: https://github.com/jaymody/picoGPT/blob/main/gpt2_pico.py

```python
def gelu(x): return 0.5 * x * (1 + np.tanh(np.sqrt(2 / np.pi) * (x + 0.044715 * x**3)))
def softmax(x):
    exp_x = np.exp(x - np.max(x, axis=-1, keepdims=True))
    return exp_x / np.sum(exp_x, axis=-1, keepdims=True)

def layer_norm(x, g, b, eps: float = 1e-5):
    mean = np.mean(x, axis=-1, keepdims=True)
    variance = np.var(x, axis=-1, keepdims=True)
    return g * (x - mean) / np.sqrt(variance + eps) + b

def linear(x, w, b): return x @ w + b
def ffn(x, c_fc, c_proj): return linear(gelu(linear(x, **c_fc)), **c_proj)
def attention(q, k, v, mask): return softmax(q @ k.T / np.sqrt(q.shape[-1]) + mask) @ v

def mha(x, c_attn, c_proj, n_head):
    x = linear(x, **c_attn)
    qkv_heads = list(map(lambda x: np.split(x, n_head, axis=-1), np.split(x, 3, axis=-1)))
    causal_mask = (1 - np.tri(x.shape[0], dtype=x.dtype)) * -1e10
    out_heads = [attention(q, k, v, causal_mask) for q, k, v in zip(*qkv_heads)]
    x = linear(np.hstack(out_heads), **c_proj)
    return x

def transformer_block(x, mlp, attn, ln_1, ln_2, n_head):
    x = x + mha(layer_norm(x, **ln_1), **attn, n_head=n_head)
    x = x + ffn(layer_norm(x, **ln_2), **mlp)
    return x

def gpt2(inputs, wte, wpe, blocks, ln_f, n_head):
    x = wte[inputs] + wpe[range(len(inputs))]
    for block in blocks:
        x = transformer_block(x, **block, n_head=n_head)
    return layer_norm(x, **ln_f) @ wte.T

def generate(inputs, params, n_head, n_tokens_to_generate):
    from tqdm import tqdm
    for _ in tqdm(range(n_tokens_to_generate), "generating"):
        logits = gpt2(inputs, **params, n_head=n_head)
        next_id = np.argmax(logits[-1])
        inputs.append(int(next_id))
    return inputs[len(inputs) - n_tokens_to_generate :]
```




The
Law
will
never
be
perfect
,
but
its
application
should
be
just
-
this
is
what
we
are
missing
,
in
my
opinion
.
<EOS>
<pad>
The
Law
will
never
be
perfect
,
but
its
application
should
be
just
-
this
is
what
we
are
missing
,
in
my
opinion
.
<EOS>
<pad>


Primary structure
amino acid sequence
Phe
Gly
Glu
Asn
Gln
Ala
Arg
Trp
Pro
Tyr
Ser
Asp
Ile
Met
Cys
Leu
Lys
alpha helix
beta sheet
Secondary structure
regular sub-structures
hemoglobin
P13 protein
Tertiary structure
three-dimensional structure
Quaternary structure
complex of protein molecules

统计物理可否解释机器学习？

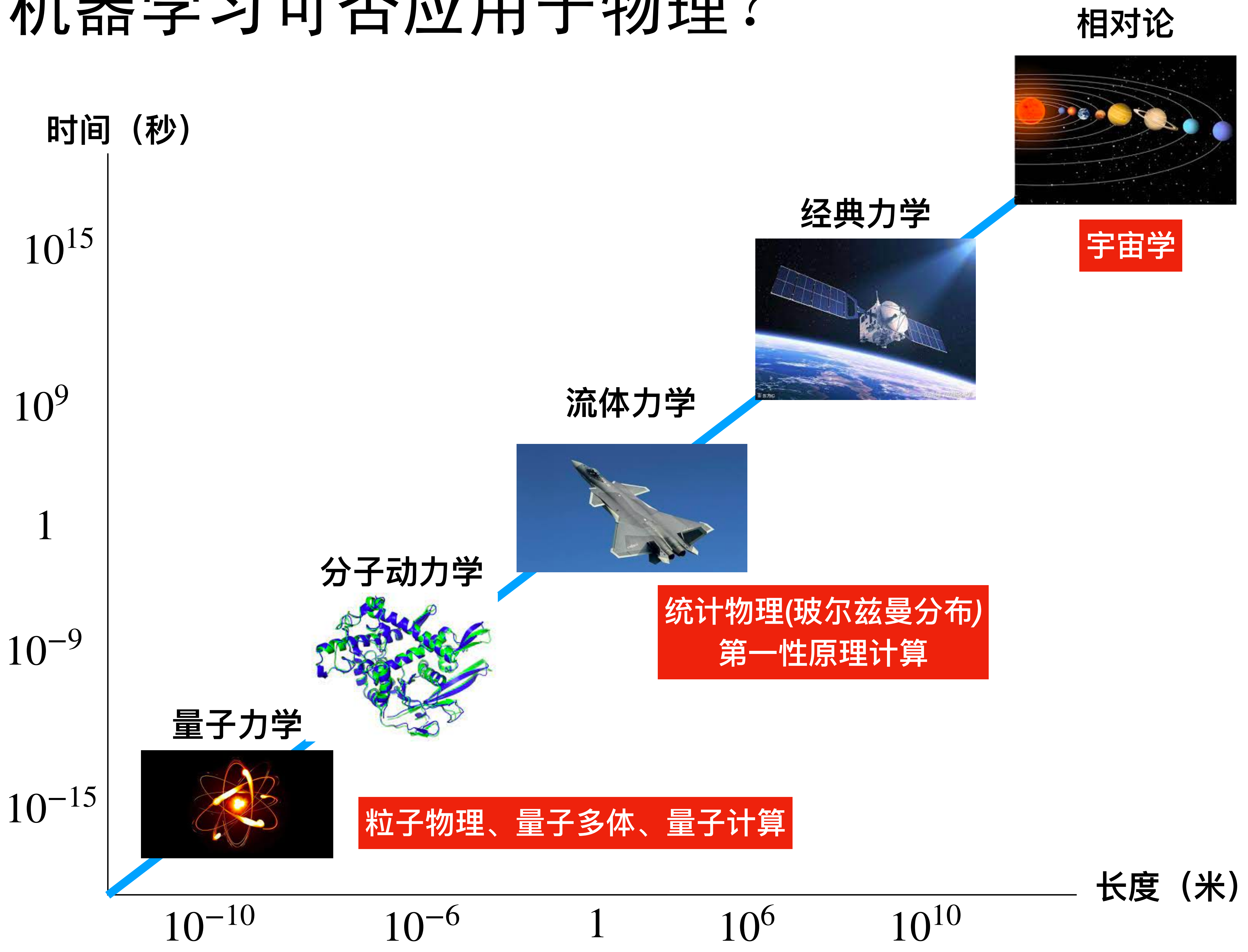
Strengths (Spin-Glass theory, Repilca method, mean-field, message passing)：

- Hopfield模型相图，动力学 [*Amit et al 1985, Coolen 2001*]
- Perceptron and generalized linear model [*Krauth, Mezard 1987, Barbier 2019*]
- 浅层、随机、线性网络相图, error landscape [*Saxe et al 2013, Gabrié et al 2019*]
- Information Bottleneck [*Tishby et al, 2000*]
- Glassiness, overparameterization *[Baity-Jesi et al., 2018, Baldassi et al., 2016*]

Weaknesses：

- 简单模型、随机模型、无限宽模型
- 难以考虑复杂数据
- 难以定量描述泛化性

机器学习可否应用于物理？



机器学习在统计物理中的应用

- 识别物质相，相变 *(Wang 2016, Carrasquilla, Mello 2017)*
  - 监督学习、非监督学习，序参量
- 统计力学，重整化 *(Li,Wang 2018, Wu et al 2019)*
  - 变分自由能计算，强化学习，采样，帮助MCMC?
- 非平衡系统 *(Tang et al 2023)*
  - 时间演化，动力学相变
- 非线性动力学系统 *(Pathak et al 2017)*
  - 预测，控制非线性系统，Reservoir 网络

机器学习在量子多体中的应用

- 多体态分类

  - 多体局域化，哈密顿量、纠缠谱 *(Hsu et al 2018)*

  - non-local序参量，拓扑不变形 *(Zhang, Kim 2017)*

- 神经网络量子态 （*Carleo, Troyer 2017)*

  - 用RBM、MLP、Autoregressive网络表达波函数

  −变分蒙卡、强化学习

  - 內禀对称性(平移、交换，backflow *Luo, Clar 2018*)

- 自动微分赋能张量网络 *(Liao et al 2019)*

  −时间演化，动力学相变

- 张量网络与监督、非监督学习

  - 线性分类器 *(Stoudenmire, schwab 2016*)

  − *MPS*玻恩机 Born Machine *(Han et al, 2018)*

机器学习与第一性原理计算、物质生成

分子结构、能量面：分子动力学模拟

- 从DFT数据学习能量和力场 *(Zhang et al 2018)*
- 自由能surface，集体变量 *(Noe 2019)*

材料性质预测

电子密度与DFT：创造新的density functional *(Nagi et al 2018)*

物质生成：从数据生成到原子、分子生成

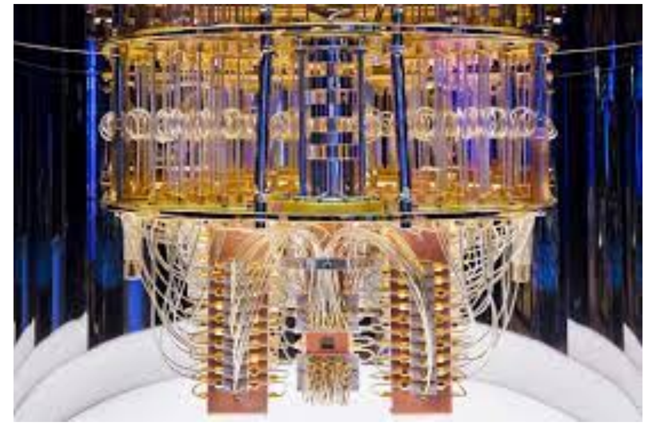
- 利用对称性，CrystalFormer *(Cao,Luo,Lv,Wang 2024)*

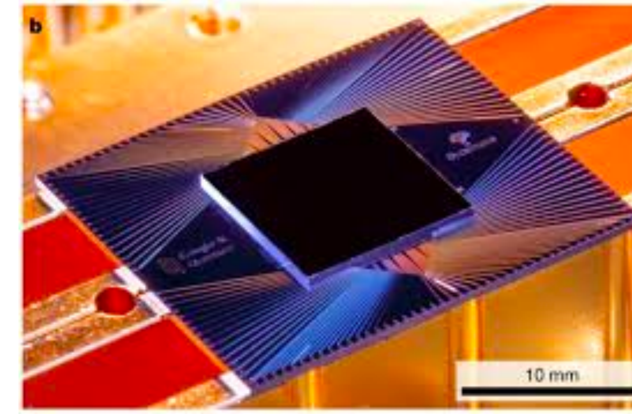
机器学习在粒子物理、宇宙学中的应用

- LHC，LSST，LIGO等大科学装置有大量数据需要处理
- 需要用量子场论、微扰方法、广义相对论得到大量模拟数据
  - –学习神经网络用于快速模拟，生成数据
  - - 快速判断是否存在（引力波）信号、LHC的Trigger系统
  - –从数据学习神经网络，用于推断模型参数
- Jet 物理: jet标记、flavor 标记、jet聚类、spectral density estimation
- 中微子物理：CNN用于信号处理，寻找中微子作用位置
- 引力波物理：信号分类，广义相对论模型参数估计
- LatticeQCD: Hamilton MCMC获取组态代价极高
  - - 利用flow model等生成模型也许可以提供好的proporsal

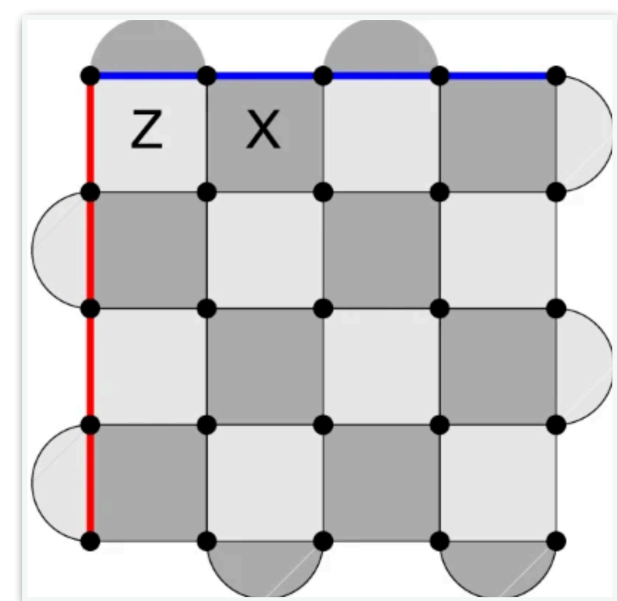
机器学习在量子计算中的应用

- 量子机器学习：
  - 量子数据表示
  - 量子算法设计
  - 量子优势
  - 混合量子-经典算法





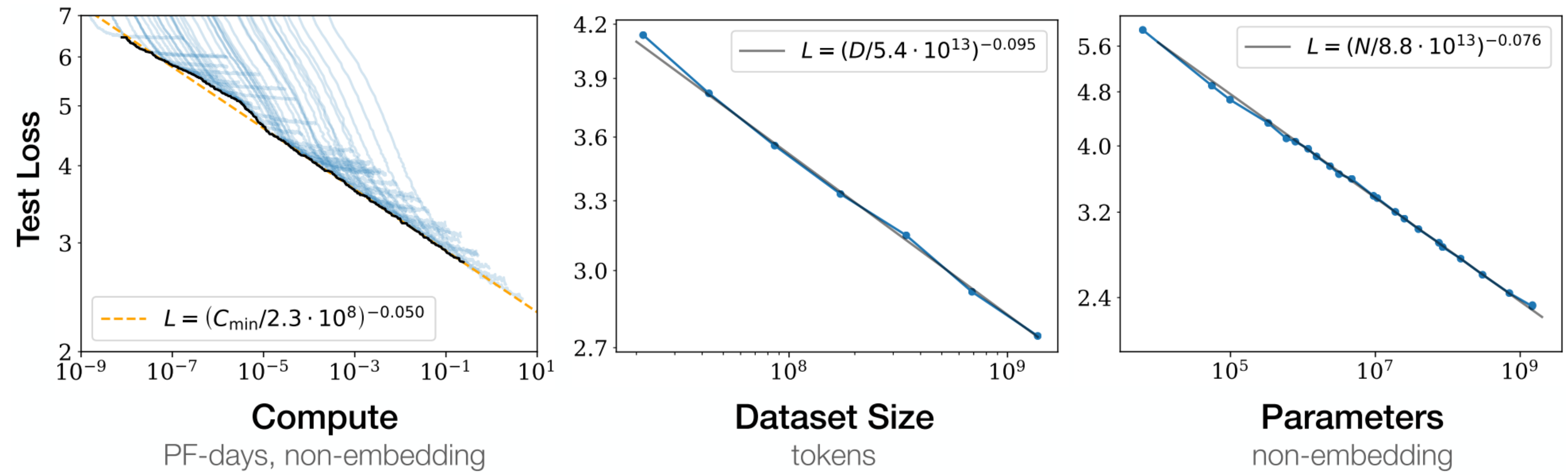
- 量子线路优化：减少线路深度，优化量子门参数
- Quantum State Tomography：相比于张量网络QST，可以表述纠缠更强的密度矩阵
- 量子纠错：
  - 噪音建模（beyond depolarizing、SI1000，MWPM的prior）
  - 从数据中学习logical operator *(Google 2024, qecGPT)*



机器学习应用于物理的挑战与机遇

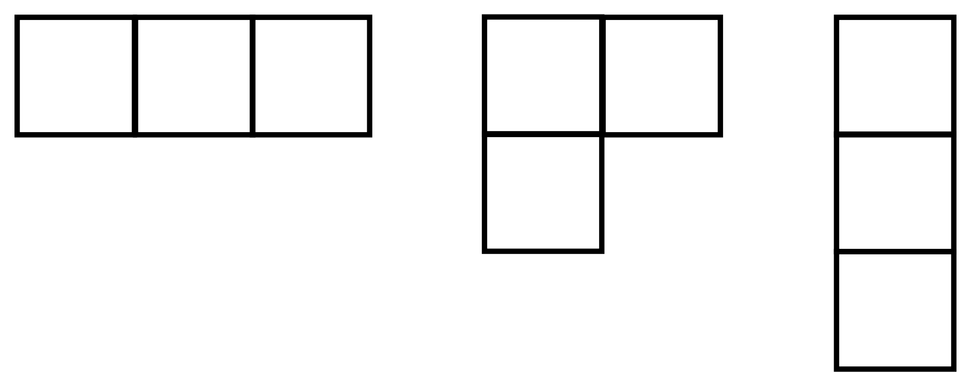
1. 数据量不够

   - 高质量数据获取昂贵



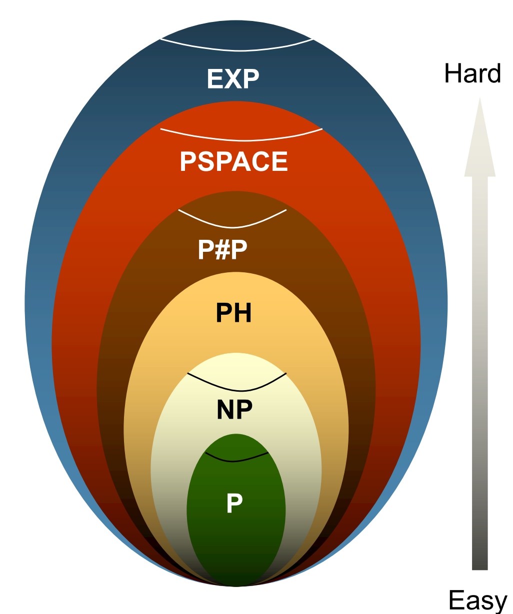
2. 尊重对称性等约束

   - 晶体点群，空间群
   - 费米子交换反对称性
   - … …



3. 需要贝叶斯主义，自由能原理

   - 玻尔兹曼分布
   - 参数推断 （引力波拟合，模型参数选择）



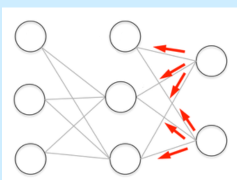
Machine Learning and Physics ?

生成学习
萌芽

酝酿

Deep Learning

Future



Back-propagation
1986

Deep belief network
Autoencoder
Pre-training
Fine-tune
2006

Alex net
2012

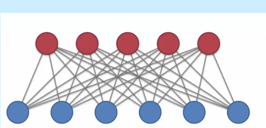
ResNet
2015

Diffusion
Model
2015

Transformer
2017

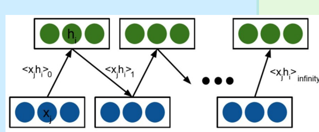
ChatGPT
2022

1985
Boltzmann
Machine



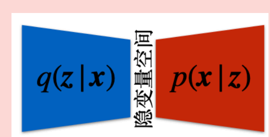
1995
Helmholtz
Machine

2002
Con. Div.
Product of Experts



2009
ImageNet

2014
GAN
VAE



2015
Flow
models

2016
AlphaGo

2018-2020
AlphaFold

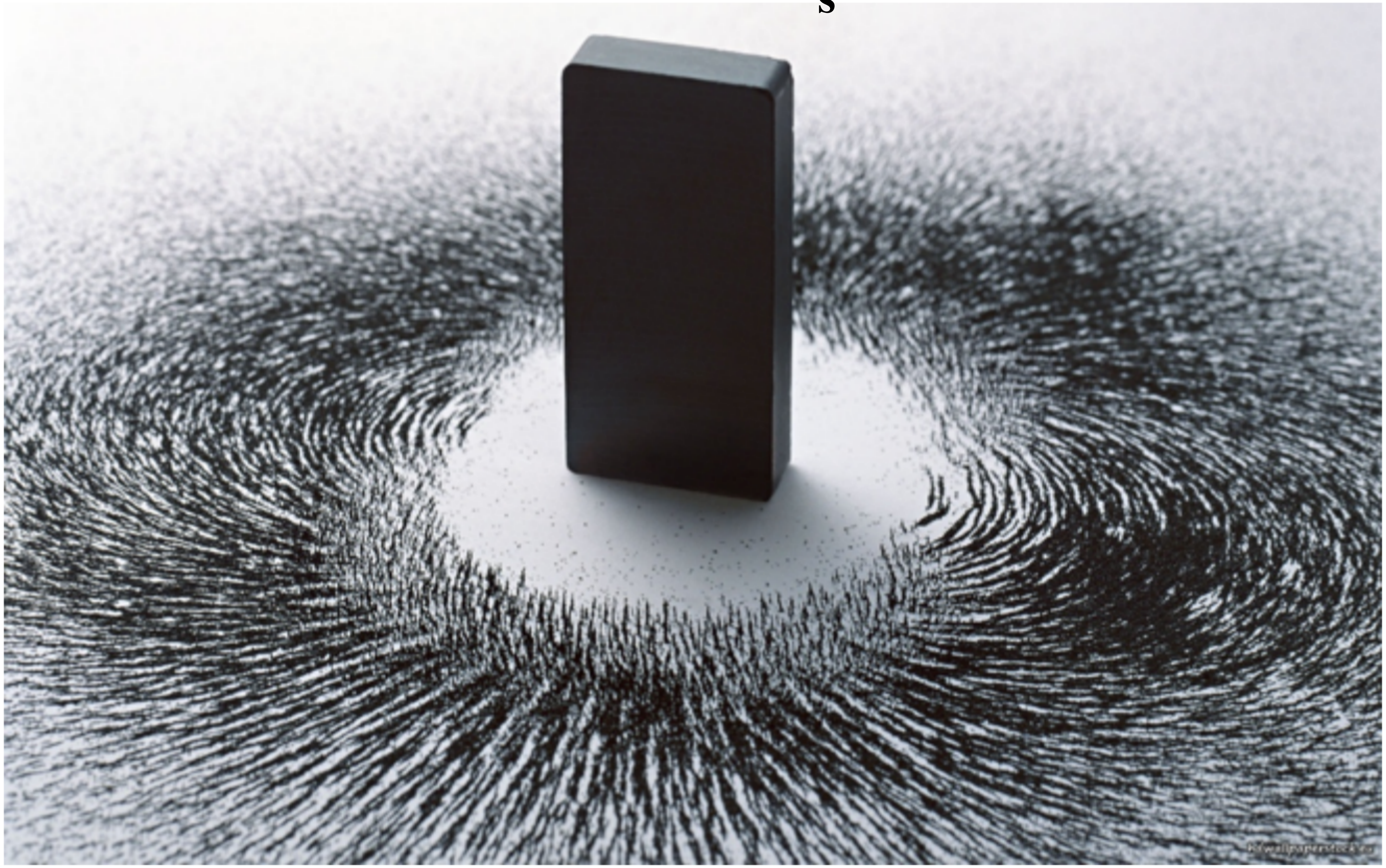
2024
Nobel
Prize

Statistical Mechanics

$$\mathbf{s} = \{+1, -1\}^n$$

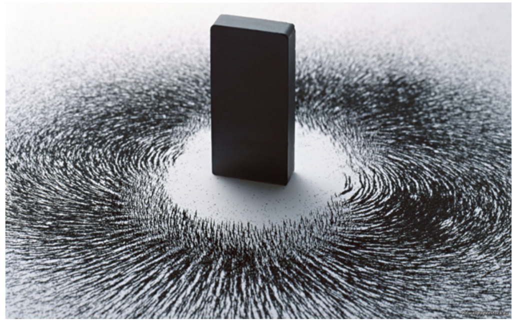
$$\uparrow \uparrow \uparrow \downarrow \uparrow \downarrow \downarrow \downarrow \downarrow \downarrow \uparrow \uparrow$$

$$P(\mathbf{s}) = \frac{1}{Z} e^{-\beta E(\mathbf{s})} \qquad Z = \sum_{\mathbf{s}} e^{-\beta E(\mathbf{s})}$$



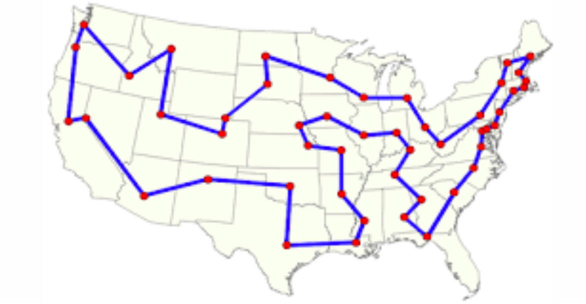
Applications of Statistical Mechanics

- In Physics: Thermodynamics, Phases / Phase transitions …

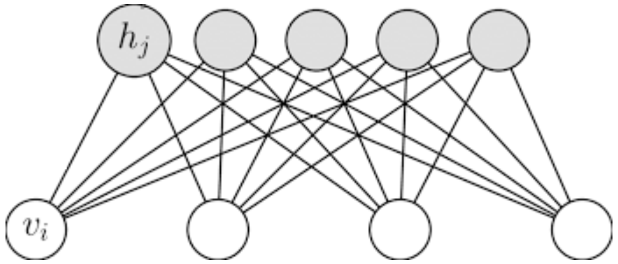


- In Combinatorial Optimization:

$$P(\mathbf{s}) = \frac{1}{Z} e^{-\beta E(\mathbf{s})} \text{ with } \beta \to \infty$$

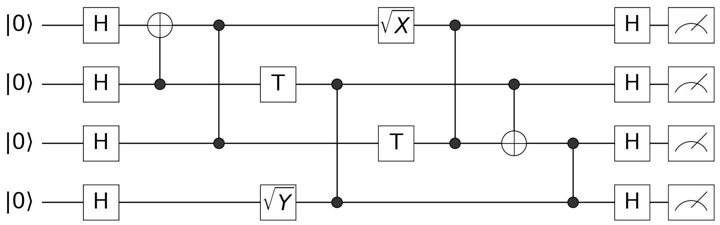


- In machine learning: associative memory
  - Hopfield model
  - Boltzmann machines



- In Statistical Inference:
  - Bayesian Inference and Max. A. Posterior

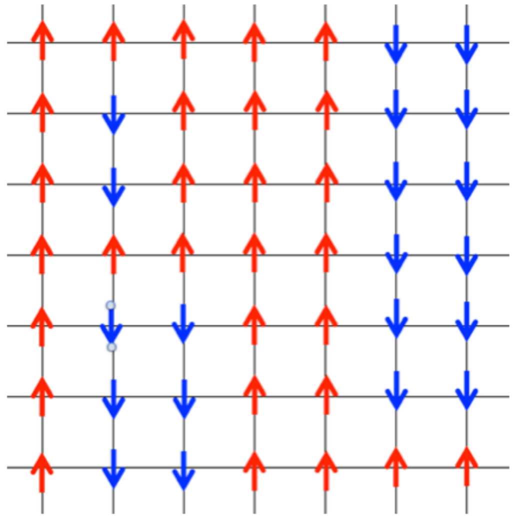
- In quantum computation
  - quantum error correction



… …

统计物理与机器学习

Parisi
2021年物理诺奖



微观构型的联合分布

$$P(\sigma) = \frac{1}{Z}\exp(-\beta E(\sigma))$$

数目变量的联合分布

Hinton
2024年物理诺奖

$P(\text{Data})$

Hopfield
2024年物理诺奖

指数大的空间
有效的方法
强大的计算能力

玻尔兹曼分布：样本生成困难，配分函数计算困难

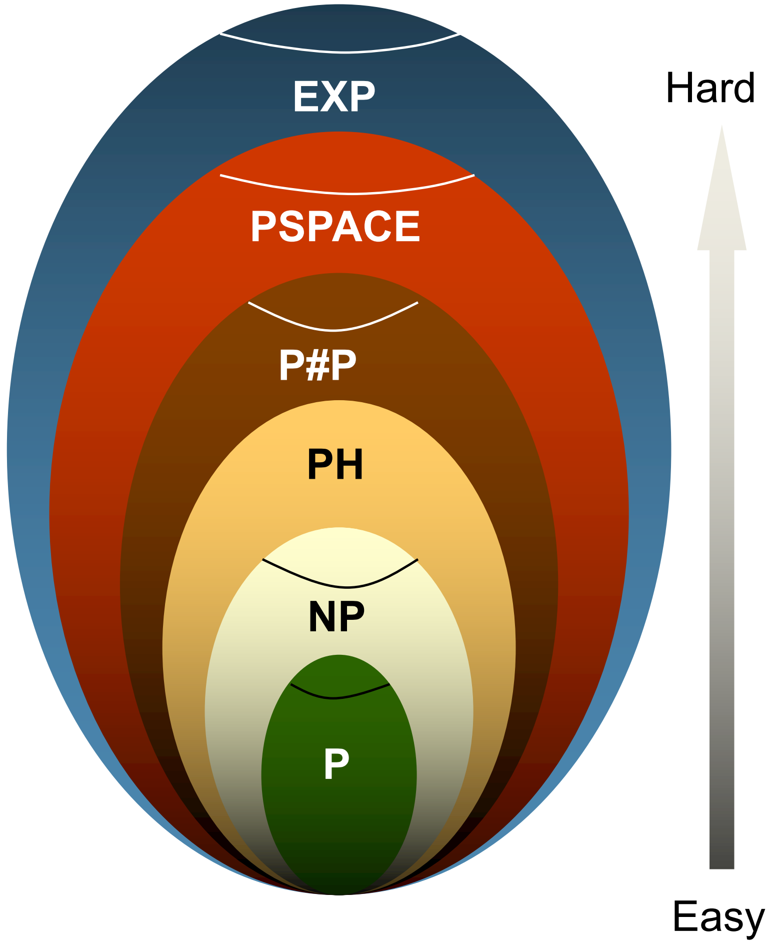
$$\mathbf{S} = \{+1, -1\}^n$$

↑↑↑↓↑↓↓↓↓↓↑↑

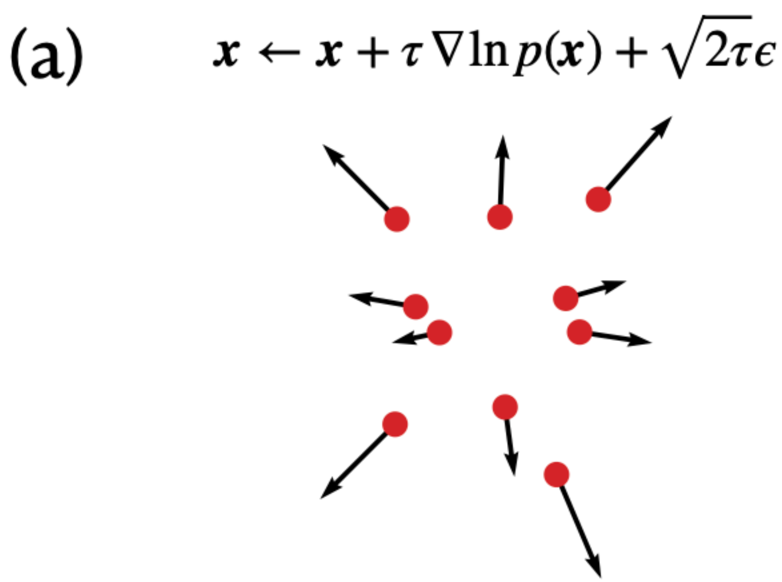
$$P(\mathbf{S}) = \frac{1}{Z} \mathrm{e}^{-\beta E(\mathbf{S})}$$

$$Z = \sum_{\mathbf{S}} \mathrm{e}^{-\beta E(\mathbf{S})}$$

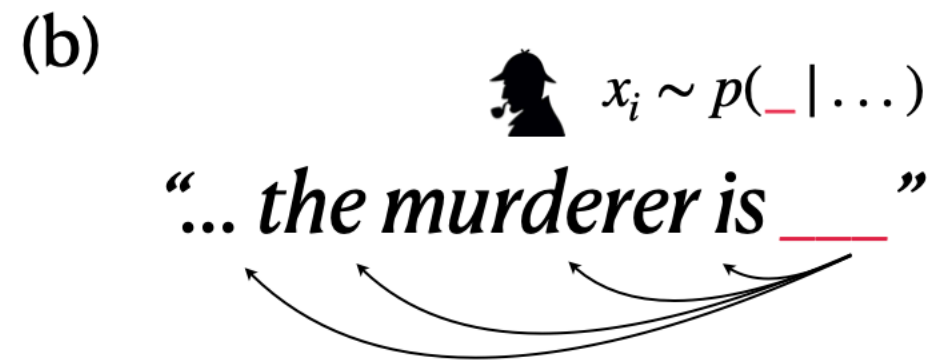
- 估计自由能
- 计算统计量/序参量
- 无偏采样



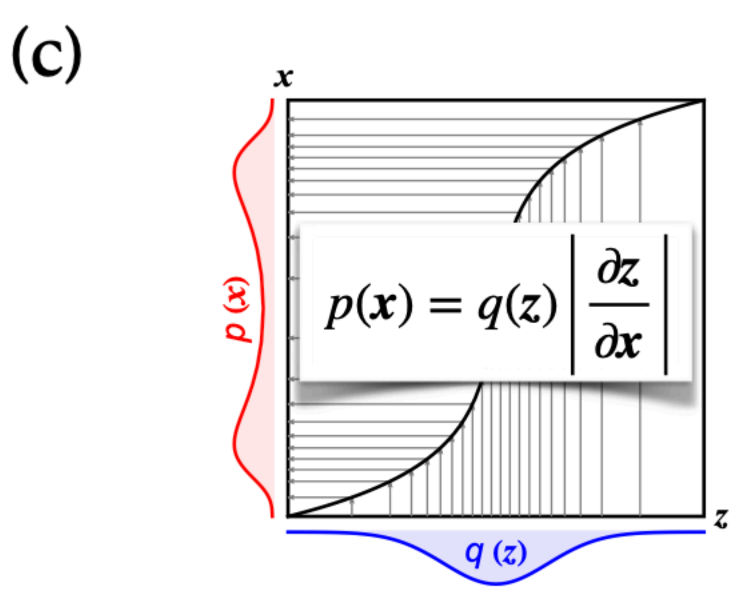
现代生成模型



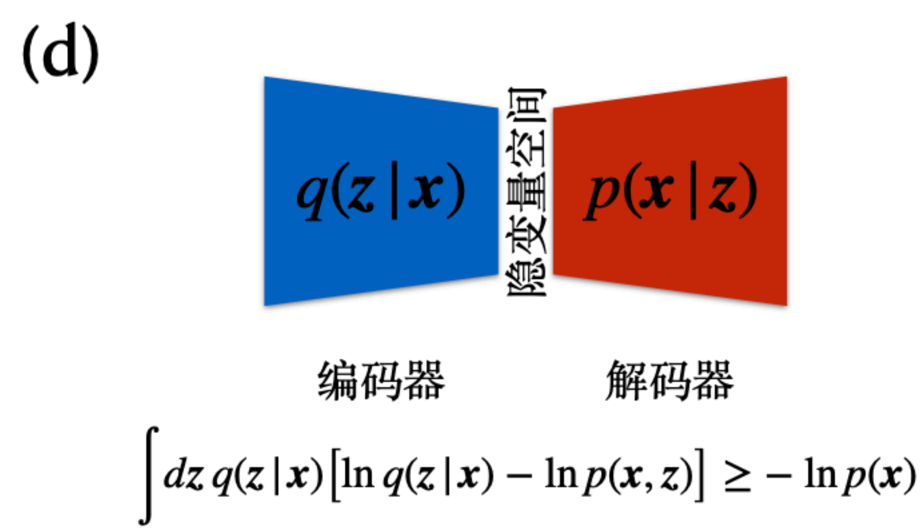
Diffusion models



Autoregressive models



Flow models



Variational autoencoder

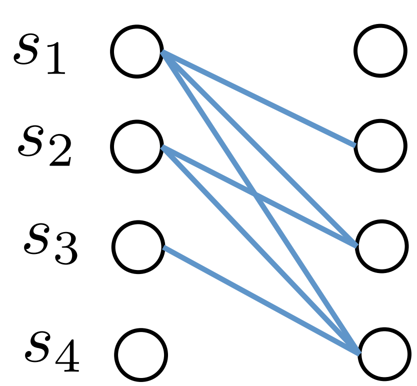
王磊，张潘，《写给物理学家的生成模型》，“物理” 2024

Auto-regressive distribution

$$q(\mathbf{s}) = \prod_i q(s_i|\mathbf{s}_{j<i})$$

$$\begin{aligned} q(s_1, s_2, s_3, s_4) &= q(s_4|s_3, s_2, s_1)q(s_3, s_2, s_1) \\ &= q(s_4|s_3, s_2, s_1)q(s_3|s_2, s_1)q(s_2, s_1) \\ &= q(s_4|s_3, s_2, s_1)q(s_3|s_2, s_1)q(s_2|s_1)q(s_1) \end{aligned}$$

Fully Visible Belief Network
[Frey 1998]



$$\hat{s}_1 = \mathrm{sigmoid}(0) = q(s_1 = 1)$$

$$\hat{s}_2 = \mathrm{sigmoid}(w_{12}s_1) = q(s_2 = 1|s_1)$$

$$\hat{s}_3 = \mathrm{sigmoid}(w_{13}s_1 + w_{23}s_2) = q(s_3 = 1|s_2, s_1)$$

$$\hat{s}_4 = \mathrm{sigmoid}(w_{14}s_1 + w_{24}s_2 + w_{34}s_4) = q(s_4 = 1|s_3, s_2, s_1)$$

$$q(s1, s2, s3, s4) = \hat{s}_1\hat{s}_2\hat{s}_3\hat{s}_4$$

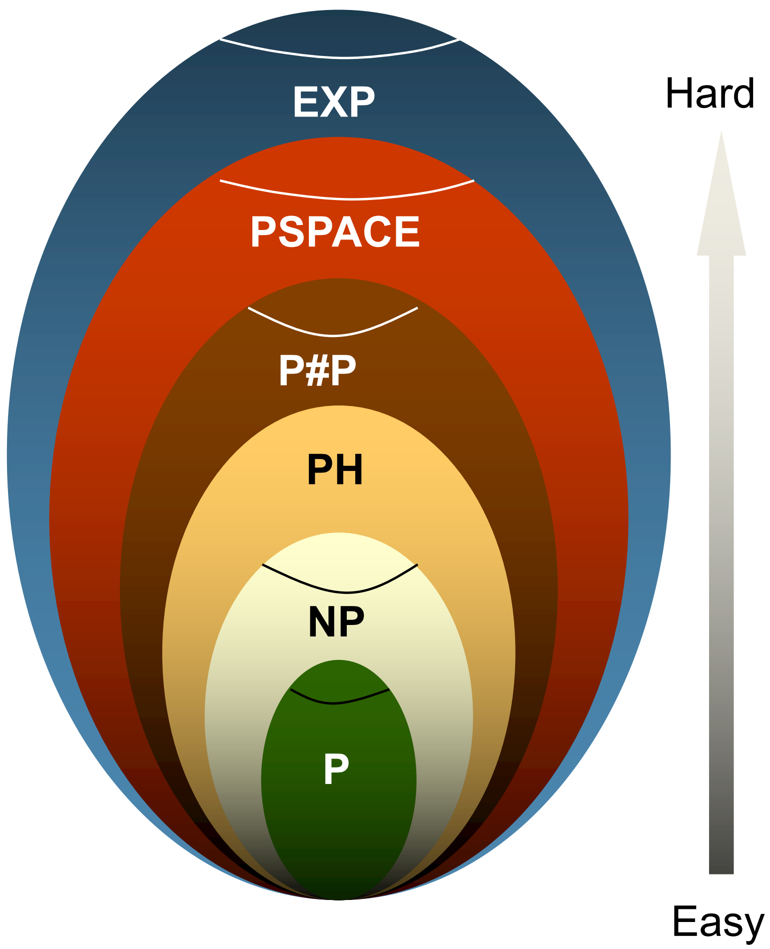
无偏采样：从条件概率采样 *ancestral sampling* [*Bishop 2006*]

利用生成模型求解玻尔兹曼分布？

$$\mathbf{S} = \{+1, -1\}^n \qquad \uparrow\uparrow\uparrow\downarrow\uparrow\downarrow\downarrow\downarrow\downarrow\downarrow\uparrow\uparrow$$

$$P(\mathbf{S}) = \frac{1}{Z} \mathrm{e}^{-\beta E(\mathbf{S})} \qquad Z = \sum_{\mathbf{S}} \mathrm{e}^{-\beta E(\mathbf{S})}$$

- 估计自由能
- 计算统计量/序参量
- 无偏采样



变分法

$$p(\mathbf{s}) = \frac{1}{Z} e^{-\beta E(\mathbf{s})}$$

玻尔兹曼分布

$$-\beta F = \ln Z = \ln \sum_{\mathbf{s}} e^{-\beta E(\mathbf{s})}$$

自由能

Introduce a **variational distribution** **q(s)**

$$\begin{aligned} F_q &= \langle E \rangle_q - \frac{1}{\beta} S_q \\ &= F + \frac{1}{\beta} D_{\mathrm{KL}}(q \| p) \end{aligned}$$

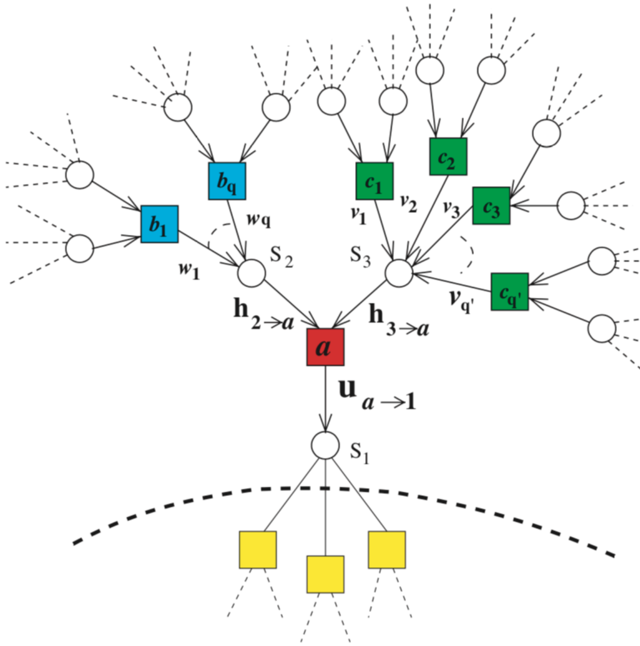
变分自由能

$$q(\mathbf{s}) = \prod_i q_i(s_i)$$

变分平均场

$$q(\mathbf{s}) = \frac{\prod_{(ij)} q_{ij}(s_i, s_j)}{\prod_i q_i(s_i)^{d_i - 1}}$$

Bethe 近似
Belief Propagation



Limitations: q(s) is not expressive

The variational Free Energy

$$P(\mathbf{s}|\mathbf{x}) = \frac{e^{\ln P(\mathbf{x}|\mathbf{s}) P_0(\mathbf{s})}}{e^{\ln P(\mathbf{x})}}$$

$$\ln Z = \ln P(\mathbf{x}) = \ln \sum_{\mathbf{s}} e^{\ln P(\mathbf{x}|\mathbf{s}) P_0(\mathbf{s})}$$

$$\ln P(\mathbf{x}) \geq \sum_{\mathbf{s}} Q(\mathbf{s}|\mathbf{x}) \ln[P(x|s)P_0(s)] - \sum_{\mathbf{s}} Q(\mathbf{s}|\mathbf{x}) \ln Q(\mathbf{s}|\mathbf{x})]$$

Variational Free Energy: Energy - Entropy

$$\ln P(\mathbf{x}) \geq \sum_{\mathbf{s}} Q(\mathbf{s}|\mathbf{x}) \ln[P(x|s)] - \mathrm{KL}\left[\ln Q(\mathbf{s}|\mathbf{x}) || P_0(\mathbf{s})\right]$$

Variational autoencoder: reconstruction error - KL regularization

Mean-field methods: Variational Mean-field

$$q(\mathbf{s}) = \prod_i q_i(s_i)$$

n parameters !

$$F_q = \sum_{\mathbf{s}} \left[q(\mathbf{s}) E(\mathbf{s}) + \frac{1}{\beta} \sum_i \ln q_i(s_i) \right]$$

$$\nabla_{\{q_i\}} F_q = 0 \Rightarrow \text{Naïve Mean-Field equations}$$

In case of the Ising model with $E(\mathbf{s}) = -\sum_{(ij)} J_{ij} s_i s_j$

$$m_i = 2q_i(s_i = 1) - 1$$

$$F_q = \sum_{(ij)} J_{ij} m_i m_j + \frac{1}{\beta} \sum_i \left(\log \frac{1+m_i}{2} + \log \frac{1-m_i}{2} \right)$$

$$m_i = \tanh(\beta \sum_{j \neq i} J_{ij} m_j)$$

Mean-field methods: Variational Mean-field

$$q(\mathbf{s}) = \prod_i q_i(s_i)$$

n parameters !

$$F_q = \sum_{\mathbf{s}} \left[q(\mathbf{s})E(\mathbf{s}) + \frac{1}{\beta} \sum_i \ln q_i(s_i) \right]$$

$$\nabla_{\{q_i\}} F_q = 0 \Rightarrow \text{Naïve Mean-Field equations}$$

Limitations: q(s) is not expressive

In case of the Ising model with $E(\mathbf{s}) = -\sum_{(ij)} J_{ij} s_i s_j$

$$m_i = 2q_i(s_i = 1) - 1$$

$$F_q = \sum_{(ij)} J_{ij} m_i m_j + \frac{1}{\beta} \sum_i \left(\log \frac{1+m_i}{2} + \log \frac{1-m_i}{2} \right)$$

$$m_i = \tanh(\beta \sum_{j \neq i} J_{ij} m_j)$$

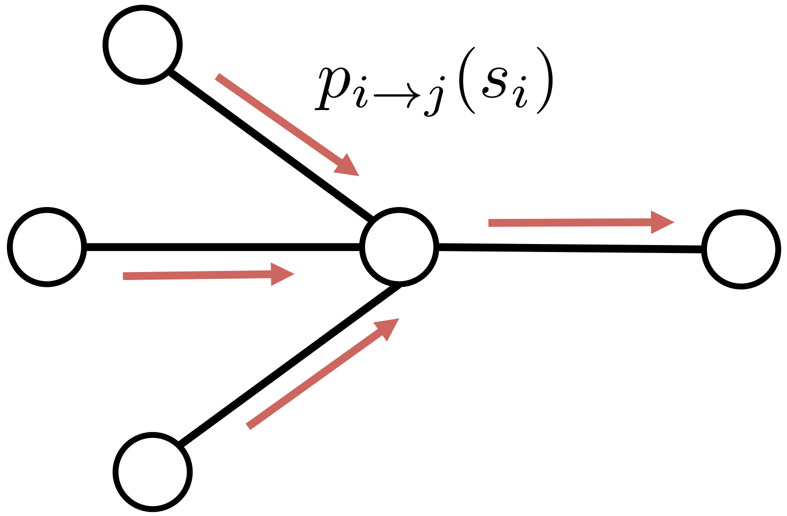
Mean-field methods: Bethe approximation

$$q(\mathbf{s}) = \frac{\prod_{(ij)} q_{ij}(s_i, s_j)}{\prod_i q_i(s_i)^{d_i - 1}}$$

- Exact on a tree
- A good approximation on sparse graphs or dense + weak systems
- However in general q(s) is not normalized on loopy graphs

$$F_q = \sum_{\mathbf{s}} q(\mathbf{s}) E(\mathbf{s}) + \frac{1}{\beta} \sum_{(ij)} \sum_{s_i, s_j} \ln q_{ij}(s_i, s_j) - \frac{1}{\beta} \sum_i (d_i - 1) \sum_{s_i} \ln q_i(s_i)$$

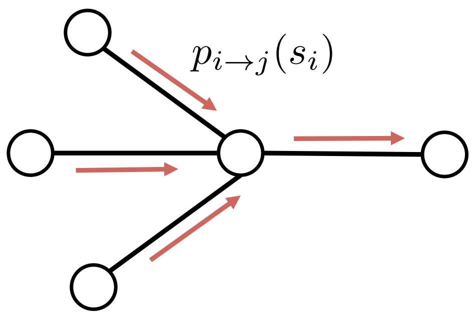
$$\nabla_{\{q_i, q_{ij}\}} F_q = 0 \Rightarrow \text{belief propagation}$$



Pros and Cons of Mean-field

Pros:

- Analytical computation of Free Energy
- Fast Message Passing
- Analysable



Cons:

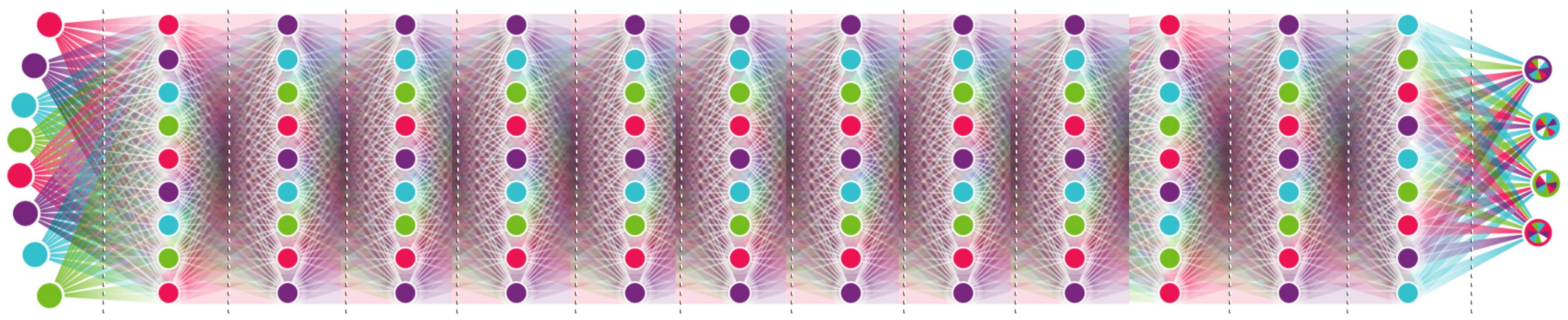
- Requires certain conditions to hold
- Low expressive power

$$q(\mathbf{s}) = \prod_i q_i(s_i)$$

$$q(\mathbf{s}) = \frac{\prod_{(ij)} q_{ij}(s_i, s_j)}{\prod_i q_i(s_i)^{d_i - 1}}$$

Variational methods with neural networks

***G*ood representation power in theory**



Challenge:

1. **Representing normalized joint distribution**
2. **Computing variational free energy**

Vatiational autoregressive neural networks.

$$q(s) = \prod_i q(s_i \,|\, s_{j<i}) \quad + \quad \textbf{Reinforcement learning}$$

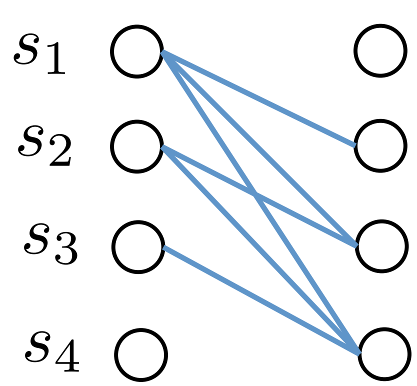
D Wu, L Wang, PZ, *Phys. Rev. Lett. 122*, 080602 (2019)

Auto-regressive distribution

$$q(\mathbf{s}) = \prod_i q(s_i|\mathbf{s}_{j<i})$$

$$\begin{aligned} q(s_1, s_2, s_3, s_4) &= q(s_4|s_3, s_2, s_1)q(s_3, s_2, s_1) \\ &= q(s_4|s_3, s_2, s_1)q(s_3|s_2, s_1)q(s_2, s_1) \\ &= q(s_4|s_3, s_2, s_1)q(s_3|s_2, s_1)q(s_2|s_1)q(s_1) \end{aligned}$$

Fully Visible Belief Network
[Frey 1998]



$$\hat{s}_1 = \mathrm{sigmoid}(0) = q(s_1 = 1)$$

$$\hat{s}_2 = \mathrm{sigmoid}(w_{12}s_1) = q(s_2 = 1|s_1)$$

$$\hat{s}_3 = \mathrm{sigmoid}(w_{13}s_1 + w_{23}s_2) = q(s_3 = 1|s_2, s_1)$$

$$\hat{s}_4 = \mathrm{sigmoid}(w_{14}s_1 + w_{24}s_2 + w_{34}s_4) = q(s_4 = 1|s_3, s_2, s_1)$$

$$q(s1, s2, s3, s4) = \hat{s}_1\hat{s}_2\hat{s}_3\hat{s}_4$$

无偏采样：从条件概率采样 *ancestral sampling* [*Bishop 2006*]

变分自由能优化：强化学习

- Minimizing the variational free energy = minimizing KL divergence

$$\hat{\theta} = \arg\min_{\theta} \mathrm{D_{KL}}(q_\theta | p) = \arg\min F_q$$

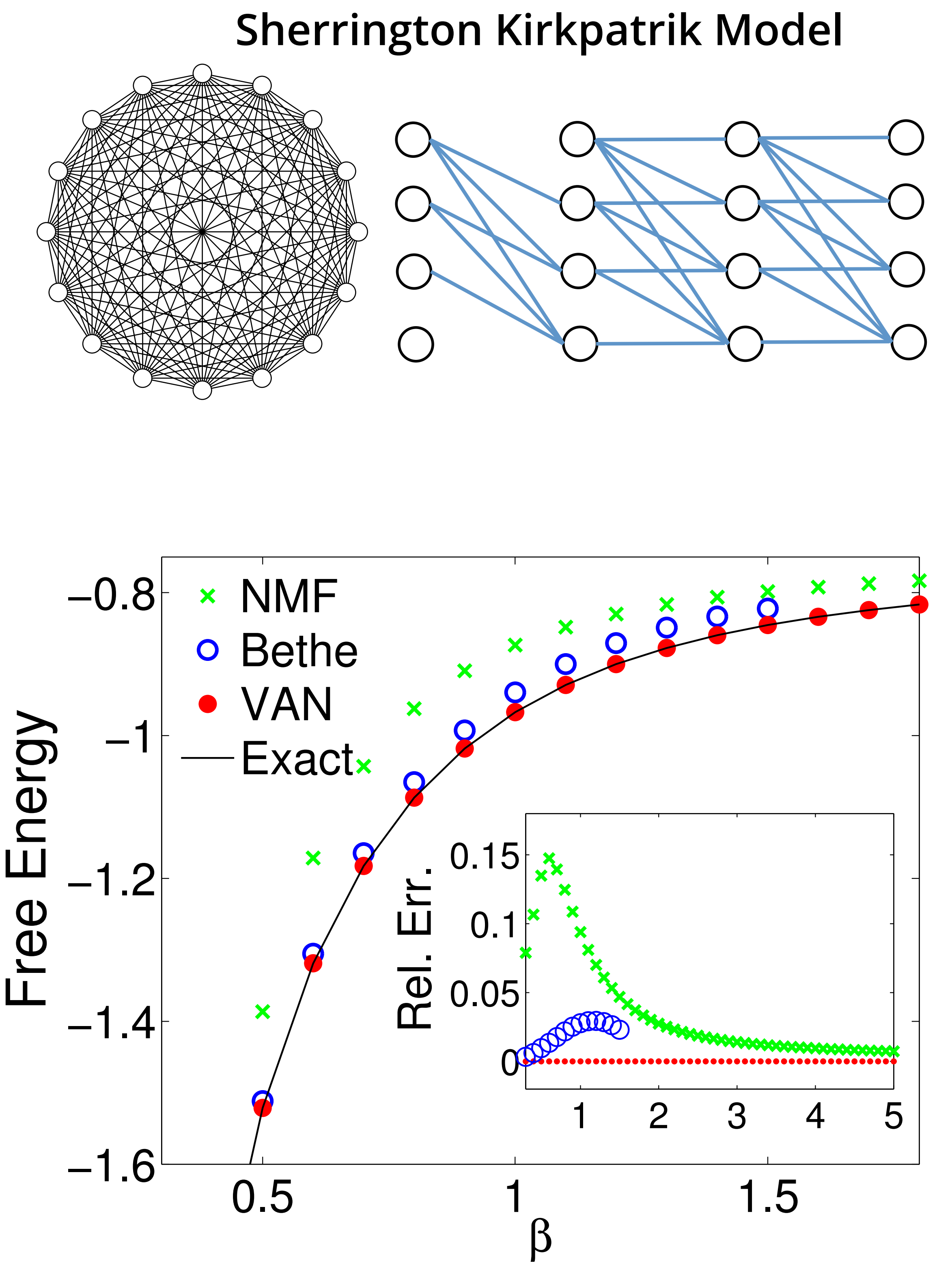
$$\beta F_q = \sum_{\mathbf{s}} q_\theta(\mathbf{s}) \left[\beta E(\mathbf{s}) + \ln q_\theta(\mathbf{s})\right]$$

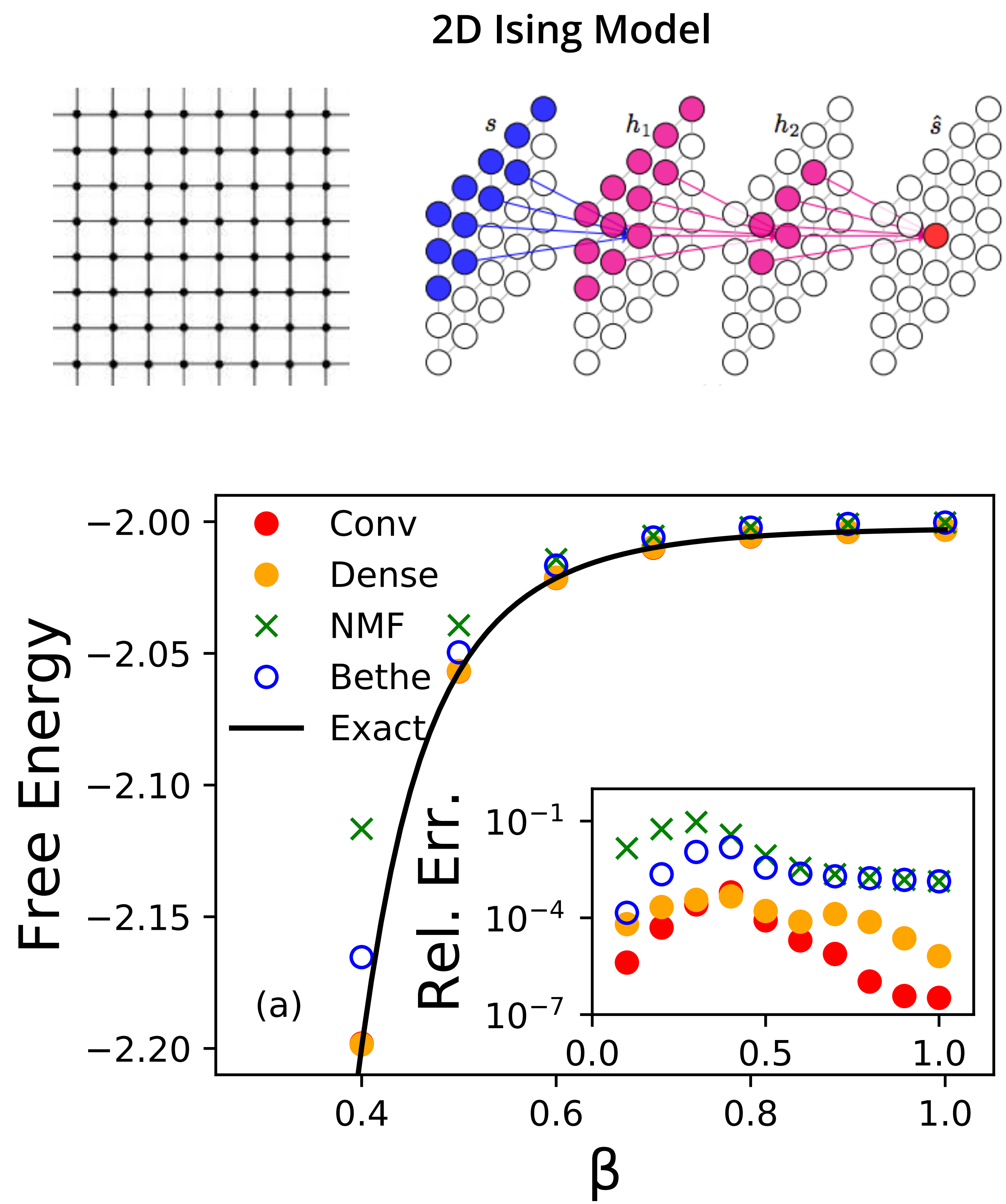
- Policy gradients:

$$\begin{aligned}
\beta \nabla_\theta F_q &= \nabla_\theta \sum_{\mathbf{s}} \left[q_\theta(\mathbf{s}) \cdot (\beta E(\mathbf{s}) + \ln q_\theta(\mathbf{s}))\right] \\
&= \sum_{\mathbf{s}} \left[\nabla_\theta q_\theta(\mathbf{s}) \cdot (\beta E(\mathbf{s}) + \ln q_\theta(\mathbf{s})) + q_\theta(\mathbf{s}) \nabla_\theta \ln q_\theta(\mathbf{s})\right] \\
&= \sum_{\mathbf{s}} \left[q_\theta(\mathbf{s}) \nabla_\theta \log q_\theta(\mathbf{s}) \cdot (\beta E(\mathbf{s}) + \ln q_\theta(\mathbf{s})) + \nabla_\theta q_\theta(\mathbf{s})\right] \\
&= \mathbb{E}_{\mathbf{s} \sim q_\theta(\mathbf{s})} \left[\nabla_\theta \ln q_\theta(\mathbf{s}) \cdot \underbrace{(\beta E(\mathbf{s}) + \ln q_\theta(\mathbf{s}))}_{R(\mathbf{s})}\right]
\end{aligned}$$

Known as the ***REINFORCE*** algorithm **[Williams 1992]**

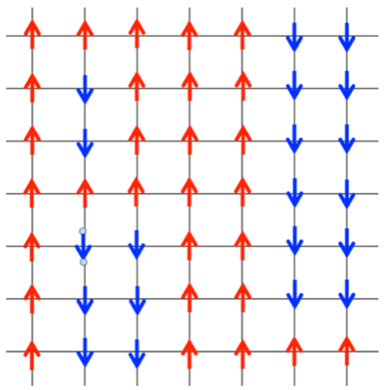
计算自旋玻璃自由能



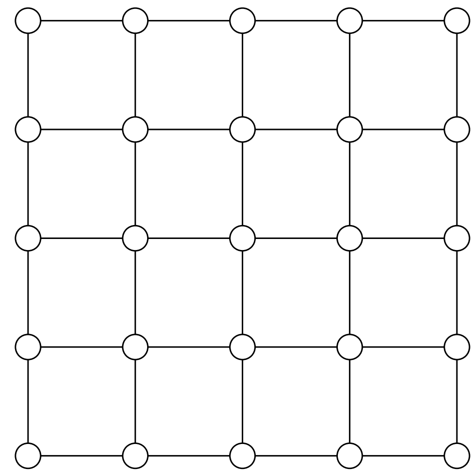


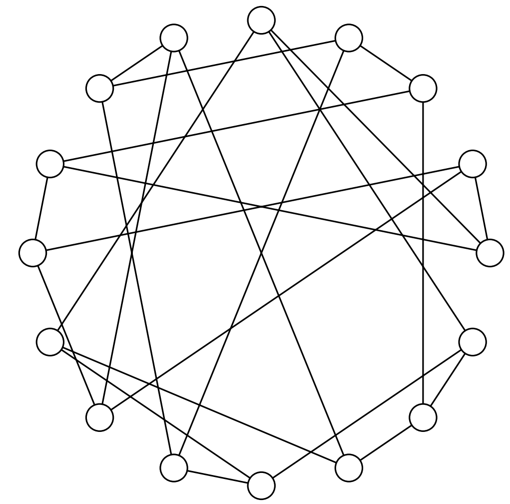
D. Wu, L. Wang, PZ, *Phys. Rev. Lett.* 122, 080602 (2019)

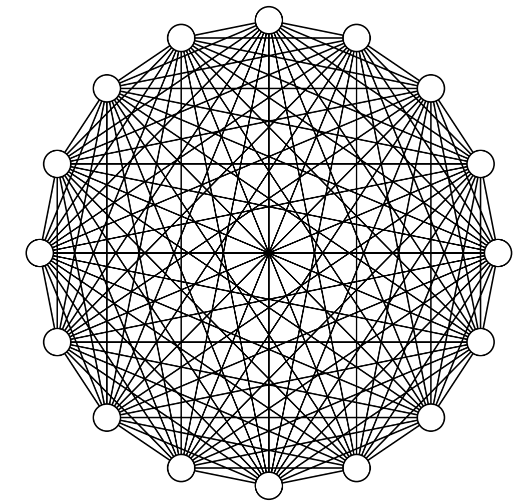
统计力学 → 量子计算机模拟

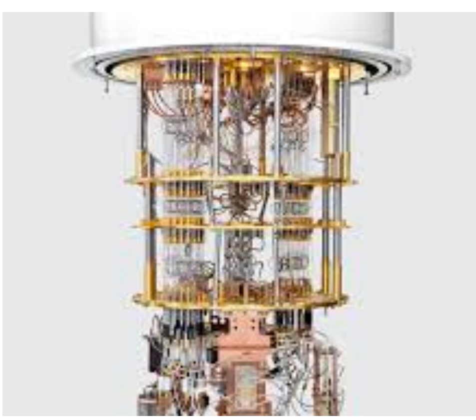


伊辛模型

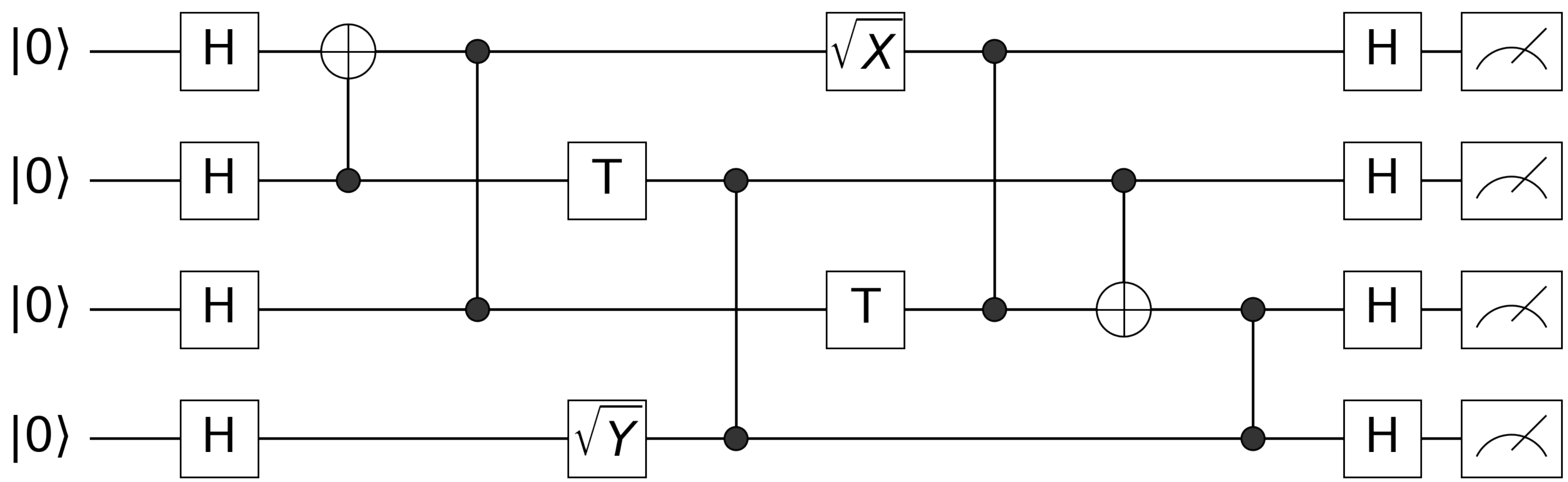






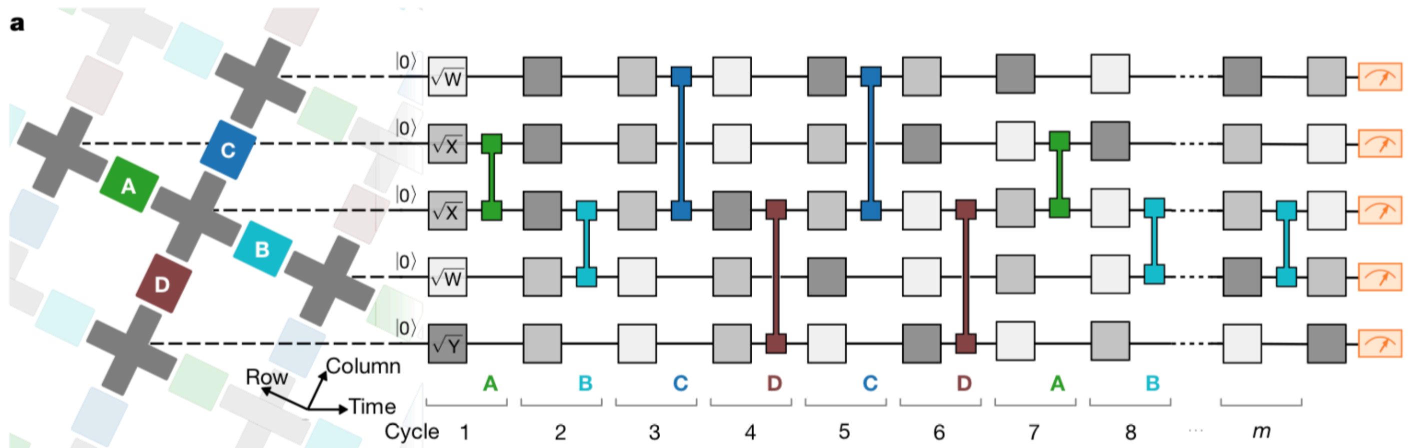


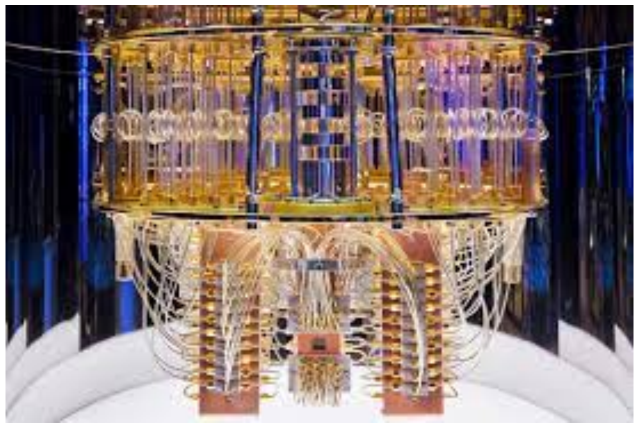
量子线路

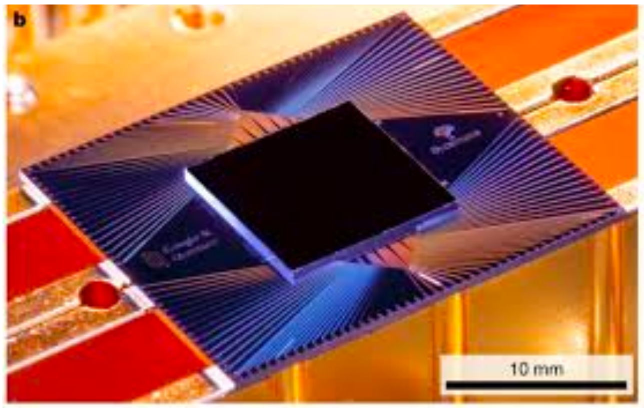


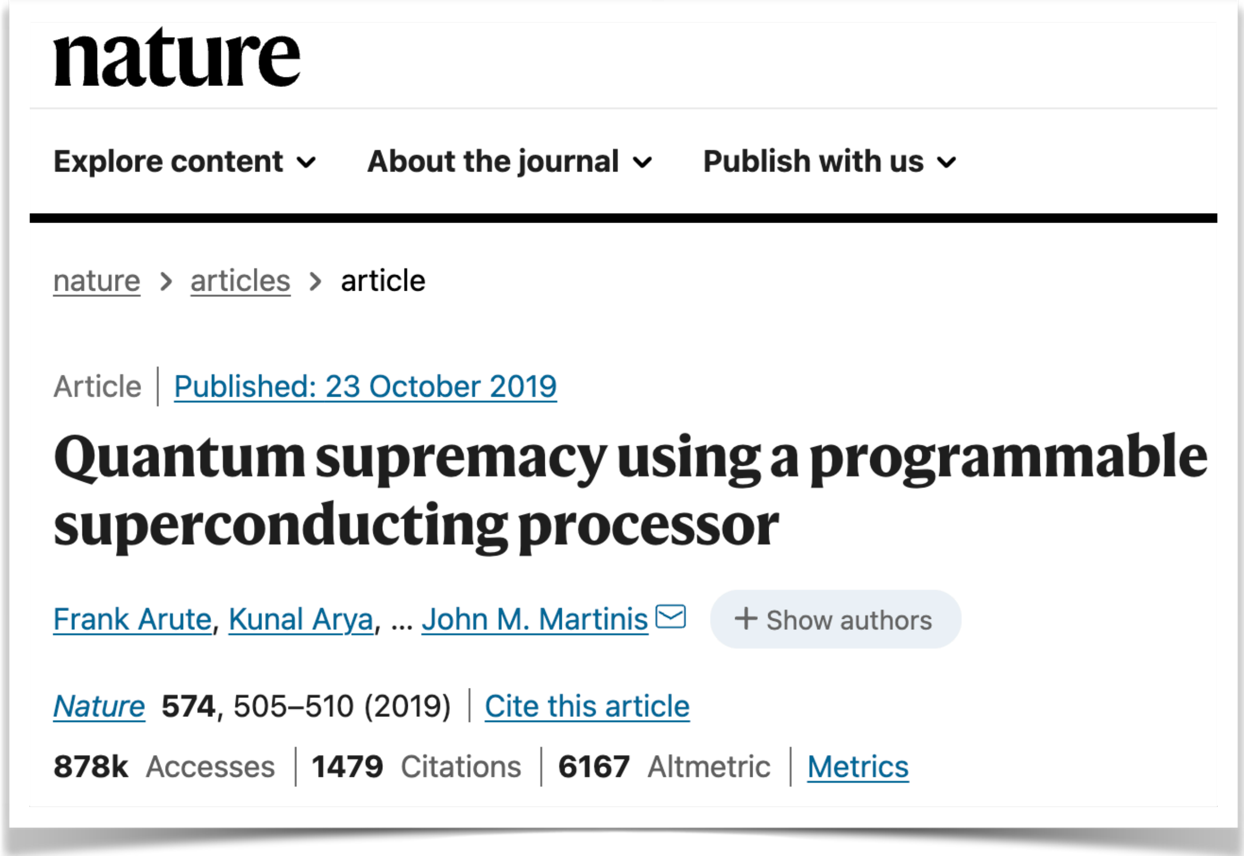
复数相互作用的伊辛模型 ⟺ 量子计算机单振幅计算

Google's Quantum Supremacy experiments









nature

Explore content ⌄ About the journal ⌄ Publish with us ⌄

nature > articles > article

Article | Published: 23 October 2019

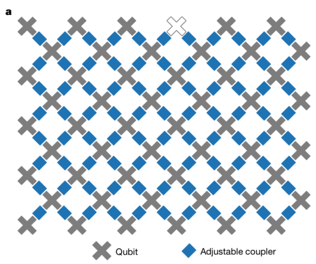
Quantum supremacy using a programmable superconducting processor

Frank Arute, Kunal Arya, ... John M. Martinis + Show authors

Nature **574**, 505–510 (2019) | Cite this article

878k Accesses | **1479** Citations | **6167** Altmetric | Metrics

- **53 qubits, 20 cycles**



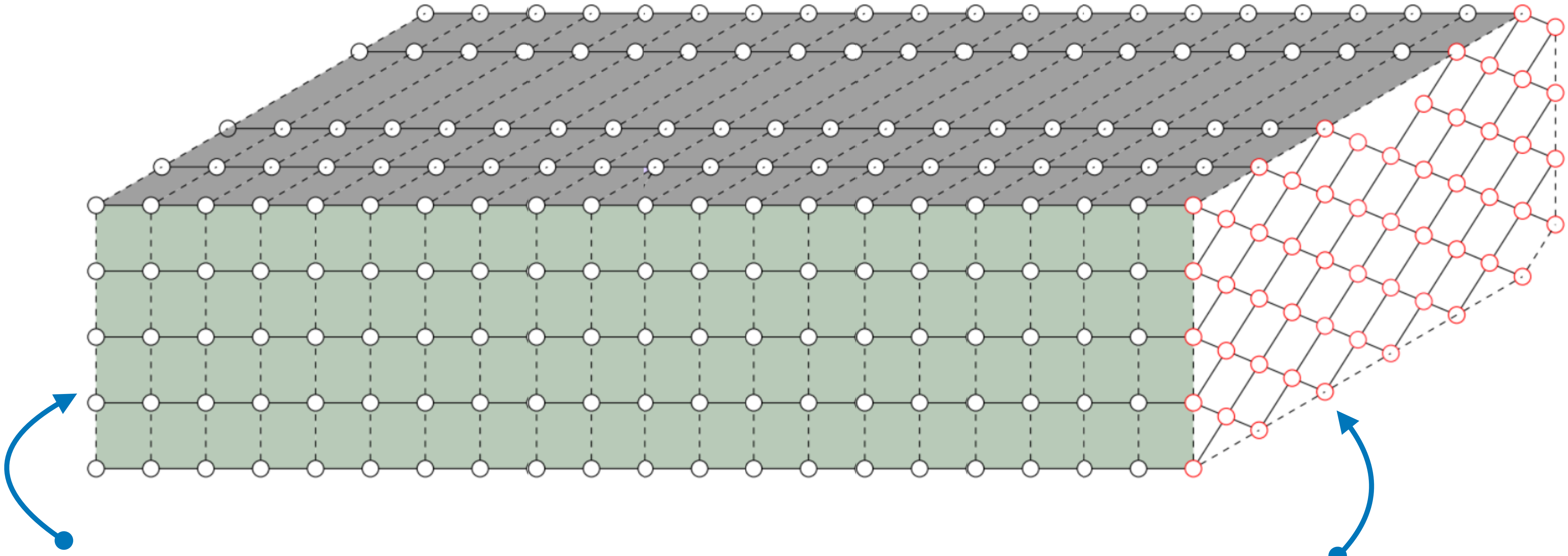
$$\mathrm{fSim}(\theta,\phi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -i\sin\theta & 0 \\ 0 & -i\sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & e^{-i\phi} \end{bmatrix}$$

- **1 million samples in 200 Sec.**
- **Linear Cross Entropy Fidelity (XEB) ≈ 0.002**
- **Classic algorithm requires 10,000 years on Summit**

$$F_{\mathrm{XEB}} = 2^n \sum_{s\in\{1,0\}^n} q(\mathbf{s}) p_U(\mathbf{s}) - 1$$
$$= 2^n \langle p_U(\mathbf{s})\rangle_q - 1$$
$$\approx \frac{2^n}{m} \sum_{s\sim q} p_U(\mathbf{s}) - 1$$

Arute et al, Nature 2019

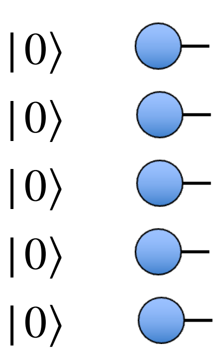
Solving the sampling problem of Sycamore



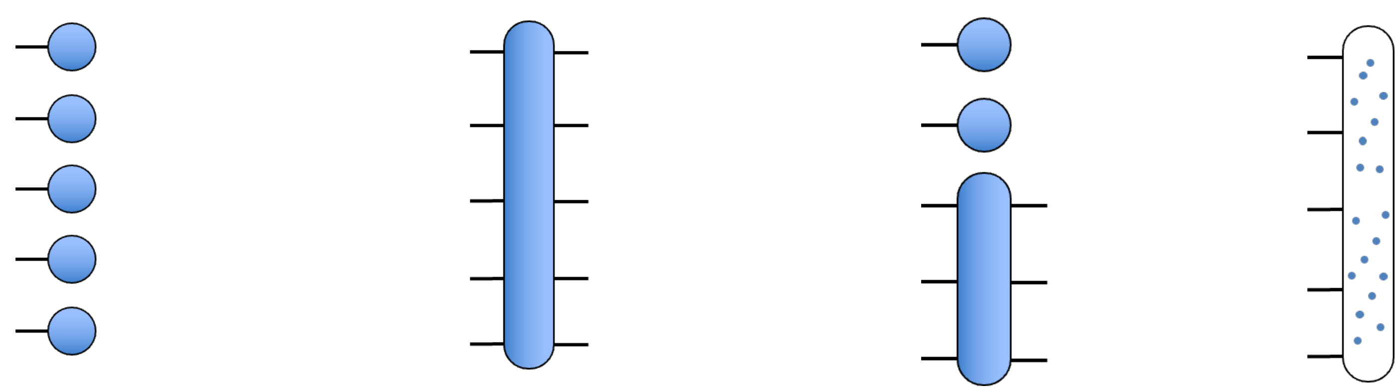
Left boundary condition:

Right boundary condition:

Product state



Single-amplitude **Full-amplitude** **Big-batch** **Sparse state**

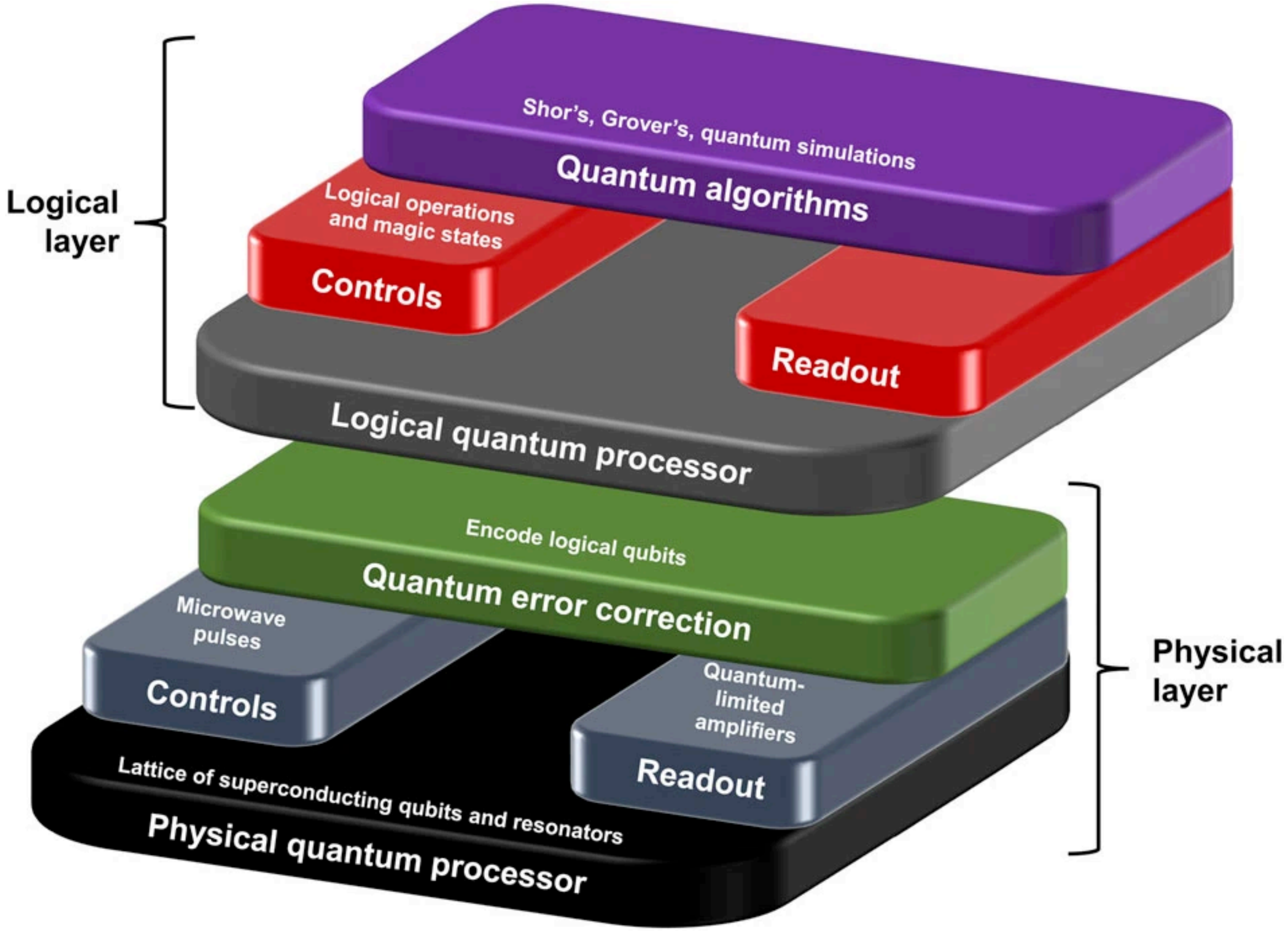


Can solve the problem in dozens of seconds

F. Pan and PZ, Phys. Rev. Lett. 128, 030501 (2022)

F. Pan, K. Chen, and PZ, Phys. Rev. Lett. 129, 090502 (2022)

Quantum Error Correction



Repetition code

Code words:

- $0_L = 000, \; 1_L = 111$

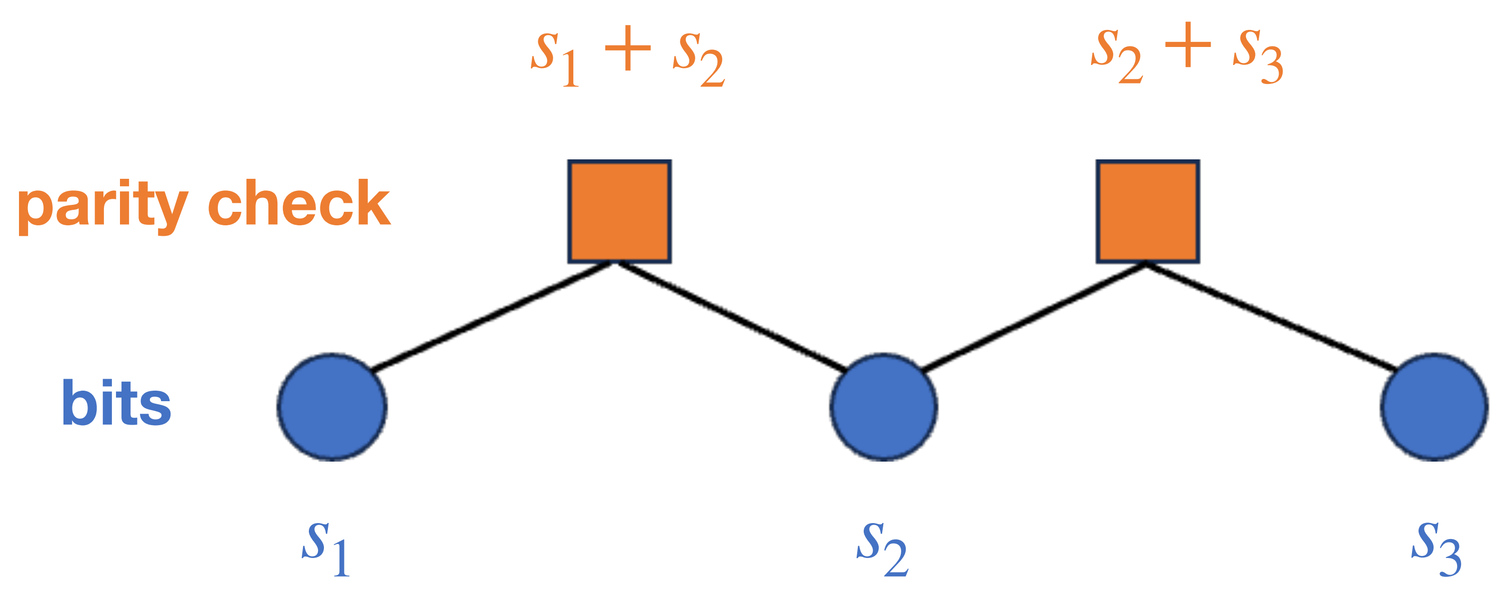
Error model:

- e.g. **bit-flip error,** prob. p

Decoding:

- $010 \longrightarrow 0_L$
- $011 \longrightarrow 1_L$

Example: classical repetition code



$$H = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

$u = (0,0,0)$ **and** $(1,1,1)$ $\qquad Hu^T = 0$

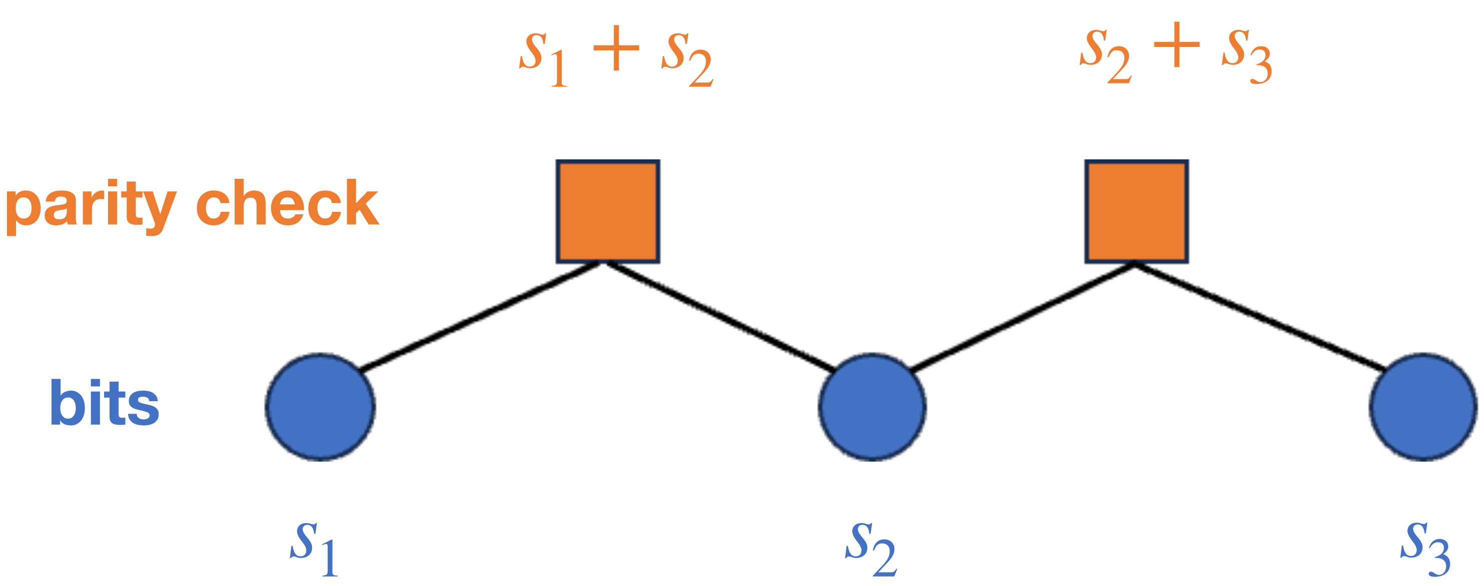
$G = (1 \quad 1 \quad 1)$ $\qquad HG^T = 0$

Parity Check matrix $\{0,1\}^{(n-k)\times n}$

Generator matrix $\{0,1\}^{k\times n}$

k**-dimensional linear subspace, spanned by rows of** G

Example: classical repetition code



$$H = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

$u = (0,0,0)$ **and** $(1,1,1)$ $\qquad Hu^T = 0$

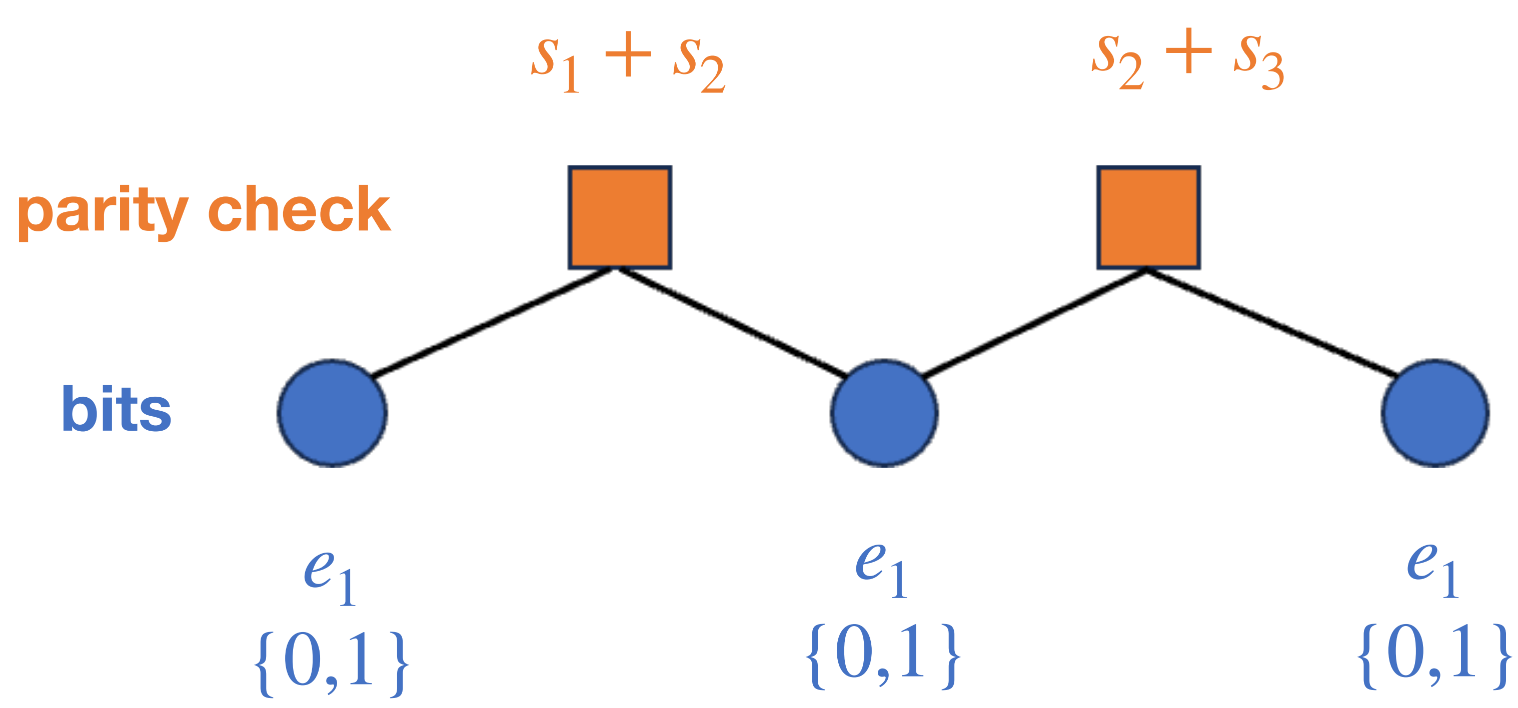
Bit-flip error e **with probability** p

$$u \longrightarrow u + e$$

$$H(u+e)^T = Hu^T + He^T = He^T = s$$ ***Syndrome***

Parity checks on Flip Errors

Repetition code



$$H = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

Decoding: find configuration $\{e_1, \ldots, e_n\} \in \{1,0\}^n$

— consistent with the syndrome

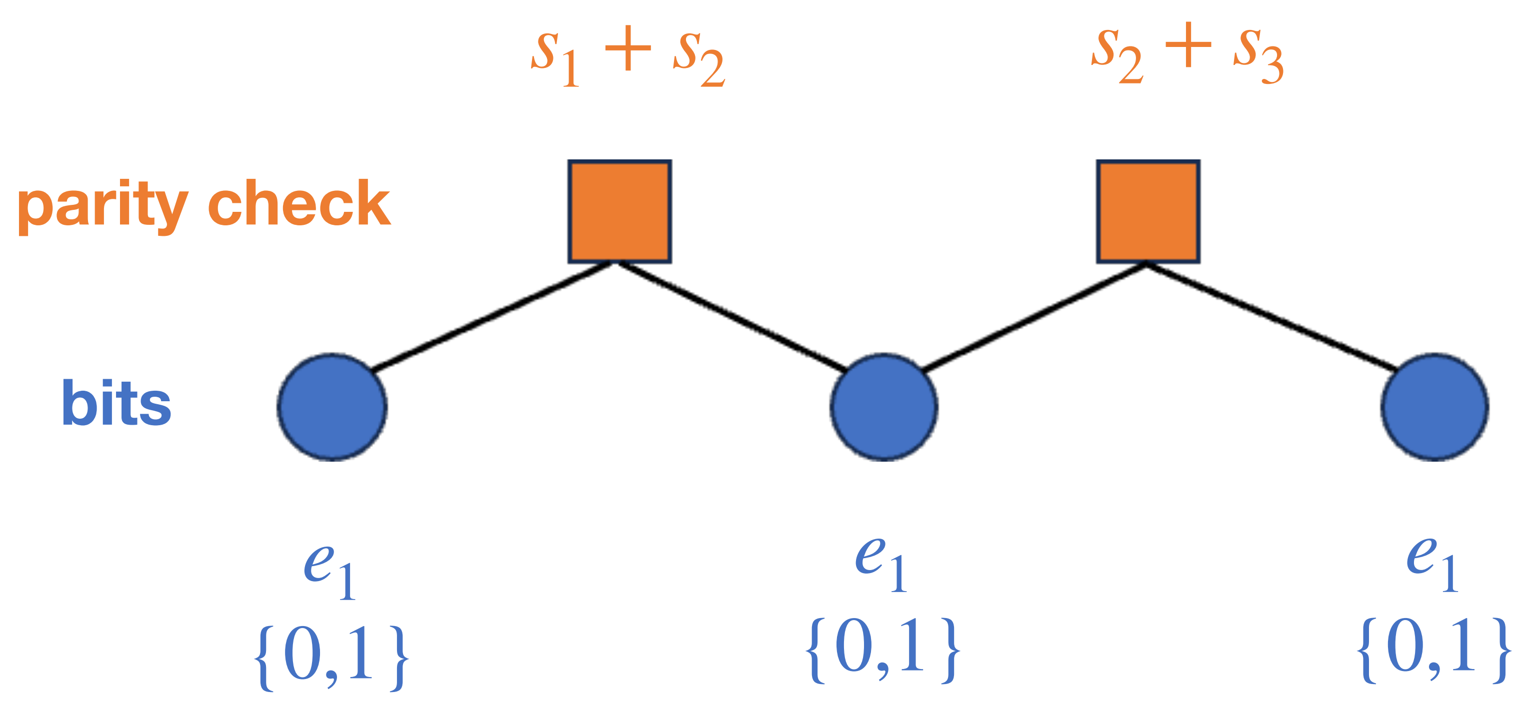
— with the maximum probability

$$P(\{e_1, \ldots, e_n\}) = \frac{1}{Z}\mathbf{1}(He = s)\prod_{i=1}^{n} p^{e_i}(1-p)^{1-e_i}$$

$$= \frac{1}{Z}e^{-\beta E(e)}$$

← **Boltzmann distribution**

Example: classical repetition code

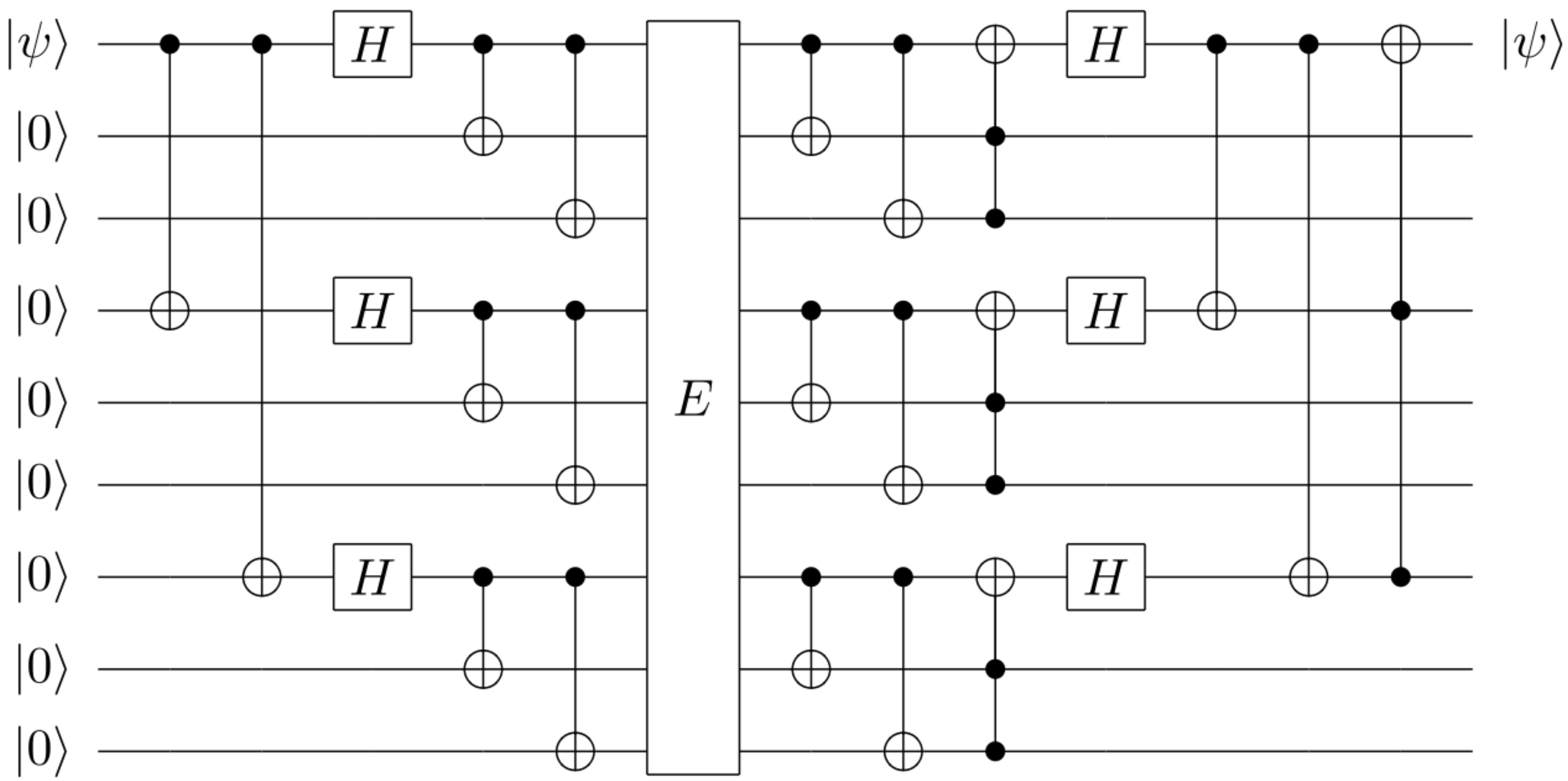


$$H = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

Syndrome $s = (1,0)$

(e_1, e_2, e_3)	$P(e_1, e_2, e_3)$	He
(0,0,0)	$(1-p)^3$	(0,0)
(0,0,1)	$(1-p)^2p$	(0,1)
(0,1,0)	$(1-p)^2p$	(1,1)
(0,1,1)	$(1-p)p^2$	(1,0)
(1,0,0)	$(1-p)^2p$	(1,0)
(1,0,1)	$(1-p)^2p$	(1,1)
(1,1,0)	$(1-p)p^2$	(0,1)
(1,1,1)	p^3	(0,0)

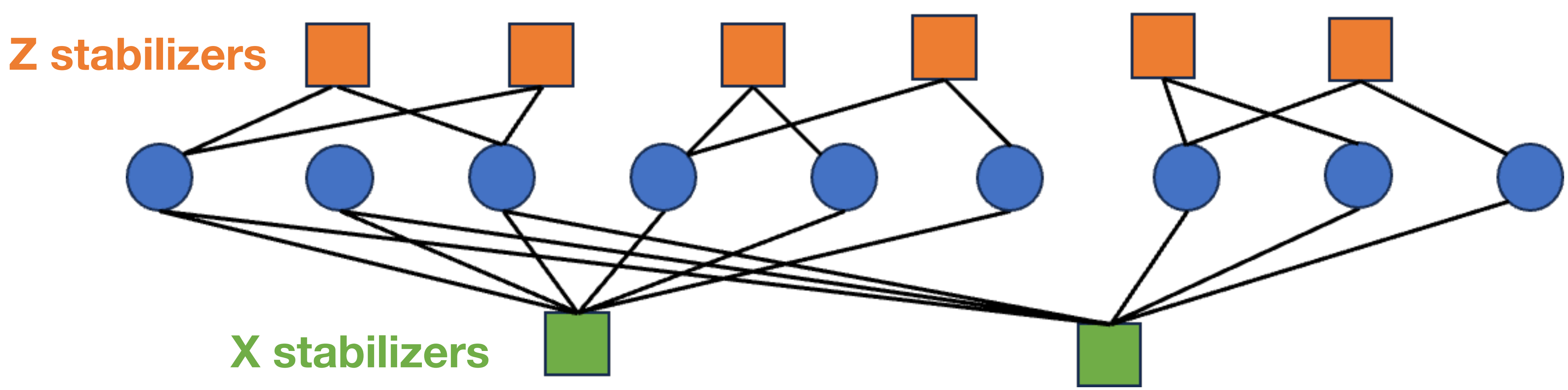
Shor's 9 qubit code



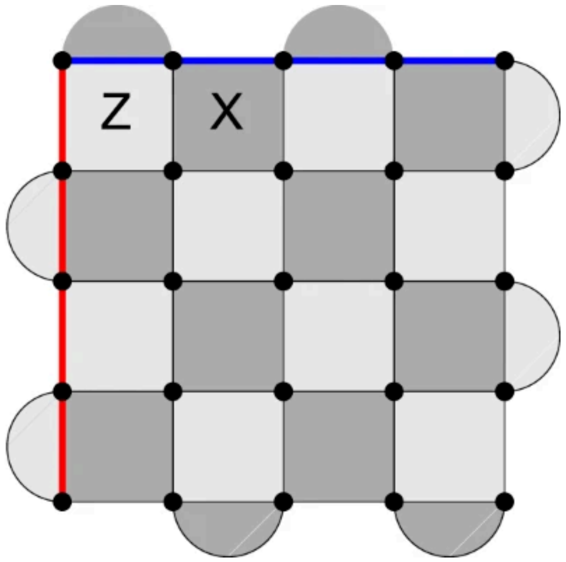
$$H = \left(\begin{array}{c|c} 111111000 & 000000000 \\ 111000111 & 000000000 \\ 000000000 & 110000000 \\ 000000000 & 101000000 \\ 000000000 & 000110000 \\ 000000000 & 000101000 \\ 000000000 & 000000110 \\ 000000000 & 000000101 \end{array}\right)$$

$$G = \left(\begin{array}{c|c} H_x & H_z \\ 111111111 & 000000000 \\ 000000000 & 111111111 \end{array}\right)$$

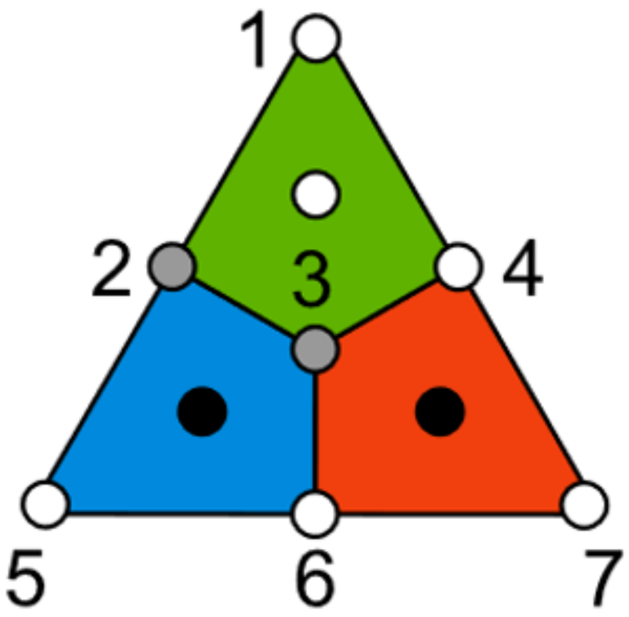
$$|0\rangle_L = (|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)$$
$$|1\rangle_L = (|000\rangle - |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)$$



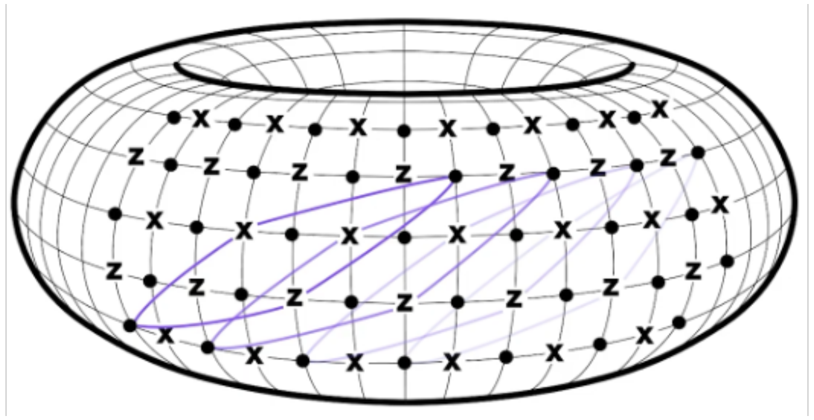
Quantum codes



Surface code



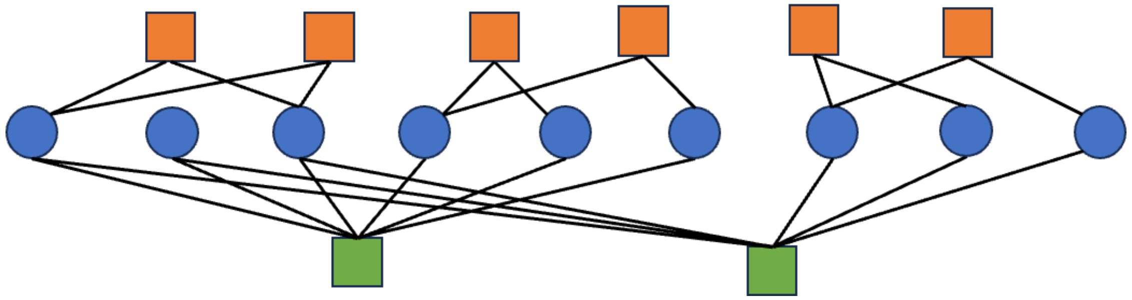
Color code



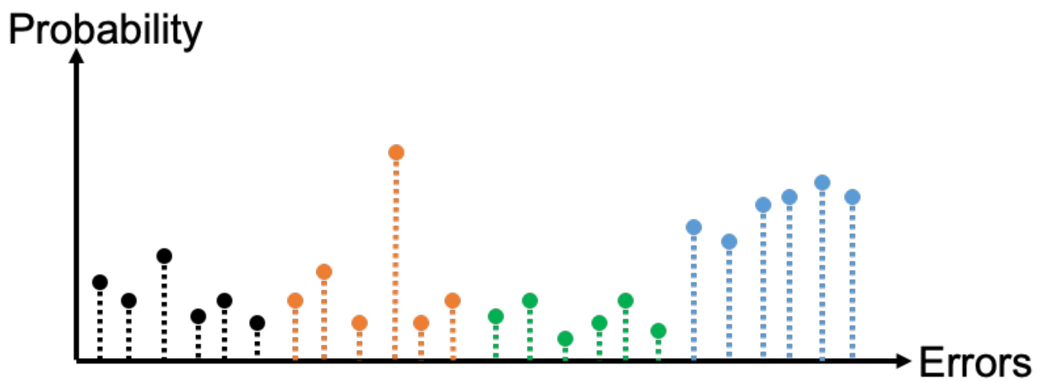
Bivariate Bicycle code

Challenges of QEC decoding

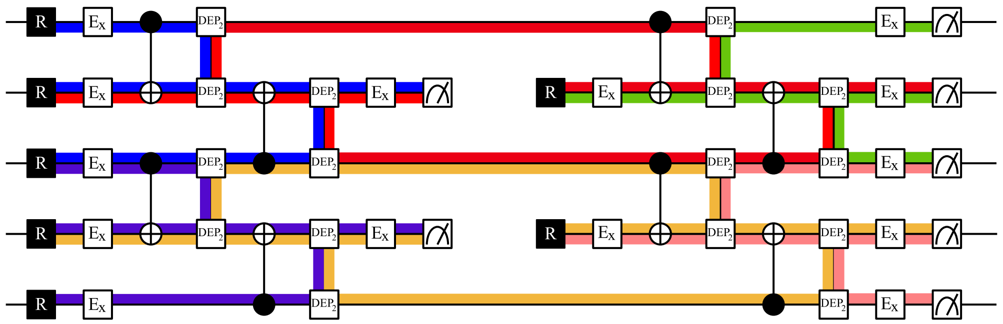
1. ***Tanner graph contains loops***
    - Commutation of stabilizers
    - BP does not work directly



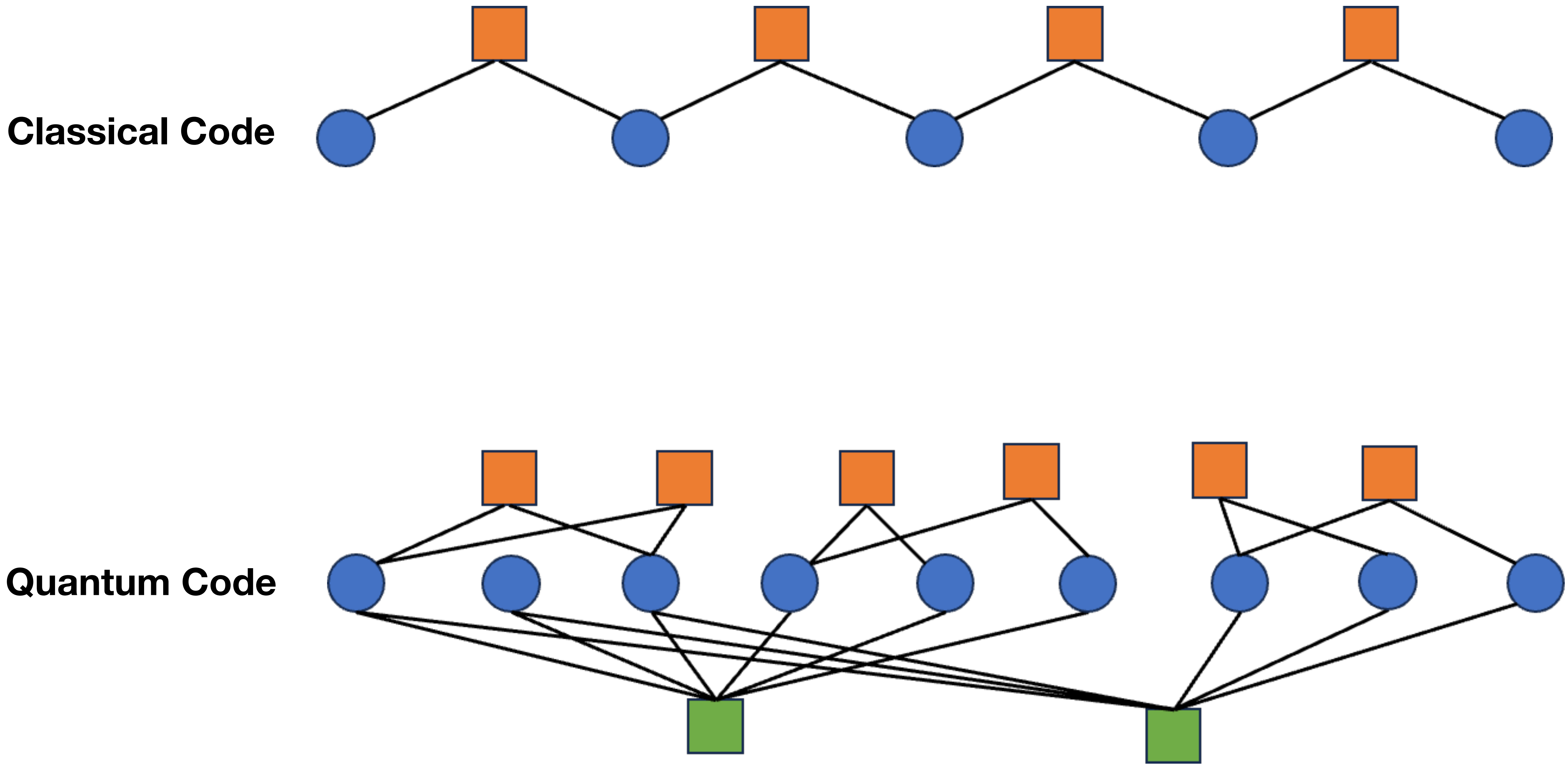
2. ***Degeneracy***
    - Many errors are equivalent



3. ***Measurement noise***
    - Repeated measurements
    - Circuit-level noise



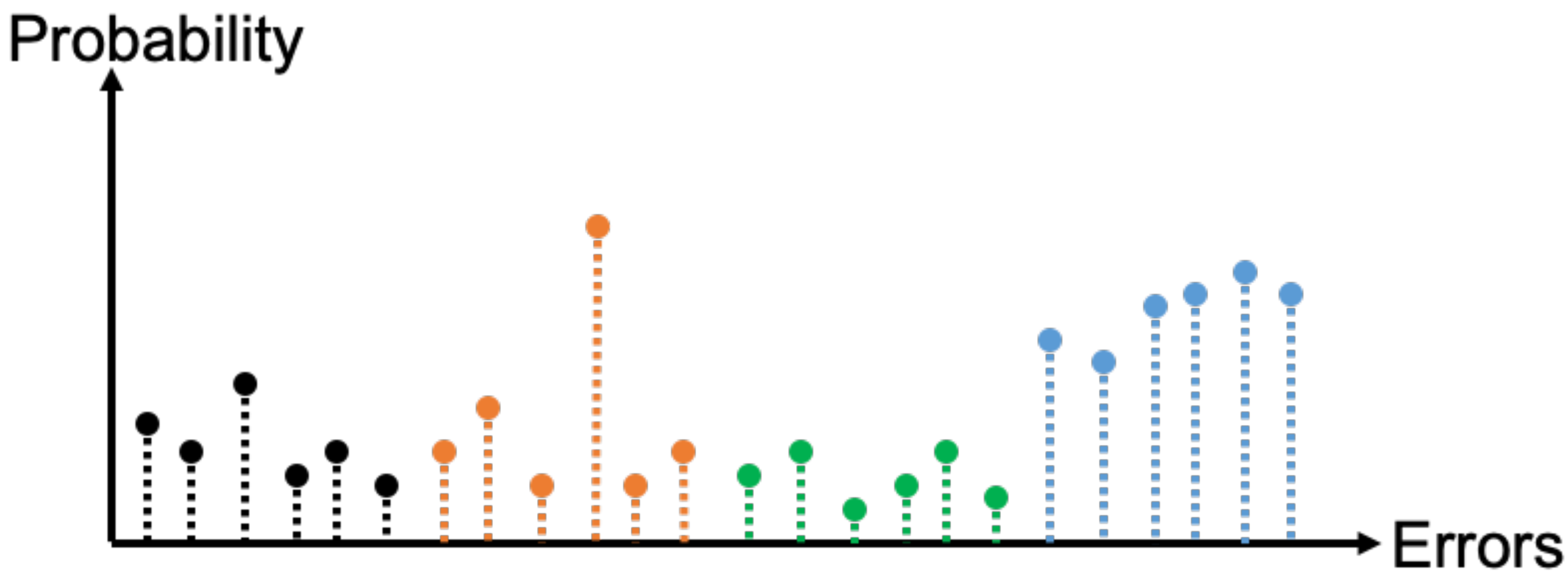
Challenges: tanner graph contains loops



Challenges: Degeneracy of errors

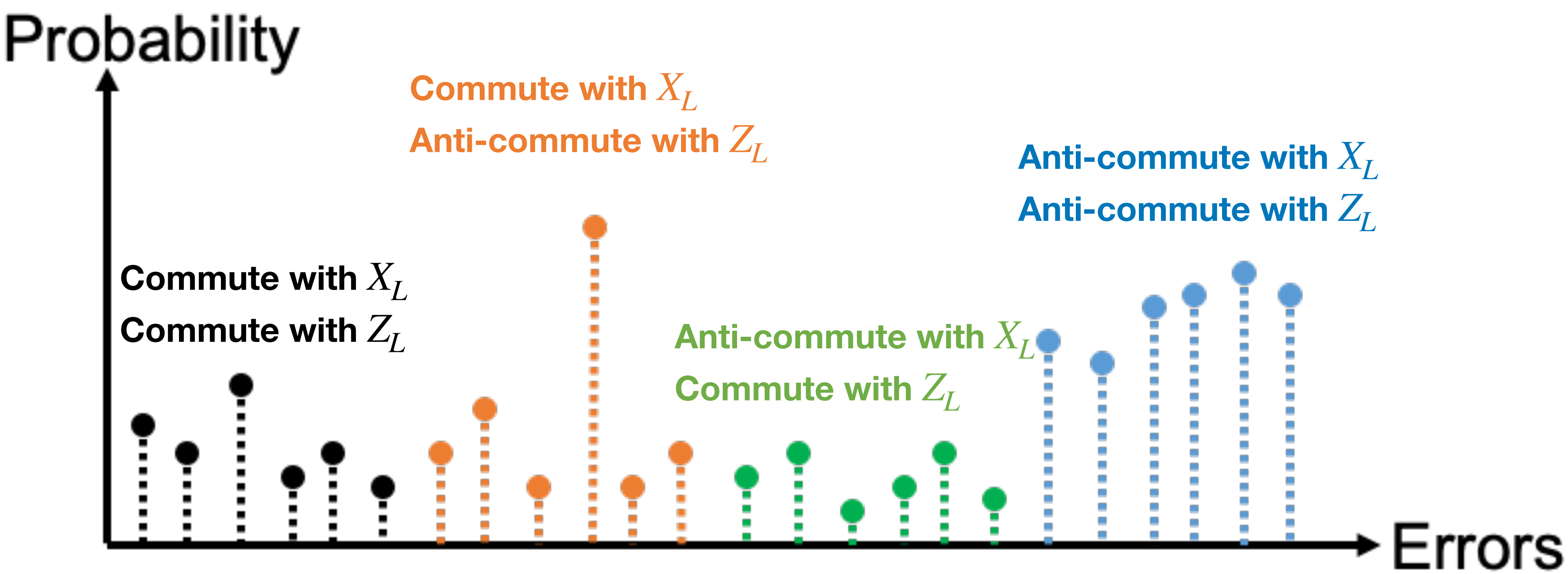
Minimum-weight decoding (e.g. MWPM) is not optimal

Degeneracy: Many errors are equivalent



Challenges: Degeneracy of errors

Group the errors into equivalent classes,
Find the *class with the maximum probability*

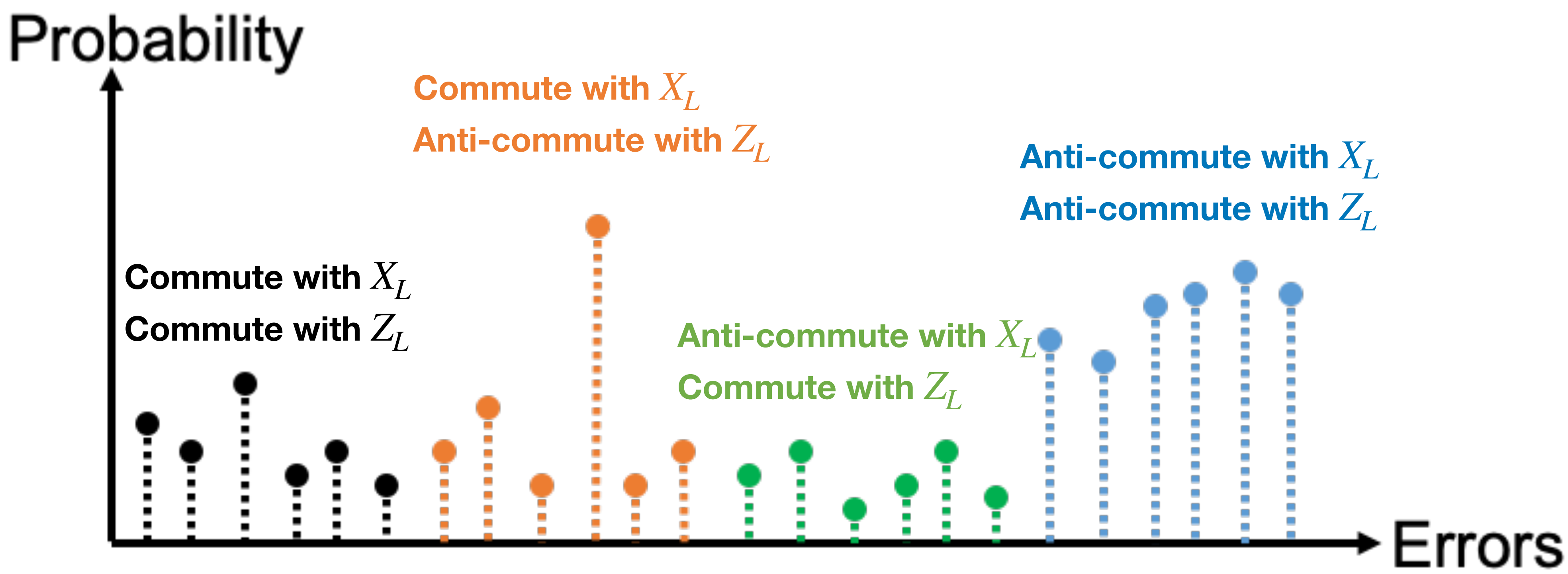


Computing the probability of a *coset* rather than an element in Pauli group

Summing over all possible 2^{n-k} stabilizer configurations

Maximum likelihood decoding (MLD)

Group the errors into equivalent classes,
Find the *class with the maximum probability*

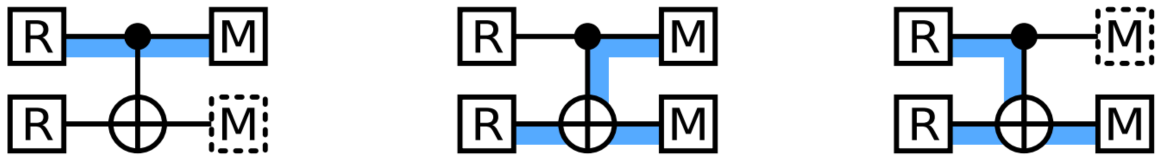


The maximum-likelihood logical equivalent class tells us how to recover.

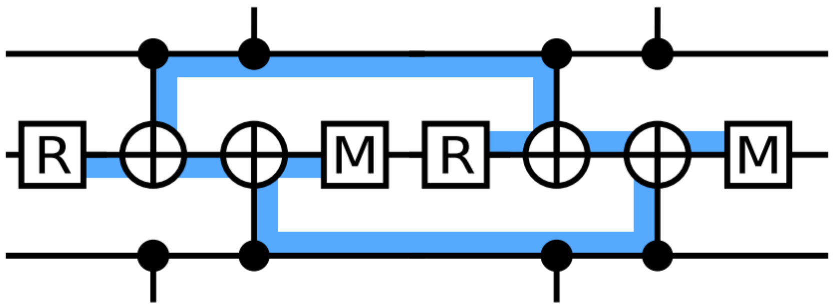
Commute with X_L, Z_L: Logical operation is I_L	no need to recover
Commute with X_L, anti-commute with Z_L	apply X_L to recover
Anti-commute with X_L, commute with Z_L	apply Z_L to recover
Anti-commute with X_L, anti-commute with Z_L	apply Y_L to recover

Noisy measurements: Circuit-level noise

Simplest detection region: from reset to measurement



Detectors and detector regions
For a CNOT gate



Bit-flip repetition code
Two consecutive measurements

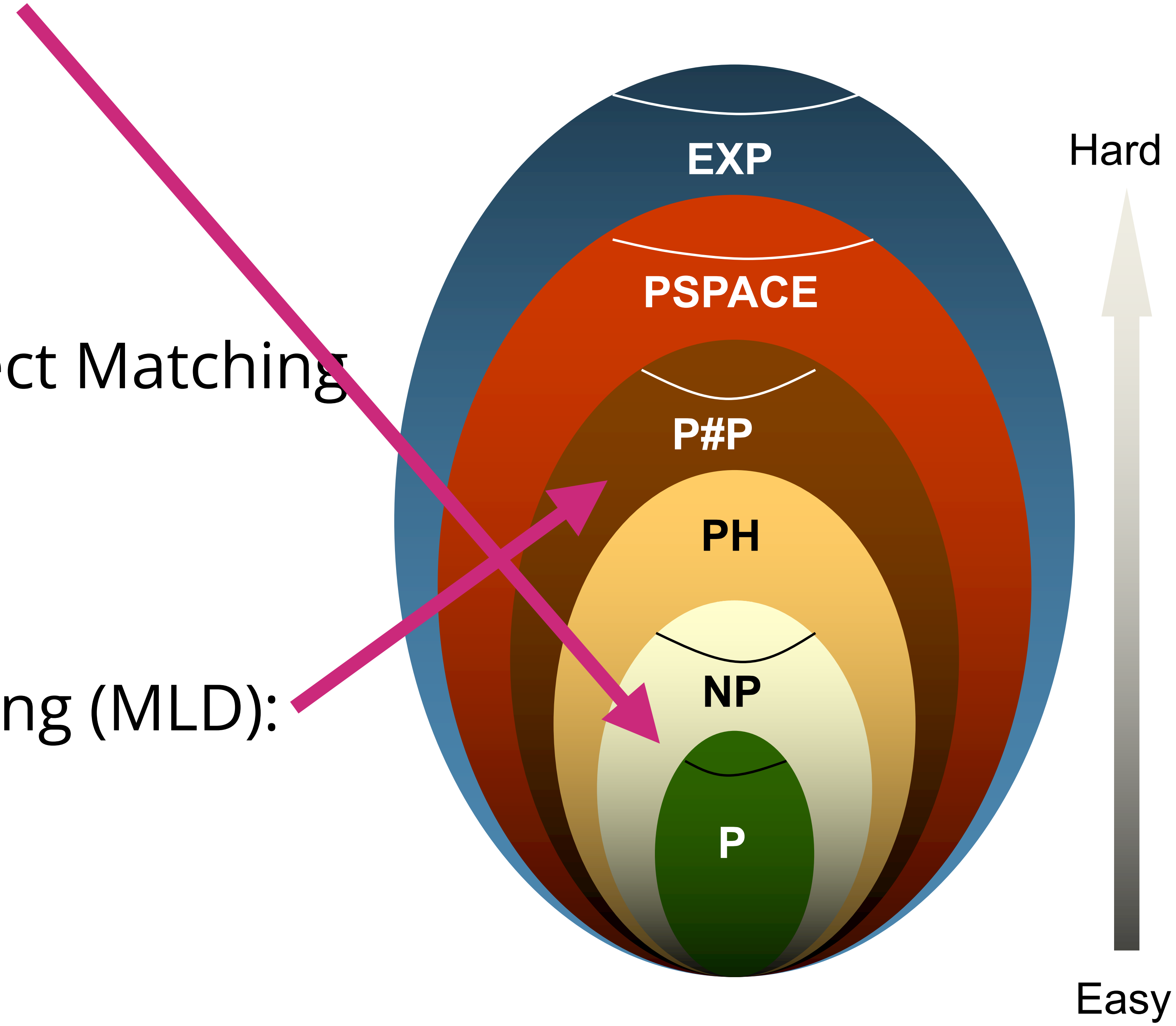
Matt McEwen, Dave Bacon, and Craig Gidney, *Quantum* 2023

Decoding

Minimum weight decoding:

- $\arg\max_{E} P(E)$

- Minimum Weight Perfect Matching (MWPM),
  Belief propagation …



Maximum likelihood decoding (MLD):

- $\arg\max_{\beta} \sum_{\alpha} P(E(\alpha, \beta, \gamma))$

- Tensor network decoding …

Solving decoding problem: $\{\mathbb{E}, \mathbb{L}, \mathbb{S}\}$ decomposition

Stabilizer operators $s \in \mathbb{S}$, Abelian, $-I \notin \mathbb{S}$

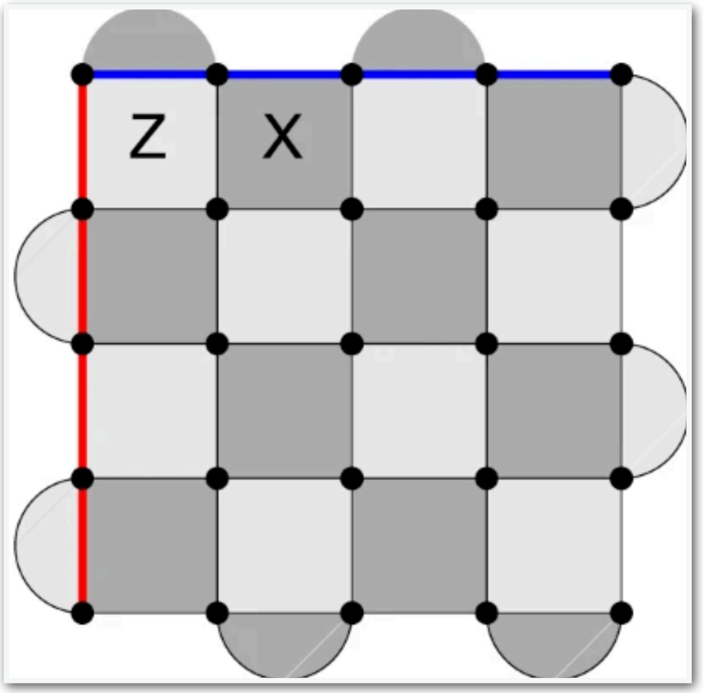
- 2^m operators, generated by $\langle g_1, g_2, \cdots, g_m \rangle$
- $g_i |\psi\rangle = |\psi\rangle \qquad g_i g_j = g_j g_i$

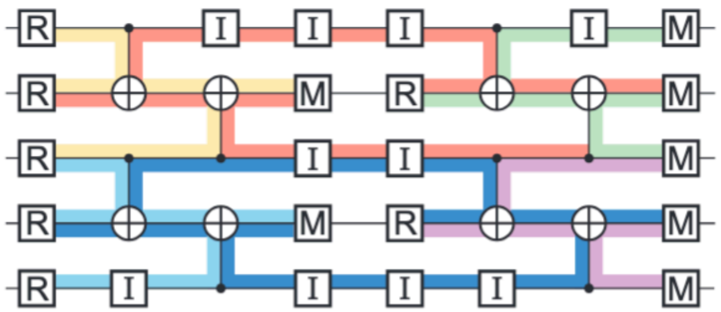
Pure error operators $e \in \mathbb{E}$, Abelian

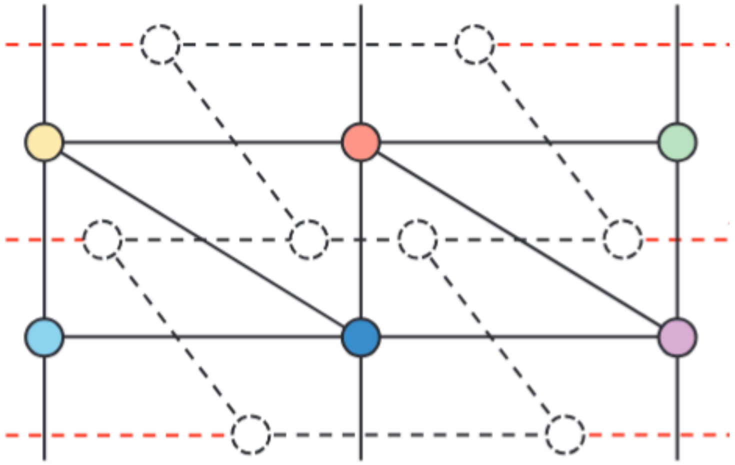
- 2^m operators, generated by $\langle e_1, e_2, \cdots, e_m \rangle$
- $g_i e_j = (-1)^{\delta_{ij}} e_j g_i \qquad e_i e_j = e_j e_i$

Logical operators $l \in \mathbb{L}$, $\mathbb{L} = \mathcal{N}(\mathbb{S})/\mathbb{S}$

- 4^k operators, generated by $\langle l_1^x, l_2^x, \cdots, l_k^x, l_1^z, l_2^z, \cdots, l_k^z \rangle$
- $l_i^{x/z} g_j = g_j l_i^{x/z}, \quad l_i^{x/z} e_j = e_j l_i^{x/z}, \quad l_i^x l_j^z = (-1)^{\delta_{ij}} l_j^z l_i^x$







Solving decoding problem: mapping to Statistical Mechanics

An error $E \in \mathbb{P}_n = \{I, X, Y, Z\}^n$

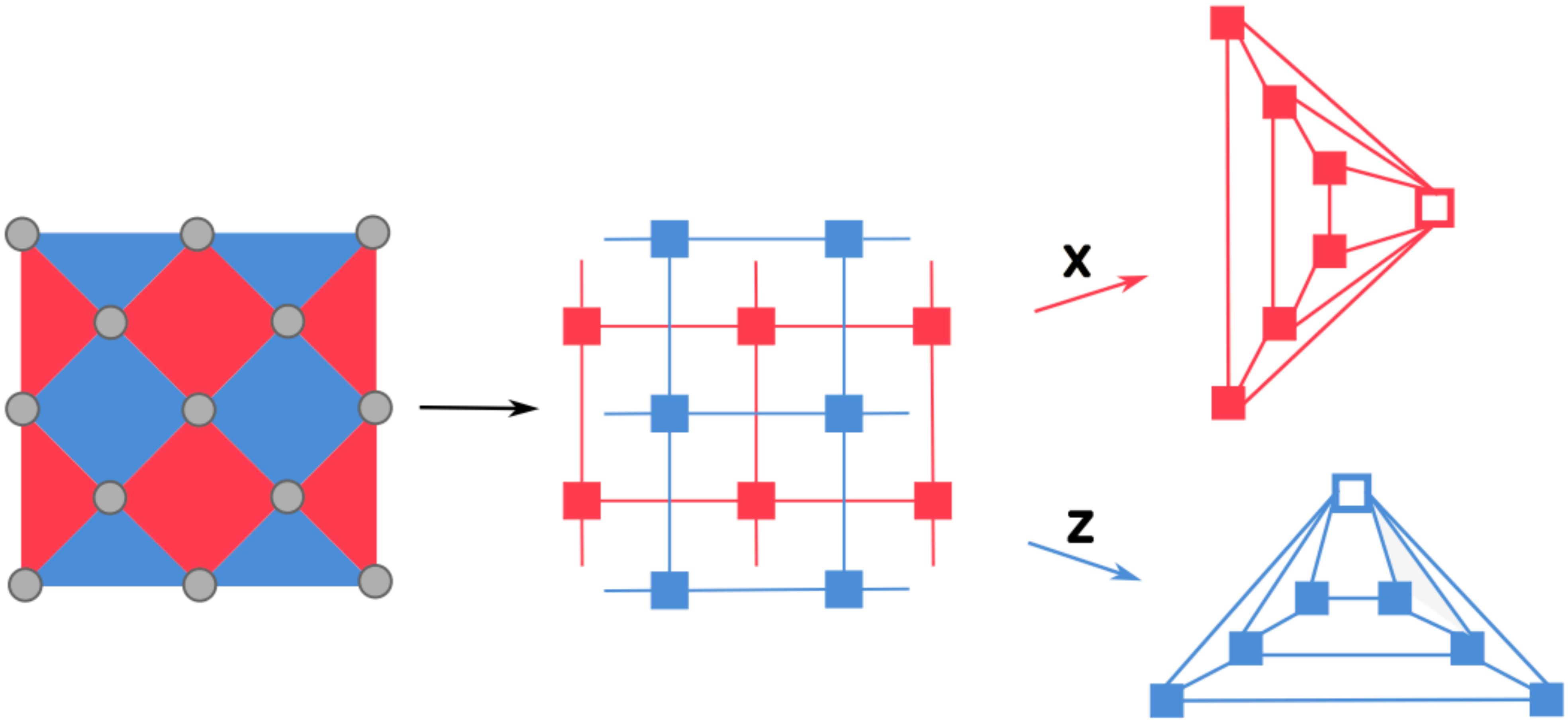
$E \Longleftrightarrow \{\alpha, \beta, \gamma\}$

- $\gamma \in \{1,0\}^m$, satisfying $Eg_i = (-1)^{\gamma_i} g_i E$ ***Syndrome***
- $\alpha \in \{1,0\}^m$, satisfying $Ee_i = (-1)^{\gamma_i} e_i E$
- $\beta^x \in \{1,0\}^k$, satisfying $El_x = (-1)^{\beta_i^x} l_i^x E$
- $\beta^z \in \{1,0\}^k$, satisfying $El_i^z = (-1)^{\beta_i^z} l_i^z E$

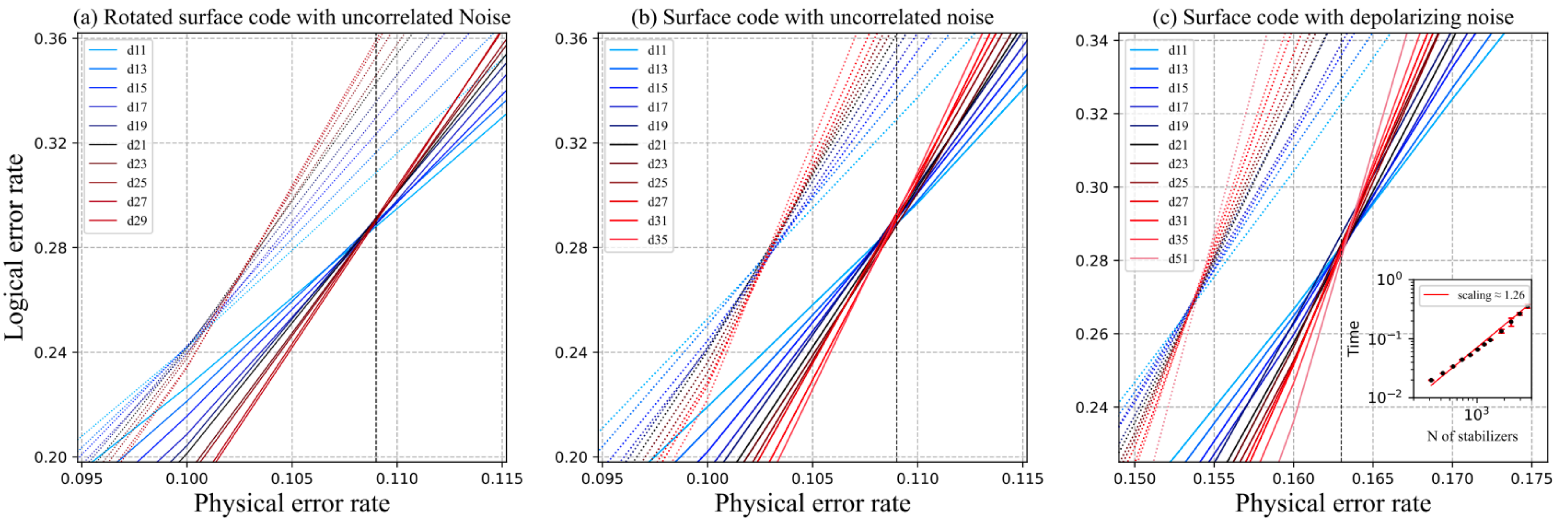
MLD: $Z(\beta, \gamma) = \sum_{\alpha} P(E(\alpha, \beta, \gamma))$ given γ and β

$\Longrightarrow$ **computing spin glass partition function**

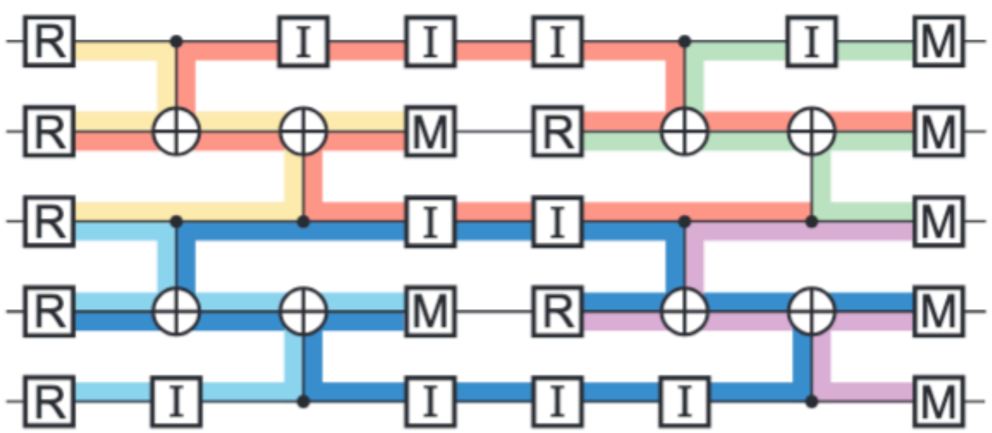
Surface code MLD到 自旋玻璃配分函数计算

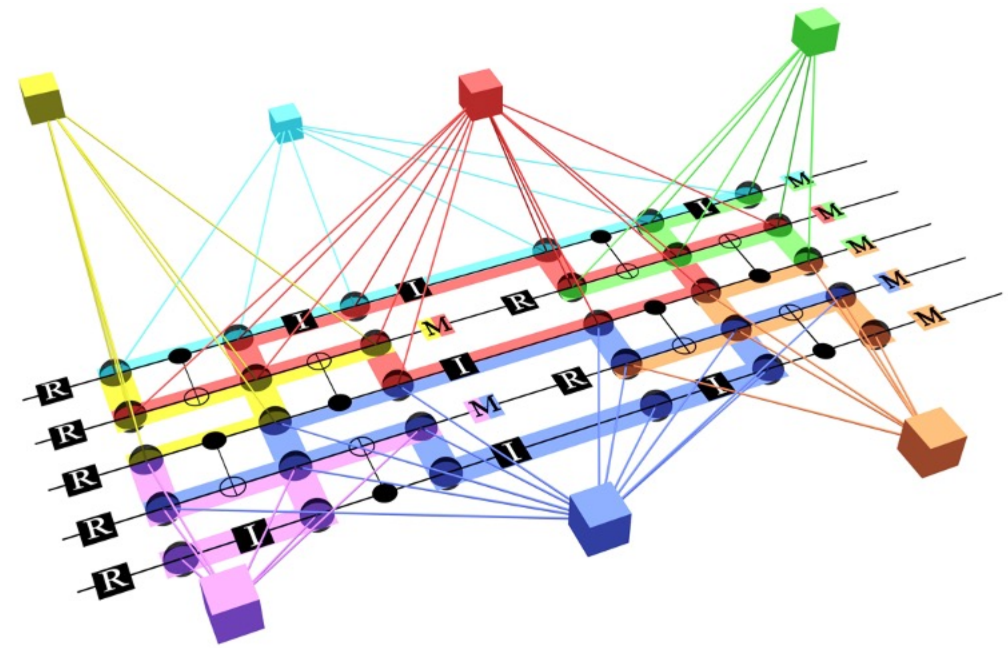


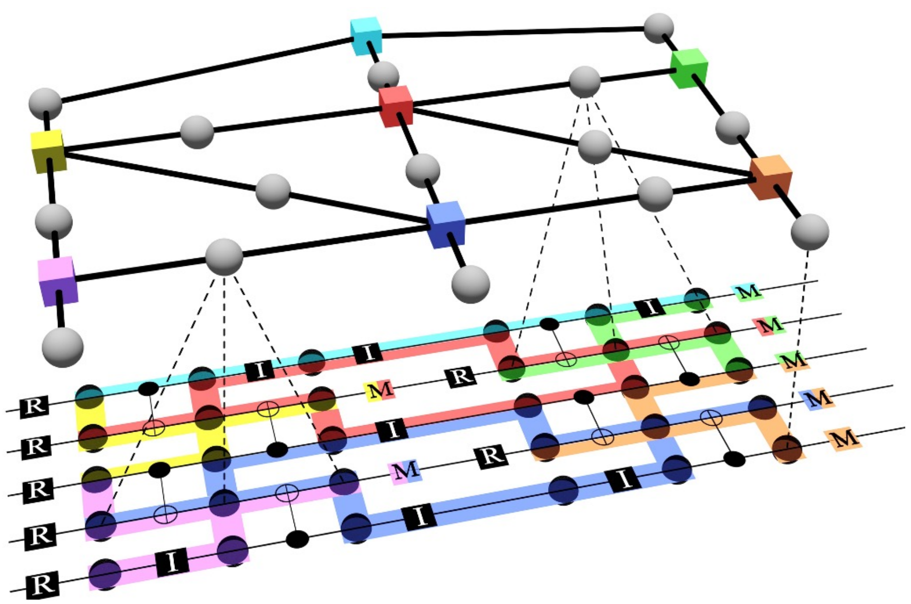
Surface code 结果

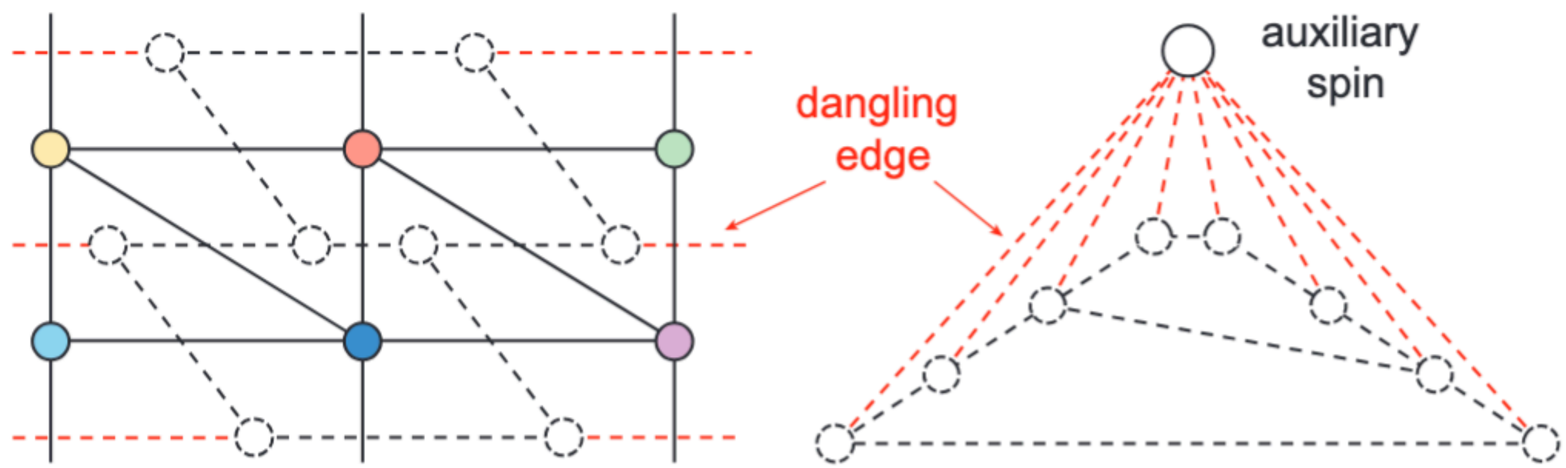


Repetition code MLD到自旋玻璃配分函数计算





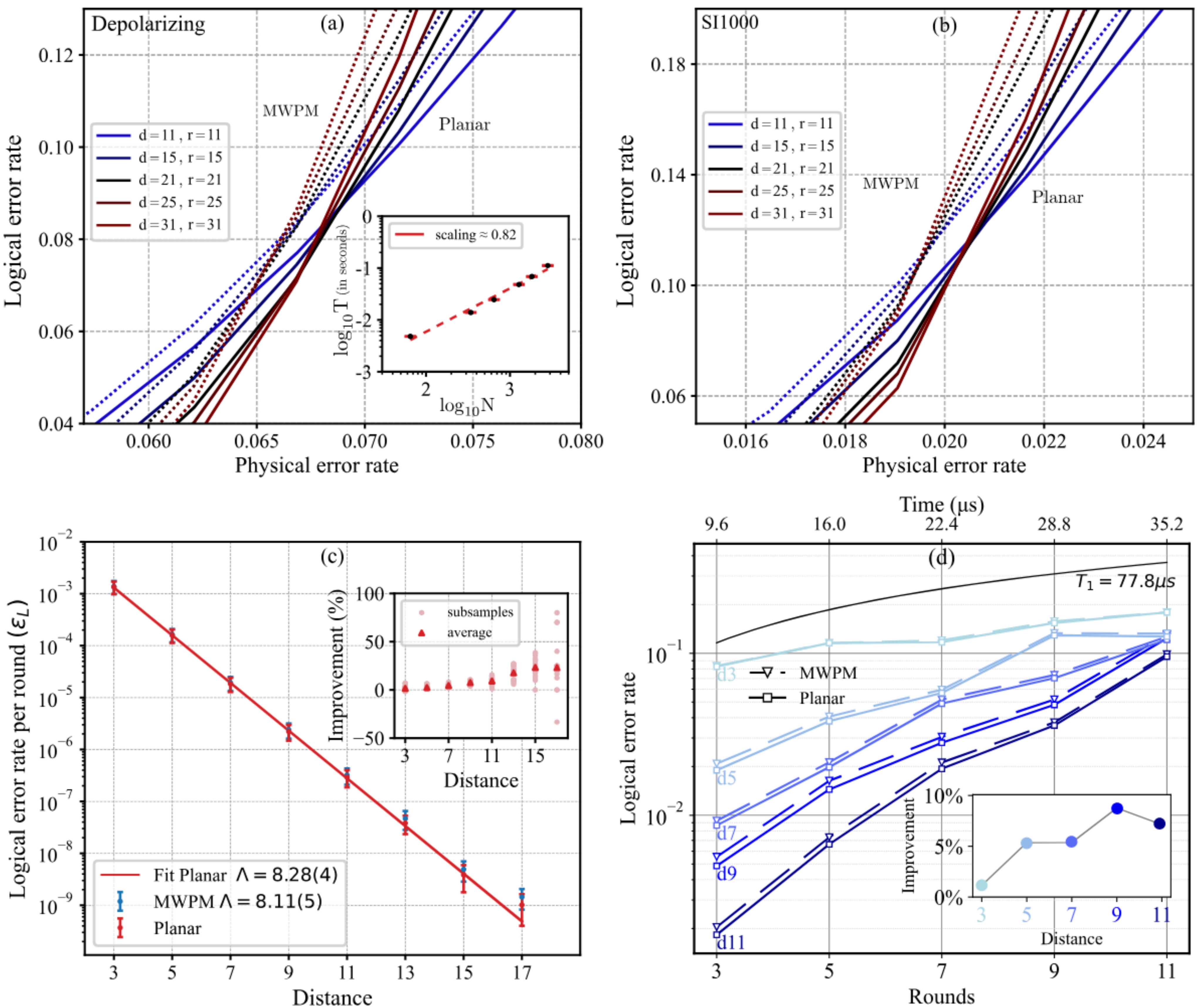




arXiv:2501.03582

Repetition code严格解

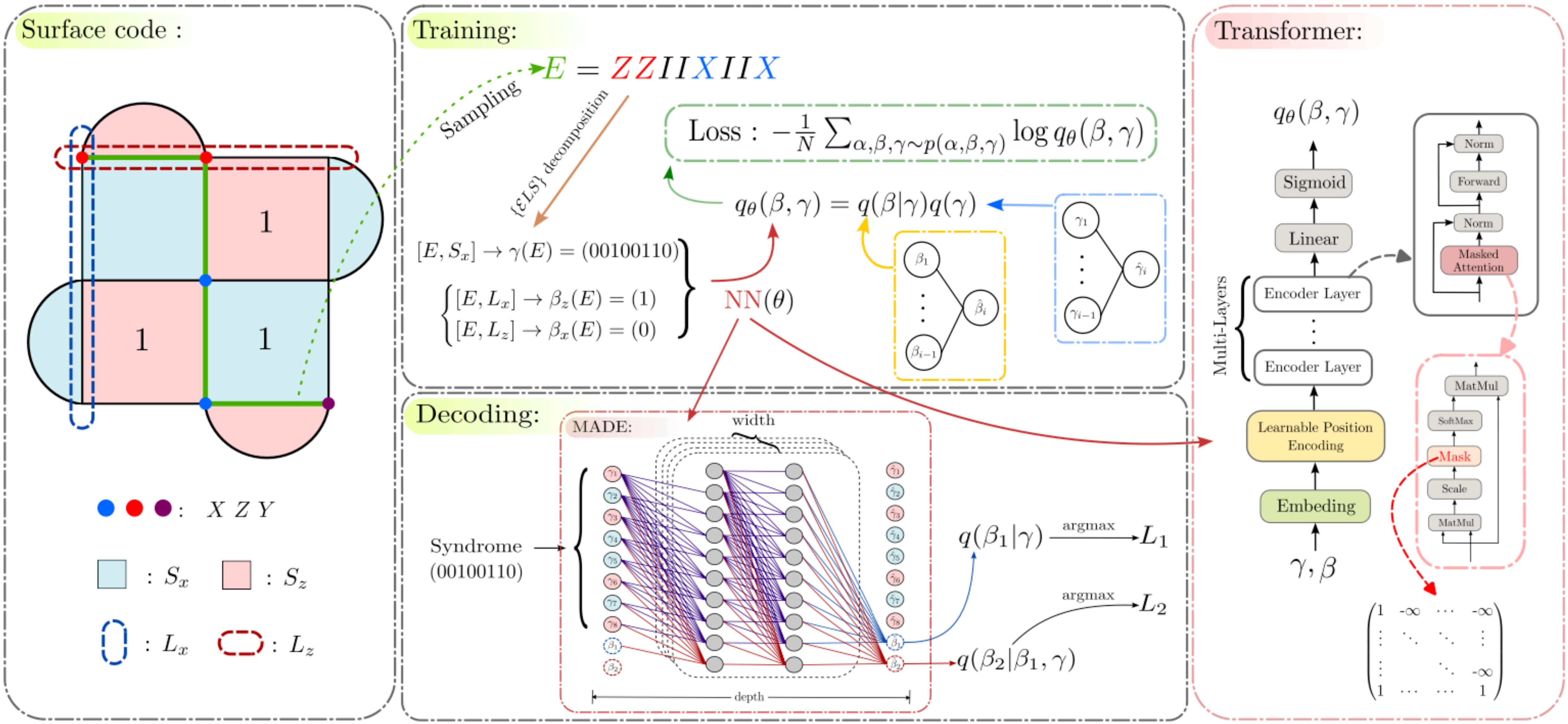
arXiv:2501.03582



Google试验

北京量子院试验

Generative neural decoding



Hanyan Cao, Feng Pan, Yijia Wang, PZ, arXiv:2307.09025

PZ group, unpublished

Generative decoding

Maximum likelihood decoding:

Evaluate all possible β configurations

Complexity: $O(4^k)$

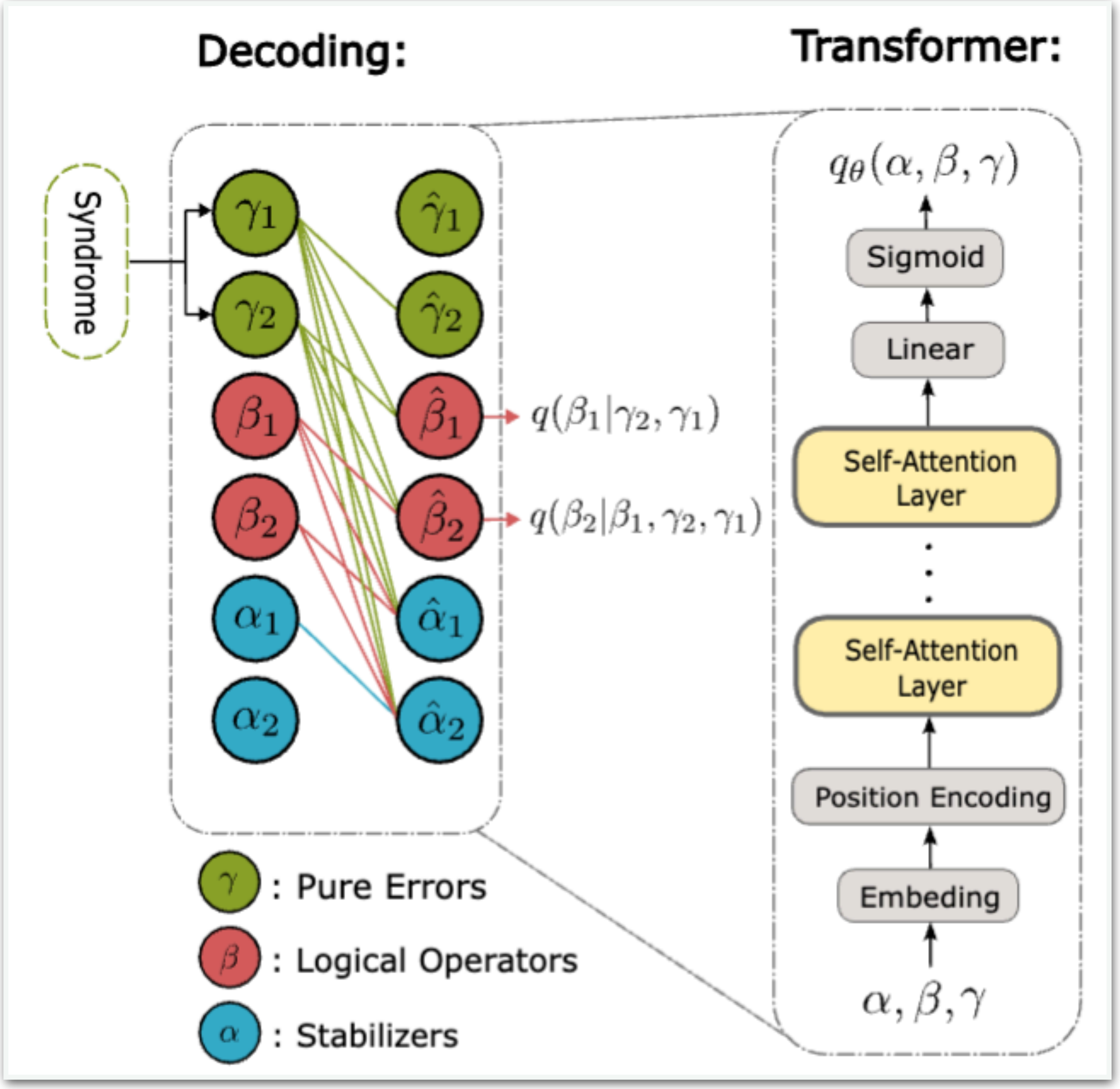
Generative decoding:

$$\hat{\beta}_1 = \arg\max_{\beta_1} q(\beta_1 | \gamma_2, \gamma_1)$$

$$\hat{\beta}_2 = \arg\max_{\beta_2} q(\beta_2 | \beta_1, \gamma_2, \gamma_1)$$

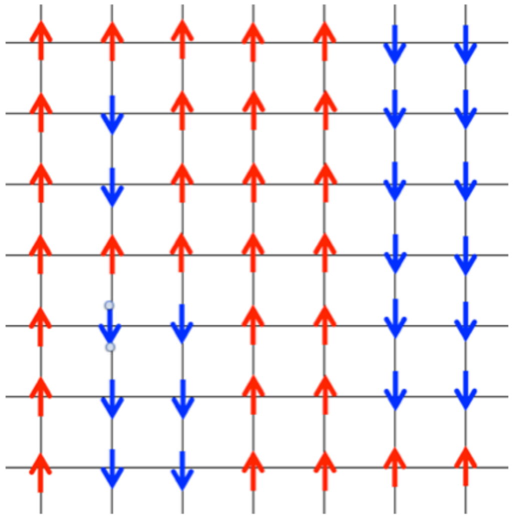
……

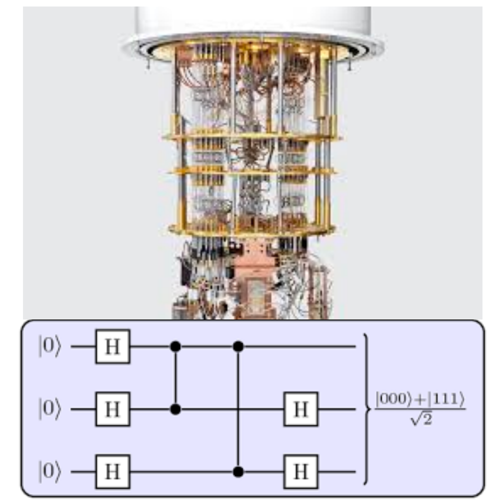
Complexity: $O(2k)$



PZ group, unpublished

统计物理，机器学习，与量子计算





微观构型分布

$$P(\sigma) = \frac{1}{Z}\exp(-\beta E(\sigma))$$

数据变量分布

$$P(\mathrm{Data})$$

控制高维空间量子态

$$|\psi\rangle$$

指数大的空间

有效的方法

强大的计算能力