PPP 2025, 60 min
Sep 3 Wed, 2025, 2:00 PM
YITP, Kyoto, Japan

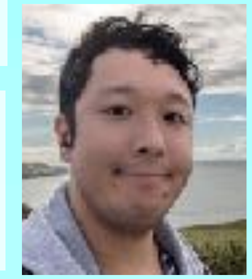
https://indico.yukawa.kyoto-u.ac.jp/event/71/contributions/1126/

This talk is based on
JPSJ 94 (2025) 3, 031006

AI技術の進展と素粒子物理学、格子QCDへの応用

Advances in AI Technologies and Their Applications to Particle Physics and Lattice QCD

富谷昭夫 [東京女子大学 (専任講師)
理研R-CCS (客員研究員)、京都大学 (特定准教授, 9/1より)]



MLPhys Foundation of 'Machine Learning Physics'
Grant-in-Aid for Transformative Research Areas (A)

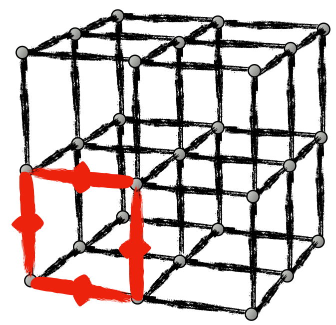
Program for Promoting Researches on the Supercomputer Fugaku
Large-scale lattice QCD simulation and development of AI technology

富岳

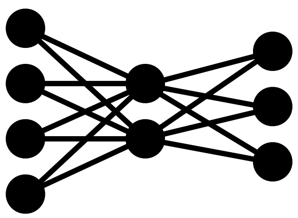
akio_at_yukawa.kyoto-u.ac.jp

Outline of my talk

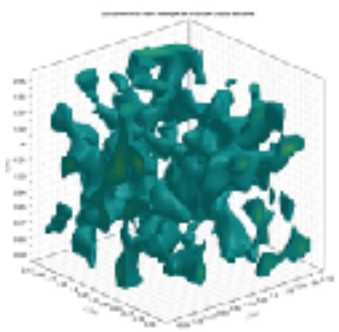
This talk is based on
JPSJ 94 (2025) 3, 031006



Lattice QCD?



Machine learning

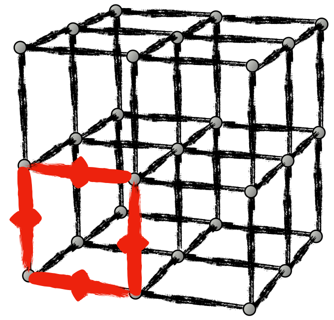


Production of configurations

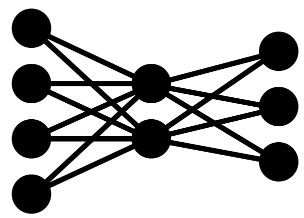
Slide

Outline of my talk

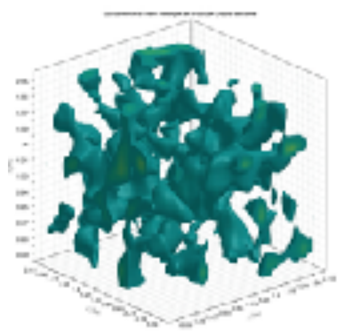
This talk is based on
JPSJ 94 (2025) 3, 031006



Lattice QCD?



Machine learning



Production of
configurations

Slide

Introduction

Lattice QCD = non-perturbative input for phenomenology

WIKIPEDIA
The Free Encyclopedia

Parameters of the Standard Model [hide]

	Symbol	Description	Renormalization scheme (point)	Value	
1	m_e	electron mass		0.510 998 950 69(16) MeV/c^2	
2	m_μ	muon mass		105.658 3755(23) MeV/c^2	
3	m_τ	tau mass		1 776.86(12) MeV/c^2	
4	m_u	up quark mass	$\mu_{\overline{MS}}$ = 2 GeV	$2.16^{+0.49}_{-0.26}$ MeV/c^2	Lattice QCD is needed
5	m_d	down quark mass	$\mu_{\overline{MS}}$ = 2 GeV	$4.67^{+0.48}_{-0.17}$ MeV/c^2	Lattice QCD is needed
6	m_s	strange quark mass	$\mu_{\overline{MS}}$ = 2 GeV	$93.4^{+8.6}_{-3.4}$ MeV/c^2	Lattice QCD is needed
7	m_c	charm quark mass	$\mu_{\overline{MS}} = m_c$	1.27(2) GeV/c^2	Lattice QCD is needed
8	m_b	bottom quark mass	$\mu_{\overline{MS}} = m_b$	$4.18^{+0.03}_{-0.02}$ GeV/c^2	Lattice QCD is needed
9	m_t	top quark mass	on-shell scheme	172.69(30) GeV/c^2	
10	θ_{12}	CKM 12-mixing angle		13.1°	Lattice QCD is needed
11	θ_{23}	CKM 23-mixing angle		2.4°	Lattice QCD is needed
12	θ_{13}	CKM 13-mixing angle		0.2°	Lattice QCD is needed
13	δ	CKM CP-violating Phase		0.995	Lattice QCD is needed
14	g_1 or g'	U(1) gauge coupling	$\mu_{\overline{MS}} = m_Z$	0.357	
15	g_2 or g	SU(2) gauge coupling	$\mu_{\overline{MS}} = m_Z$	0.652	
16	g_3 or g_s	SU(3) gauge coupling	$\mu_{\overline{MS}} = m_Z$	1.221	Lattice QCD is needed
17	θ_{QCD}	QCD vacuum angle		~0	
18	v	Higgs vacuum expectation value		246.2196(2) GeV/c^2	
19	m_H	Higgs mass		125.18(16) GeV/c^2	

https://en.wikipedia.org/wiki/Mathematical_formulation_of_the_Standard_Model

Introduction

Lattice QCD = non-perturbative input for phenomenology

$$|V_{us}|$$

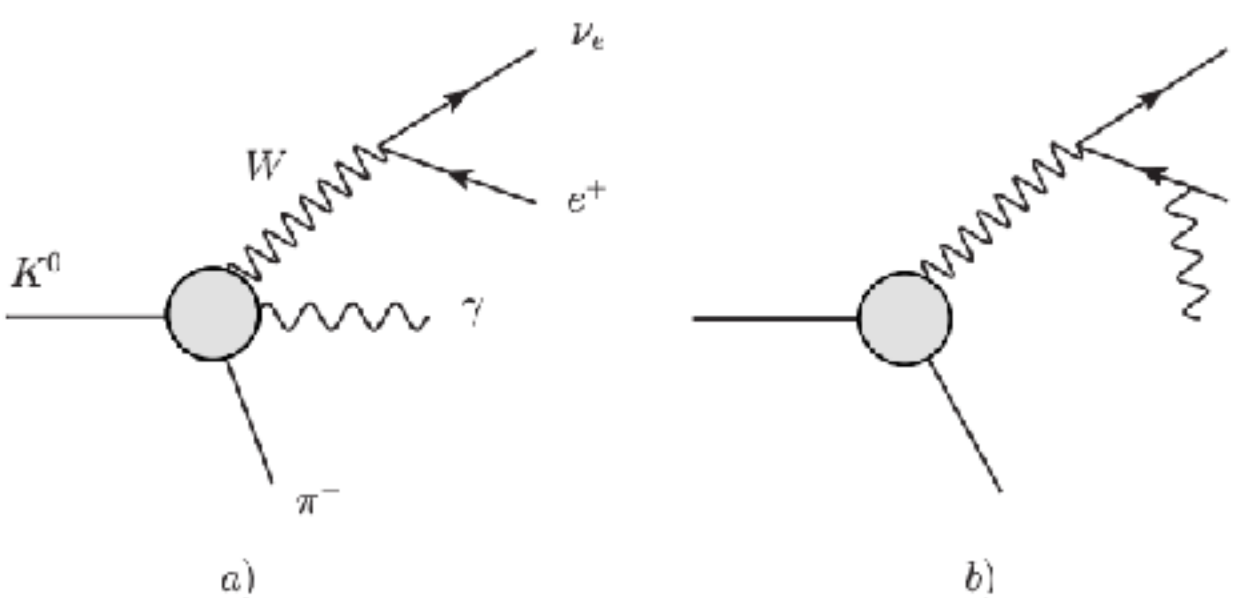


FIG. 4 Diagrams describing the $K^0_{\ell 3\gamma}$ amplitude.

Experimentalist

$$\Gamma^{(\mathrm{exp})}_{K\ell 3}$$

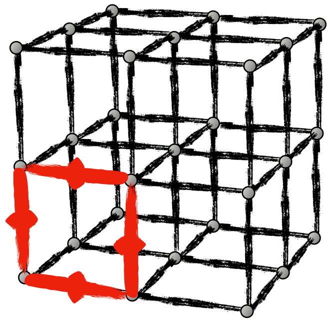
Compare

$$\Gamma^{(\mathrm{th})}_{K\ell 3}$$

Introduction

Lattice QCD = non-perturbative input for phenomenology

Lattice QCD



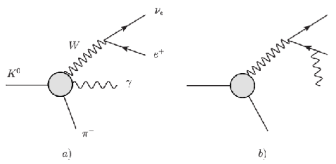


FIG. 4 Diagrams describing the $K^0_{\ell 3\gamma}$ amplitude.

Phenomenology

Experimentalist

$$\Gamma^{(\mathrm{exp})}_{K\ell 3}$$

Calculate

only from $(\Lambda_{\mathrm{QCD}}, m_{\mathrm{ud}}, m_{\mathrm{s}})$

Calculate

Compare

$$\overbrace{\left[f_+^{K\pi}(0)\right]^2}^{\text{Lattice form factor}} \times \overbrace{\frac{G_F^2 M_K^5}{192\pi^3}\left|V_{us}\right|^2 I(\lambda)}^{\text{Phenomenology}} = \Gamma^{(\mathrm{th})}_{K\ell 3}$$

SM parameter

Lattice QCD?

Akio Tomiya

Imaginary time is "Equivalent" to real time

Lattice calculation is done with "Euclidean time" and "path integral"

$$\langle \mathcal{O} \rangle = \frac{1}{Z} \lim_{T} \langle \Omega |\, \mathrm{T}\, e^{-\mathrm{i}H[\phi]T} \mathcal{O}(\phi) \,|\Omega\rangle$$

Operator, real time

Equivalent (⇄)

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

Path integral, real time

$t \to -\mathrm{i}t$ (⇅)

(same Hamiltonian)

$t \to -\mathrm{i}t$ (⇅)

(same eigenvalue of H)

$$\langle \mathcal{O} \rangle = \frac{1}{Z} \lim_{T} \langle \Omega |\, \mathrm{T}\, e^{-H[\phi]T} \mathcal{O}(\phi) \,|\Omega\rangle$$

Operator, **imaginary time**

Equivalent (⇄)

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

Path integral, imaginary time

Lattice QCD?

Akio Tomiya

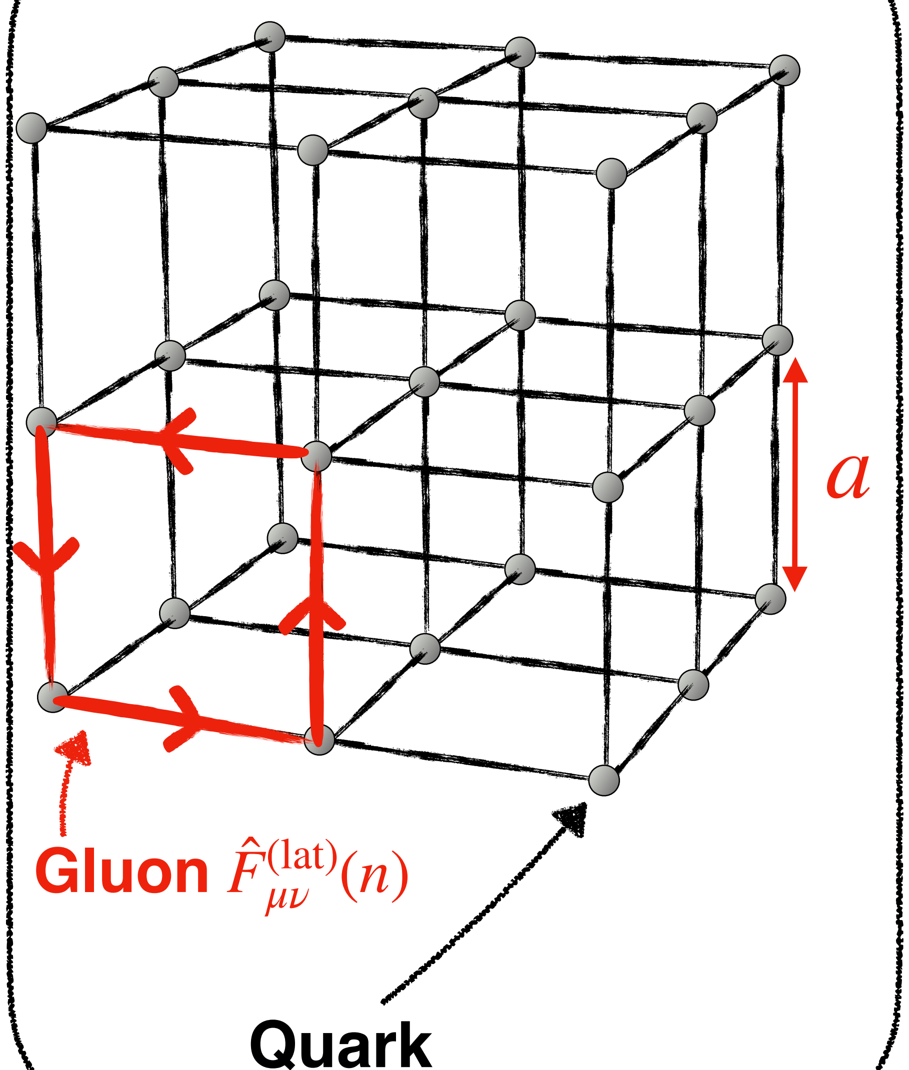
Lattice QCD = QCD on a discretized spacetime

Quantum expectation value $\langle \mathcal{O} \rangle = \frac{1}{Z} \int \underline{DU} e^{-S_{\rm QCD}[U]} \mathcal{O}(U)$ $\quad 10^{11}$ dim. integral

$$DU \equiv \prod_{n \in {\rm lat}} \prod_{\mu=1}^{4} {\rm d}U_\mu(n)$$

Finite dim. integral!

4 dimensional lattice



a [fm] is a lattice spacing (unit of discretization) needed to define the theory (as differentiation)

Theory on the lattice spacetime

$$S_{\rm QCD}[U] = \sum_n \left[-\frac{1}{g^2} {\rm Re\ tr}\ e^{ia^2 \hat{F}^{\rm (lat)}_{\mu\nu}[U]} \right] + ({\rm Quarks})$$

(after all calculations, we take $a \to 0$ limit with tuning of g)

This is just finite multi-dimensional ($256^4 \times 4 \times 8 \approx 10^{11}$ dimensional) integration.
Not a simulation but a just integration of regulated theory.

We evaluate this integral using Markov-Chain Monte Carlo

Lattice QCD?

Akio Tomiya

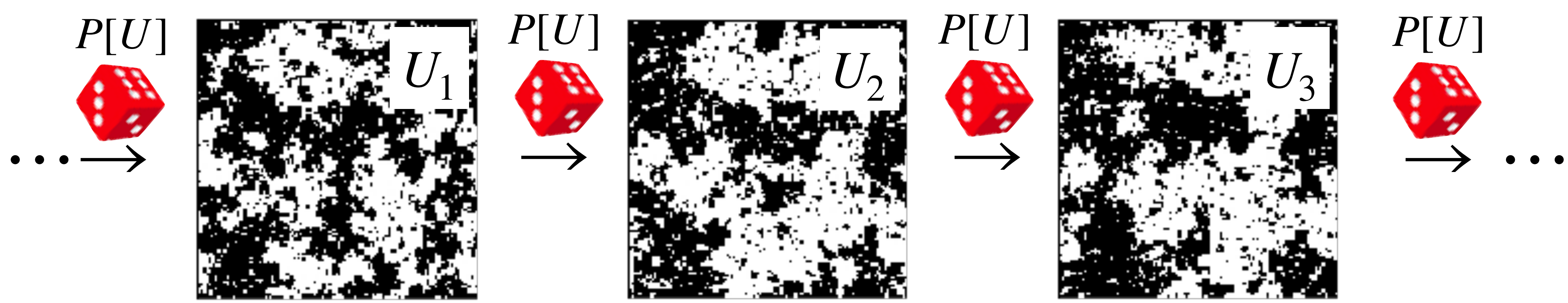
Monte-Carlo integration is available

HMC: Simon Duane, Anthony Kennedy, Brian Pendleton and Duncan Roweth1987

Quantum expectation value $\langle \mathcal{O} \rangle = \frac{1}{Z} \int \underline{DU} e^{-S_{\rm QCD}[U]} \mathcal{O}(U)$ $\quad S_{\rm QCD}[U] = S_{\rm gauge}[U] - \log \det(\not{D}[U] + m)$

10^{11} dim. integral

Monte-Carlo: Generate field configurations with "$P[U] \propto e^{-S_{\rm eff}[U]}$". Stochastically estimate $\langle \mathcal{O} \rangle$



Lattice QCD?

Monte-Carlo integration is available

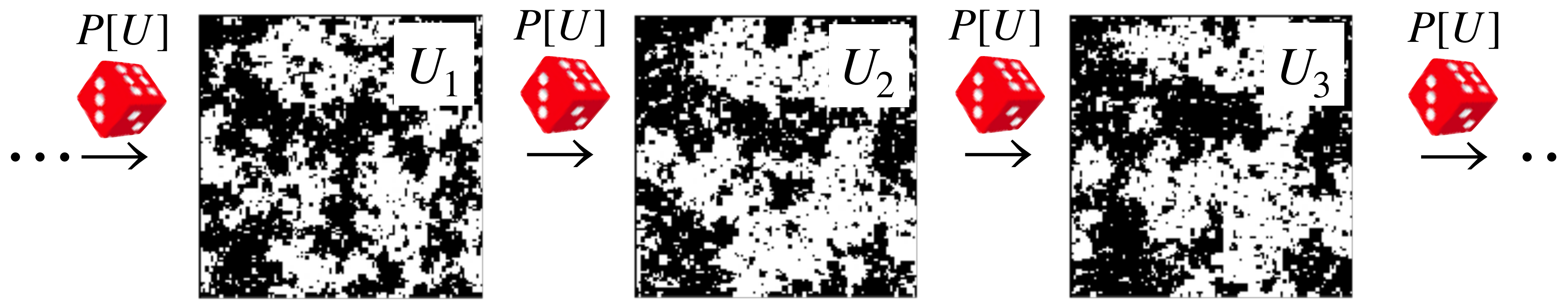
Akio Tomiya

HMC: Simon Duane, Anthony Kennedy, Brian Pendleton and Duncan Roweth1987

Quantum expectation value $$\langle \mathcal{O} \rangle = \frac{1}{Z} \int \underline{DU} e^{-S_{\rm QCD}[U]} \mathcal{O}(U)$$ 10^{11} dim. integral

$$S_{\rm QCD}[U] = S_{\rm gauge}[U] - \log\det(\not{D}[U] + m)$$

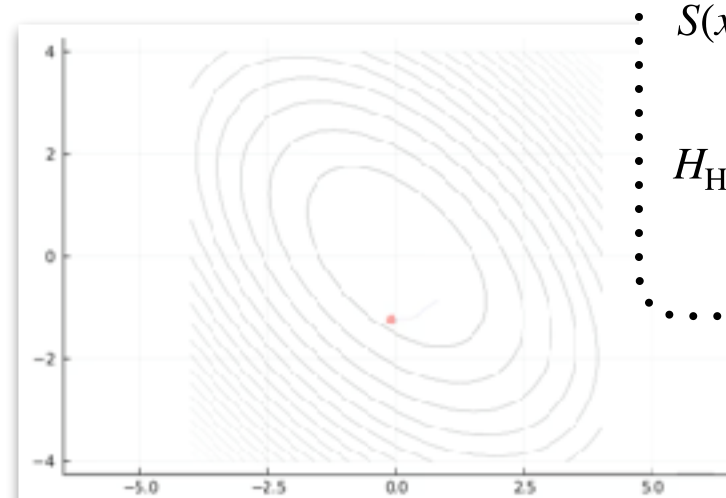
Monte-Carlo: Generate field configurations with "$P[U] \propto e^{-S_{\rm eff}[U]}$". Stochastically estimate $\langle \mathcal{O} \rangle$



 = Hybrid/Hamiltonian Monte-Carlo (HMC)
(De-facto standard Exact algorithm)

= Random momentum + EOM
Here we *regard* $S_{\rm QCD}$ as a potential for U

≈Molecular dynamics with random p & given U



$$S(x,y) = \frac{1}{2}(x^2 + y^2 + xy)$$
$$H_{\rm HMC} = \frac{p_x^2}{2} + \frac{p_y^2}{2} + S(x,y)$$
init p_x, p_y = random

$$\langle \mathcal{O} \rangle = \frac{1}{N_{\rm sample}} \sum_{k=1}^{N_{\rm sample}} \mathcal{O}[U_k] \quad (N_{\rm sample} \to \infty)$$

Introduction

Akio Tomiya

Monte-Carlo integration is available

M. Creutz 1980

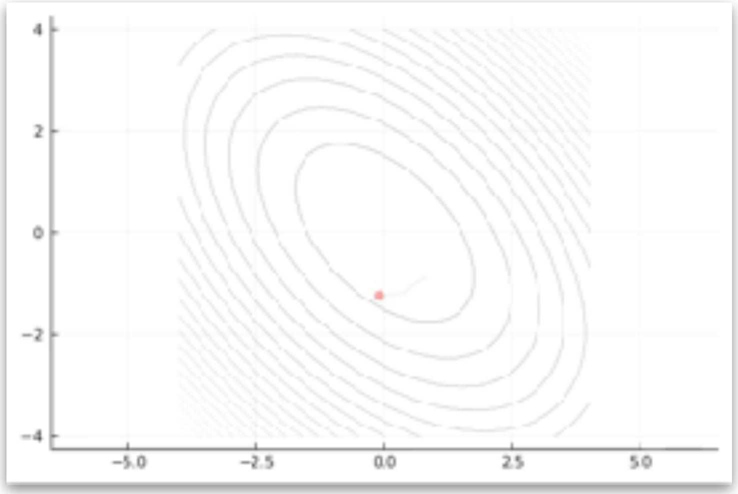
Quantum expectation value $\langle \mathcal{O} \rangle = \frac{1}{Z}\int \underline{DU} e^{-S_{\mathrm{QCD}}[U]}\mathcal{O}(U)$

10^{11} dim. integral

$S_{\mathrm{QCD}}[U] = S_{\mathrm{gauge}}[U] - \log\det(\not{D}[U] + m)$

HMC: Hybrid (Hamiltonian) Monte-Carlo
De-facto standard algorithm (Exact)

Random momentum + EOM
= Random walk like algorithm



$S(x, y) = \frac{1}{2}(x^2 + y^2 + xy)$

In each time step of “EOM”, we have to solve a linear equation $\not{D}$ (Dslash = covariant derivative), which is very expensive. **It dominates 50-90 % of numerical cost.**

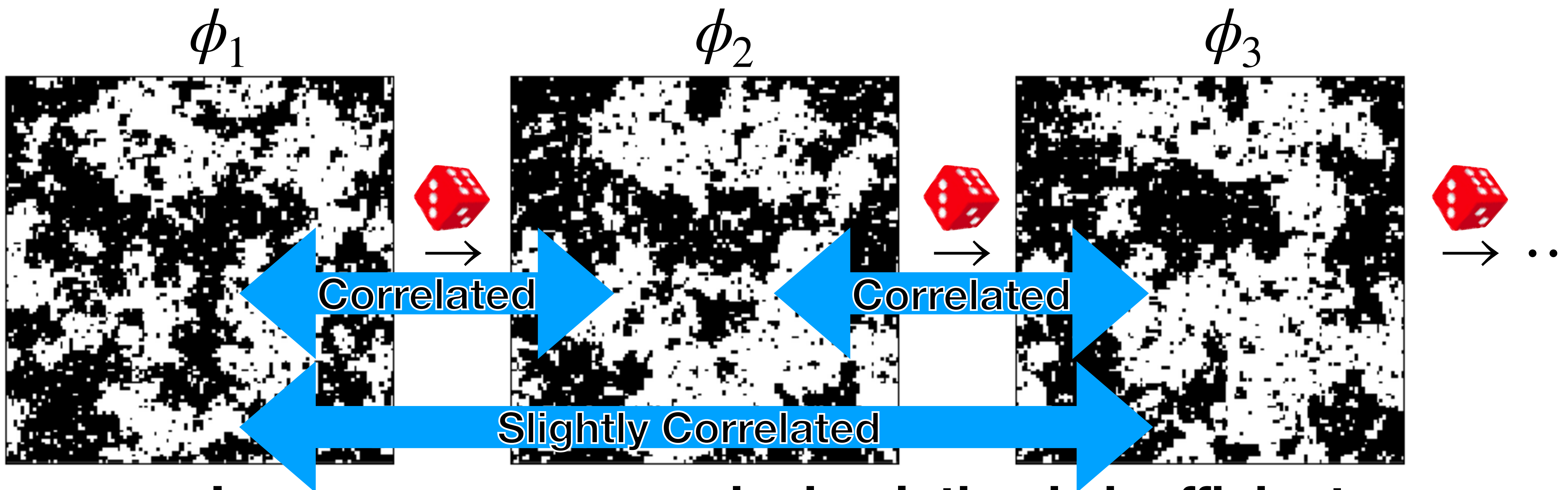
$$\not{D}\vec{x} = \vec{b}$$

A huge Linear equation. it is solved by the conjugate gradient

10^9 dimension for L = 256 (L^4 is the system size)

Introduction

Correlation between samples = inefficiency of calculation

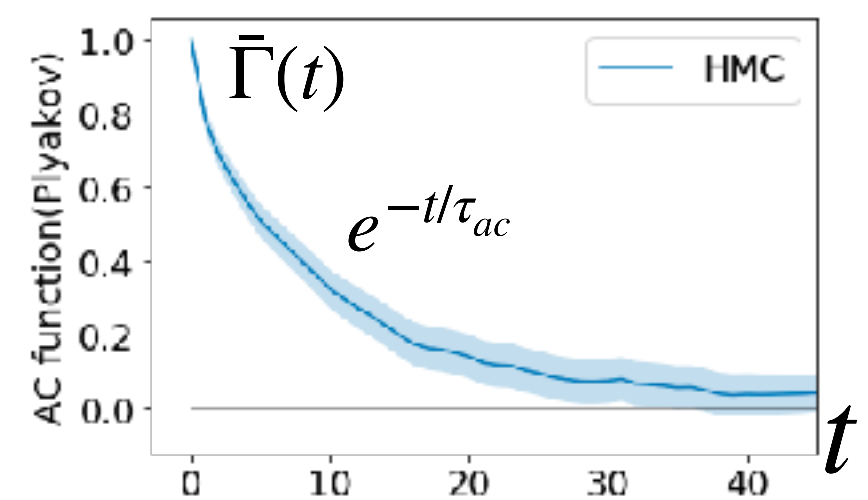


Large τ_{ac} means, such simulation is inefficient

$$\langle O[\phi] \rangle = \frac{1}{N}\sum_{k}^{N} O[\phi_k] \ \pm\ O\Big(\frac{1}{\sqrt{N_{\text{indep}}}}\Big)$$

$$N_{\text{indep}} = \frac{N_{\text{sample}}}{2\tau_{ac}}$$

$$\bar{\Gamma}(t) = \frac{1}{N-t}\sum_{k} (O[\phi_{k+t}] - \bar{O})(O[\phi_k] - \bar{O}) \ \sim e^{-t/\tau_{ac}}$$



Introduction

MCMC is not “totally random” -> auto-correlation

Data from
Nf=3, standard staggered
with magnetic field

$L^3 \times N_t = 16^3 \times 4$

$ma = 0.03$

β=6/g²	N_{conf}	τ_{ac}	N_{indep}
5.166	15k	47	160
5.167	20k	224	45
5.168	20k	656	15
5.169	**20k**	2940	3
5.170	15k	1306	6
5.171	14k	58	116
5.172	10k	48	106

k=1000

Critical temp.

$$N_{indep} = \frac{N_{conf}}{2\tau_{ac}}$$

$$\langle O[\phi] \rangle = \frac{1}{N_{conf}} \sum_{k}^{N_{conf}} O[\phi_k] \pm O\left(\frac{1}{\sqrt{N_{indep}}}\right)$$

$$\tau_{ac} \sim \xi^z \sim L^z$$

z : Dynamic critical exponent (see 1703.03136)

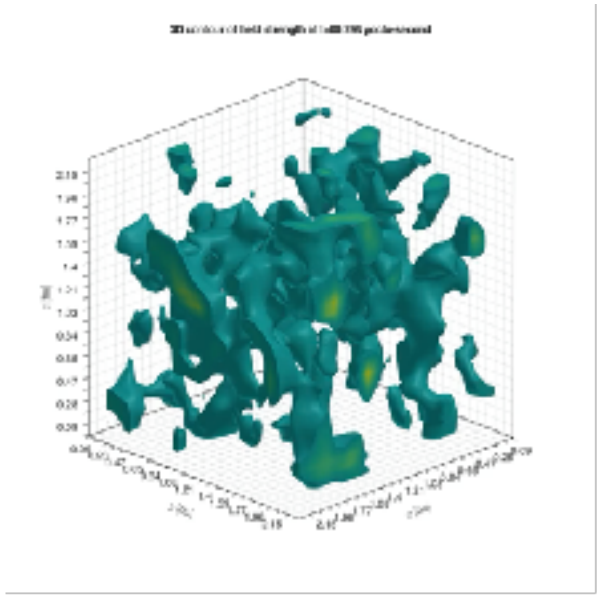
τ_{ac}: **algorithm dependent** (N. Madras et. al 1988)

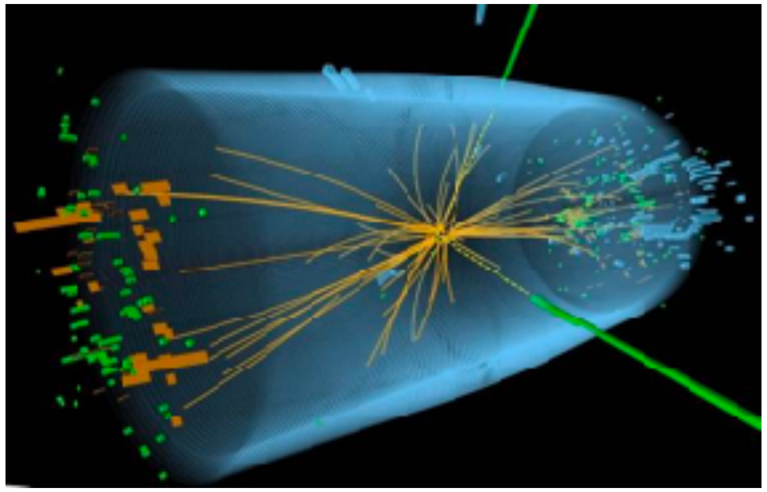
If one finds an algorithm with small z (small tau), we can reduce the numerical cost

Can we use ML?

Lattice QCD?

What is our final goal for our research field?





Form factors,
Running coupling,
Equation of states,
Topological susceptibility
etc.

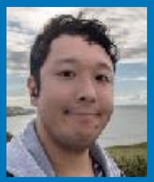
Parameters of the Standard Model [hide]

Symbol	Description	Renormalization scheme (point)	Value
m_e	electron mass		0.510 998 950 69(16) MeV/c^2
m_μ	muon mass		105.658 3755(23) MeV/c^2
m_τ	tau mass		1 776.86(12) MeV/c^2
m_u	up quark mass	$\mu_{\overline{MS}} = 2$ GeV	$2.16^{+0.49}_{-0.26}$ MeV/c^2
m_d	down quark mass	$\mu_{\overline{MS}} = 2$ GeV	$4.67^{+0.48}_{-0.17}$ MeV/c^2
m_s	strange quark mass	$\mu_{\overline{MS}} = 2$ GeV	$93.4^{+8.6}_{-3.4}$ MeV/c^2
m_c	charm quark mass	$\mu_{\overline{MS}} = m_c$	1.27(2) GeV/c^2
m_b	bottom quark mass	$\mu_{\overline{MS}} = m_b$	$4.18^{+0.03}_{-0.02}$ GeV/c^2
m_t	top quark mass	on-shell scheme	172.69(30) GeV/c^2
θ_{12}	CKM 12-mixing angle		13.1°
θ_{23}	CKM 23-mixing angle		2.4°
θ_{13}	CKM 13-mixing angle		0.2°
δ	CKM CP-violating Phase		0.995
g_1 or g'	U(1) gauge coupling	$\mu_{\overline{MS}} = m_Z$	0.357
g_2 or g	SU(2) gauge coupling	$\mu_{\overline{MS}} = m_Z$	0.652
g_3 or g_s	SU(3) gauge coupling	$\mu_{\overline{MS}} = m_Z$	1.221
θ_{QCD}	QCD vacuum angle		~ 0
v	Higgs vacuum expectation value		246.2196(2) GeV/c^2
m_H	Higgs mass		125.18(16) GeV/c^2

Lattice QCD?

Open source LQCD code in Julia Language

Akio Tomiya




AT & Y. Nagai

Julia lang = Fast as C, Productive as Python, ML friendly, Portable

arXiv:1209.5145

Introductive notebook: http://bit.ly/4mej3oe

Not as Python, other languages are not necessary since it is fast

LatticeQCD.jl

Run almost everywhere: Laptop/Colab/Jupyter/Supercomputers

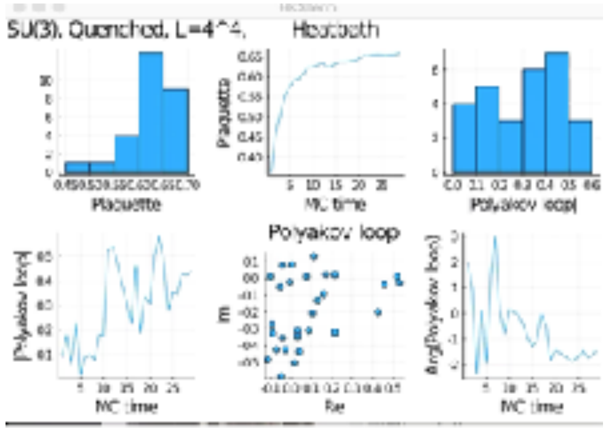
Advantage: Portability, no-explicit compile, fast, Quick trial-and-error feedback loop

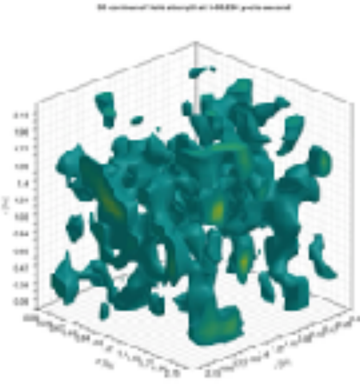
Functionality: SU(Nc)-heatbath, (R)HMC, Self-learning HMC, Dynamical fermions
Measurements (chiral condensate, topological charge, etc), MPI, GPU

Start LQCD in **5 min**

1. Download Julia binary
2. Add the package through Julia package manager
3. Execute!

https://github.com/akio-tomiya/LatticeQCD.jl





Code

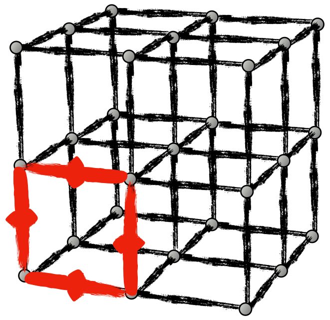
Lattice QCD?

Akio Tomiya

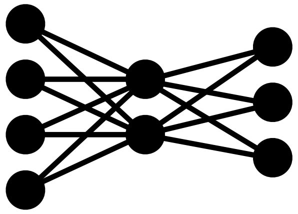
Demo!

Outline of my talk

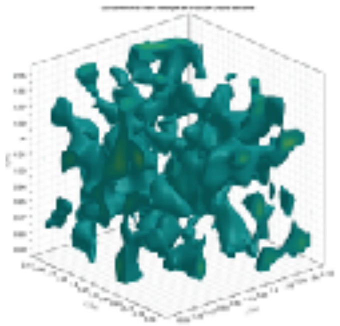
This talk is based on
JPSJ 94 (2025) 3, 031006



Lattice QCD?



Machine learning



Production of configurations

Slide

Machine learning

Akio Tomiya

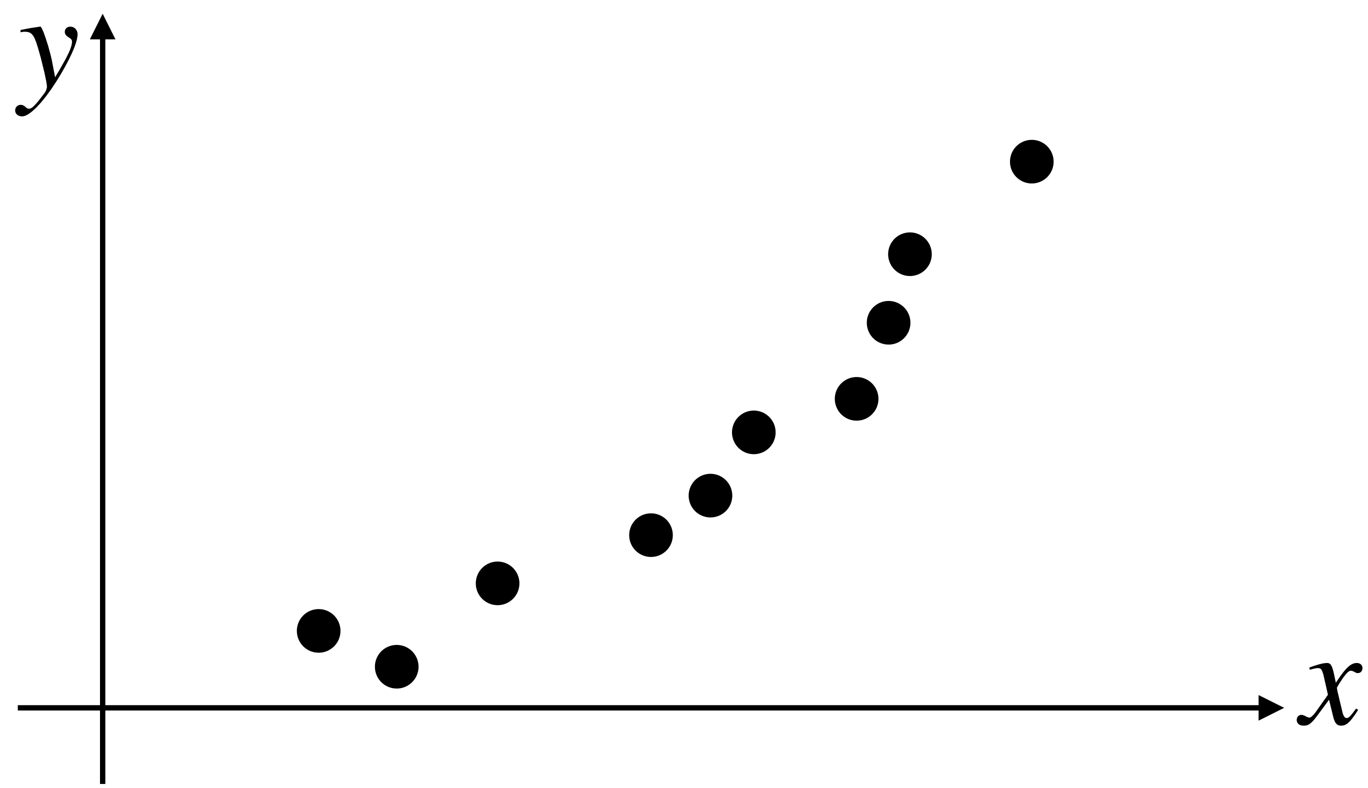
History of 3rd AI boom (4th industrial revolution)

Year	Event	Tomiya
2012	Breakthrough with CNN. AlexNet wins ILSVRC 2012	Master's degree
2013	**Higgs discovery**	
2015		PhD @ U. of Osaka → Wuhan
2016	AlphaGo wins	I started machine learning
2017	Transformer	
2018		Wuhan -> BNL
2020	GPT-3	Invited Talk in PPP, ML+Phys
2021	AlphaFold2	BNL → IPUT Osaka (Faculty)
2022	ChatGPT released.	「学習物理学」(MLPhys)
2024	Nobel Prize in Physics (Hopfield, Hinton), Chemistry	IPUT Osaka → TWCU (Faculty)
2025	?	Invited Talk in PPP, ML+Phys

What is machine learning?

E.g. Linear regression ∈ Supervised learning

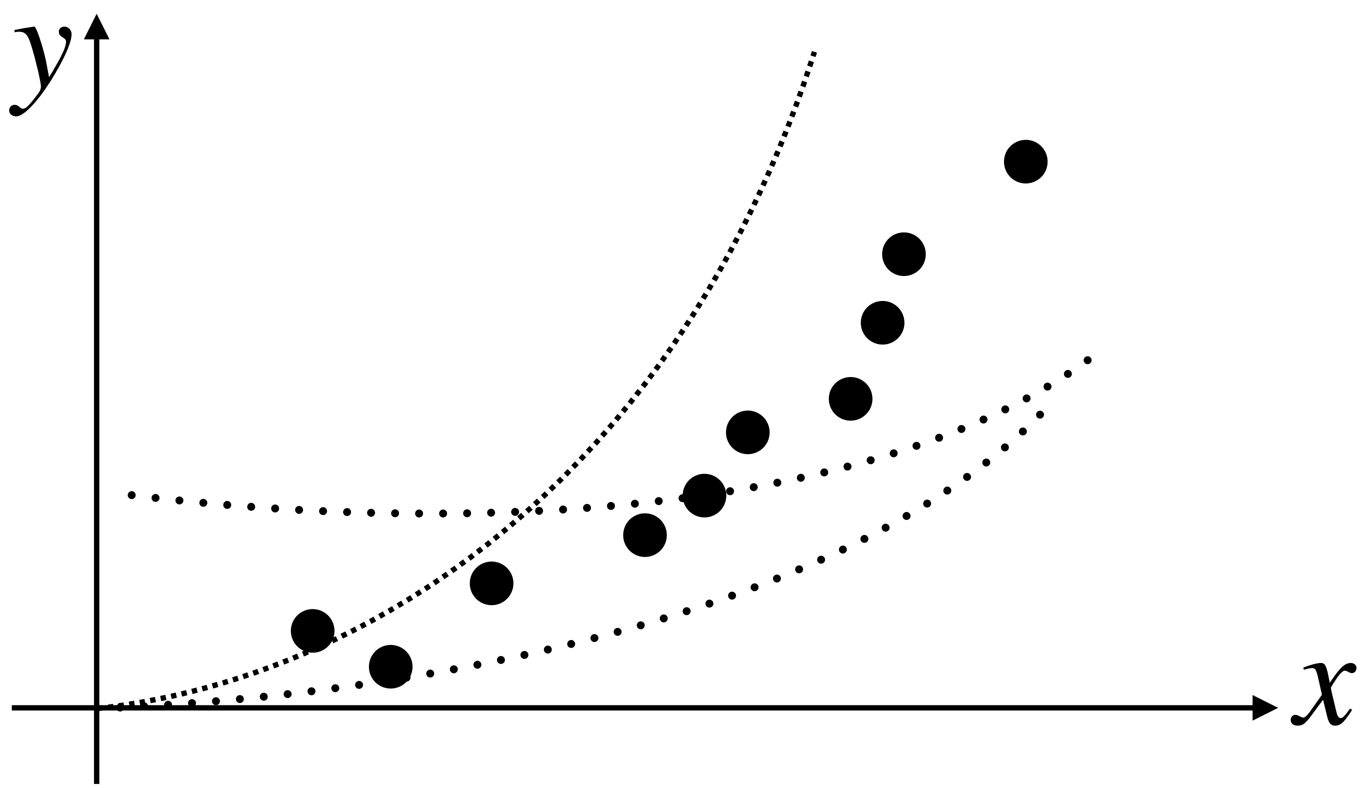
Data: $D = \left\{ (x^{(1)}, y^{(1)}),\ (x^{(2)}, y^{(2)}),\ \cdots \right\}$



What is machine learning?

E.g. Linear regression ∈ Supervised learning

Data: $D = \left\{(x^{(1)}, y^{(1)}),\ (x^{(2)}, y^{(2)}),\ \cdots\right\}$



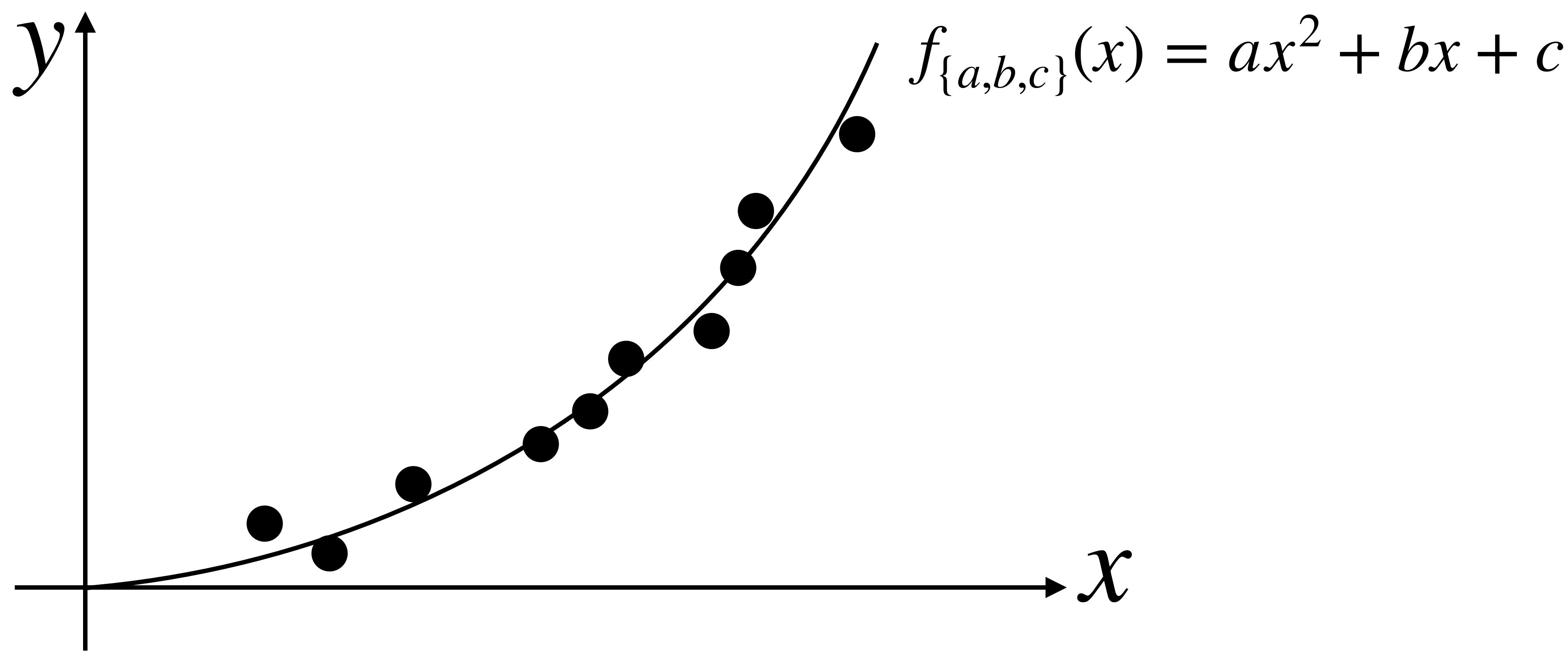
$$f_{\{a,b,c\}}(x) = ax^2 + bx + c \qquad E = \frac{1}{2}\sum_d \left|f_{\{a,b,c\}}(x^{(d)}) - y^{(d)}\right|^2$$

a, b, c, are determined by minimizing E
(training = fitting by data)

What is machine learning?

E.g. Linear regression ∈ Supervised learning

Data: $D = \left\{ (x^{(1)}, y^{(1)}), \, (x^{(2)}, y^{(2)}), \, \cdots \right\}$



$f_{\{a,b,c\}}(x) = ax^2 + bx + c$ $\qquad$ $E = \frac{1}{2} \sum_d \left| f_{\{a,b,c\}}(x^{(d)}) - y^{(d)} \right|^2$

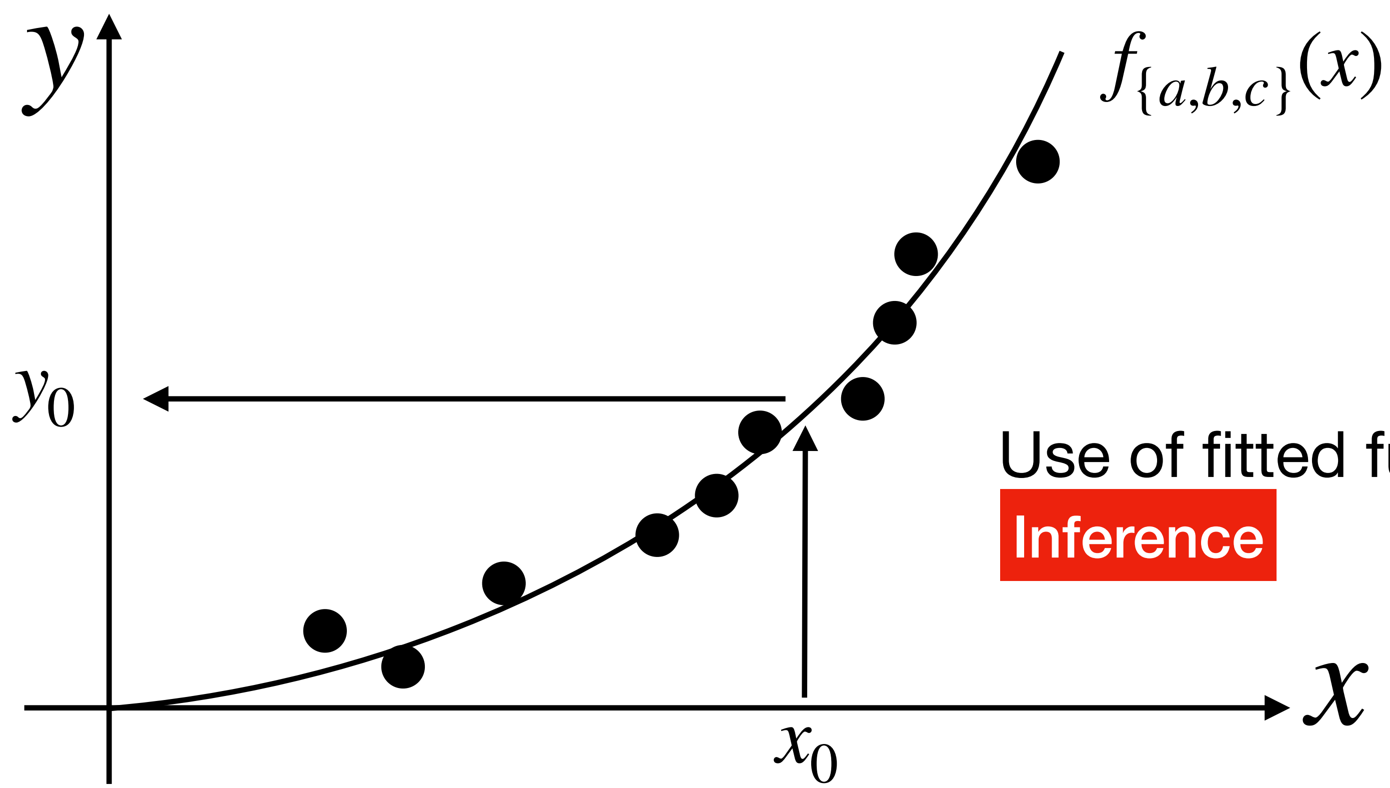
$a, b, c,$ are determined by minimizing E
(training = fitting by data)

What is machine learning?

E.g. Linear regression ∈ Supervised learning

Data: $D = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots\}$

$f_{\{a,b,c\}}(x) = ax^2 + bx + c$



Use of fitted function = **Inference**

Now we can predict y value which not in the data

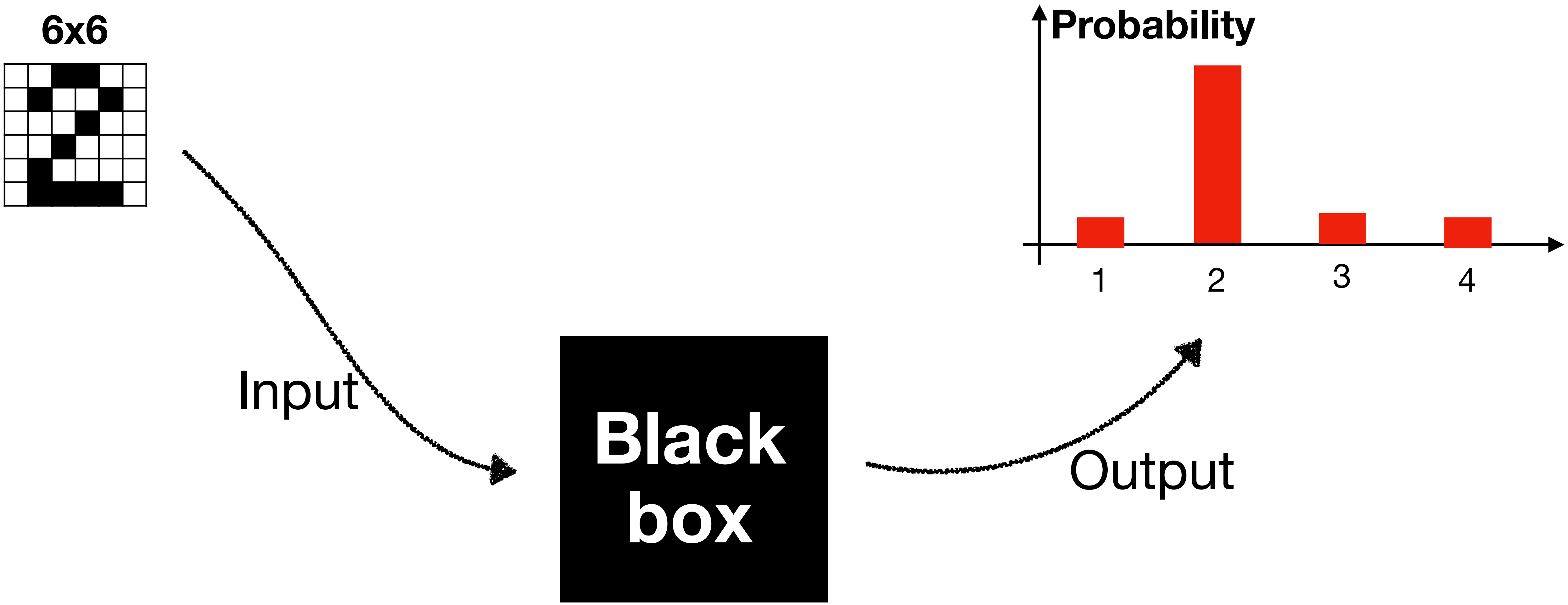
In physics language, variational method

What is the neural networks?

Akio Tomiya

Neural network is a *universal* approximation function

Example: Recognition of hand-written numbers (0-9)



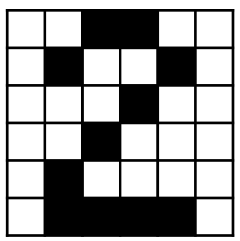
How can we formulate this “Black box”?
Ansatz?

What is the neural networks?

Neural network is a *universal* approximation function

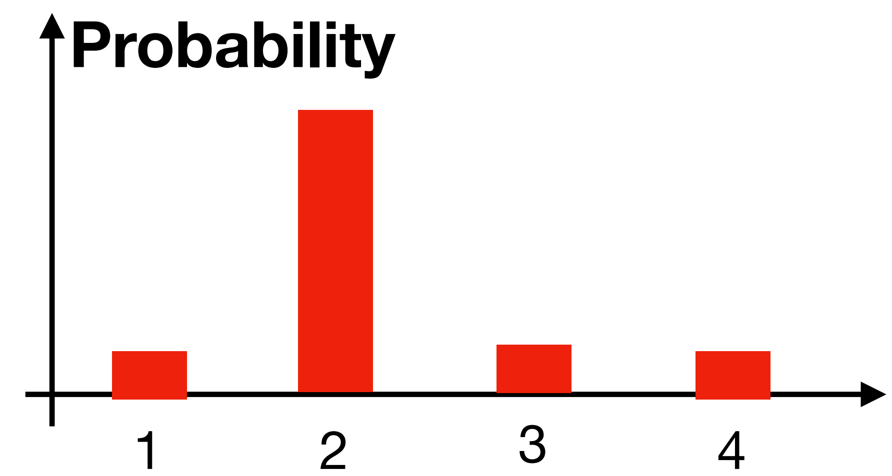
Example: Recognition of hand-written numbers (0-9)

6x6



$$= \begin{pmatrix} 0.000 \\ 0.000 \\ 0.8434 \\ 0.756 \\ 0.3456 \\ 0.64 \\ 0.251 \\ \vdots \end{pmatrix} = \vec{x}$$

Image is a vector (6x6=36 dim)



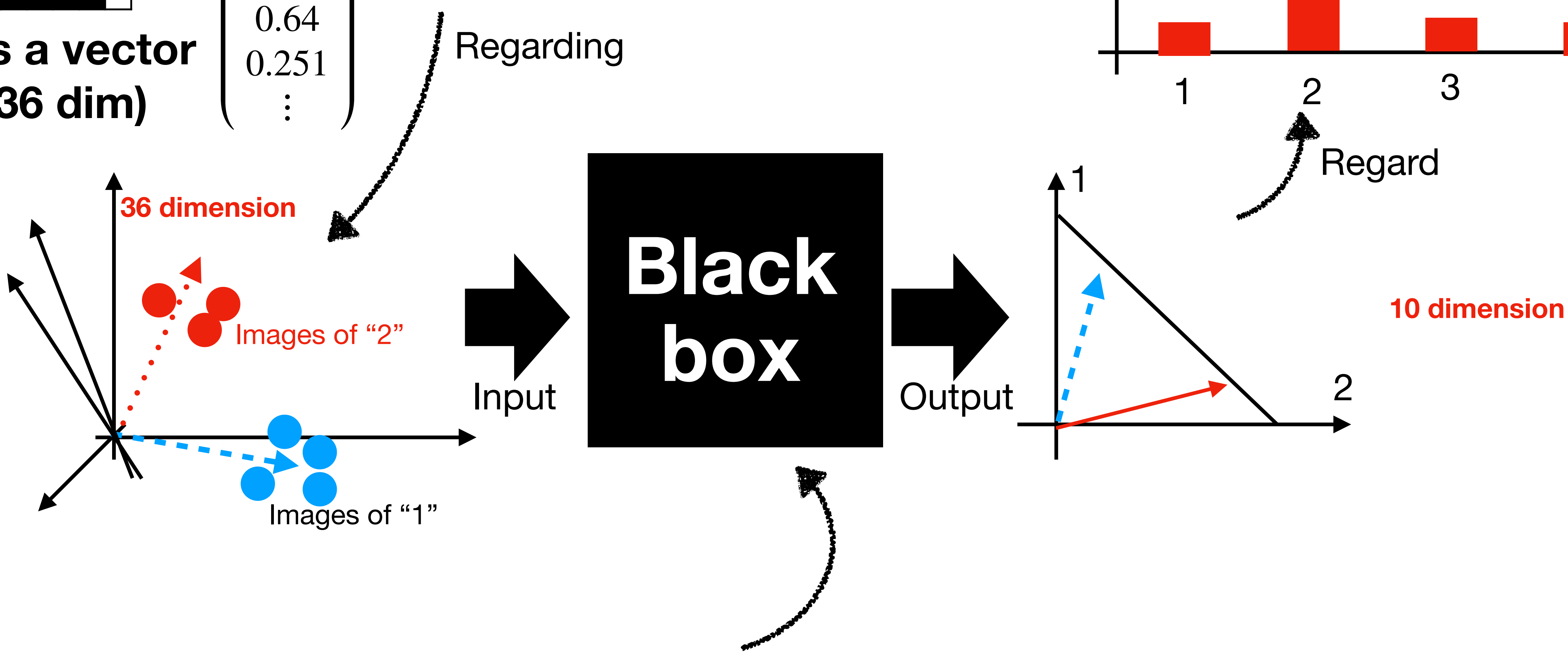


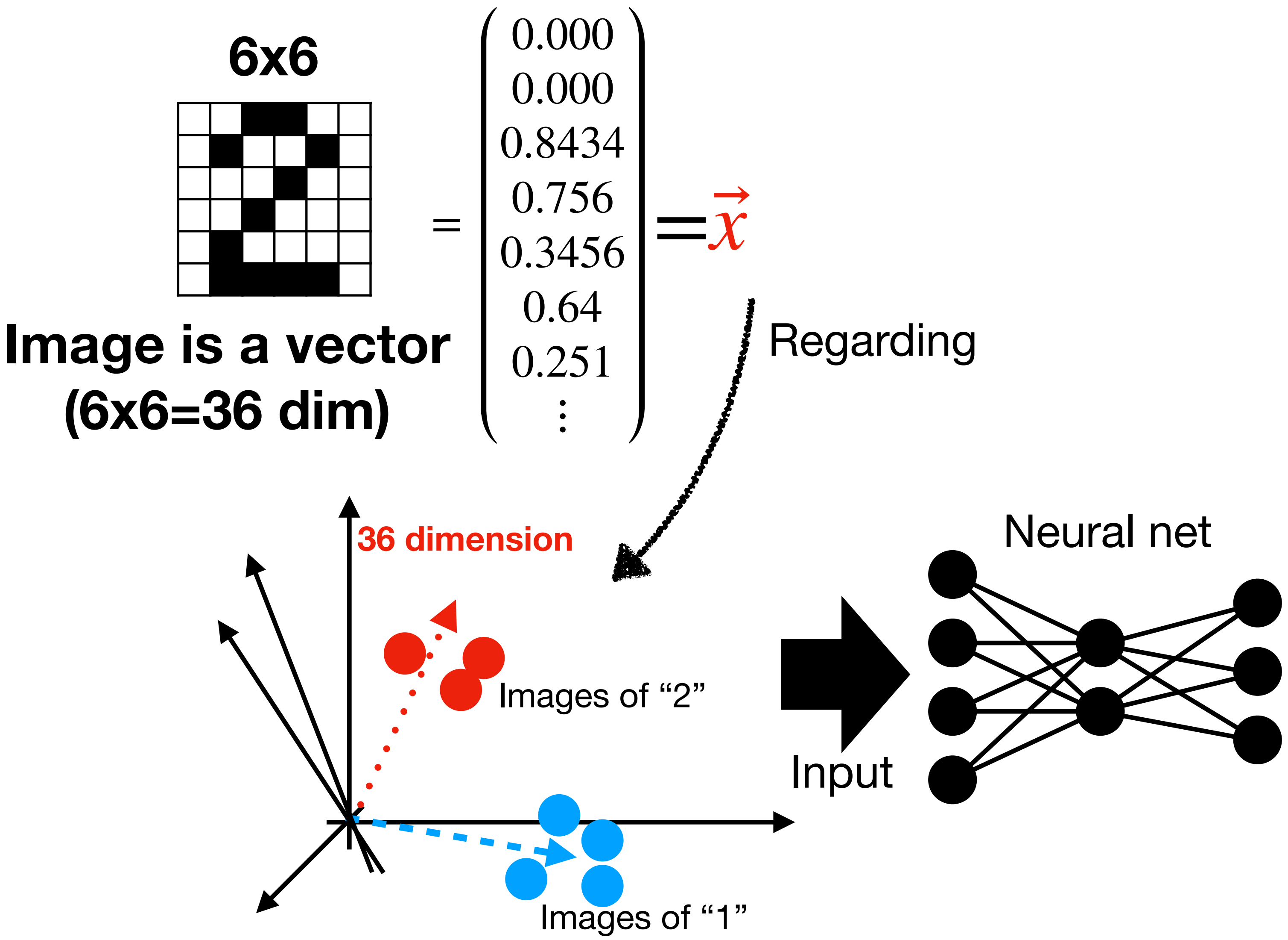
Image recognition = Find a map between two vector spaces

What is the neural networks?

Akio Tomiya

Neural network is a *universal* approximation function

Example: Recognition of hand-written numbers (0-9)



What is the neural networks?

Akio Tomiya

Affine transformation + element-wise transformation

Layers of neural nets $l = 2,3,\cdots, L,\ \vec{u}^{(1)} = \vec{x}$

W^{l}, $\vec{b}^{(l)}$ are fit parameters

$$\begin{cases} \vec{z}^{(l)} = W^{(l)}\vec{u}^{(l-1)} + \vec{b}^{(l)} \\ u_i^{(l)} = \sigma^{(l)}(z_i^{(l)}) \end{cases}$$

Affine transformation
(b=0 called linear transformation)

Element-wise (local) non-linear.
hyperbolic tangent-ish function

A fully connected neural net = composite function (Linear&non-linear)

$$f_\theta(\vec{x}) = \sigma^{(3)}(W^{(3)}\sigma^{(2)}(W^{(2)}\vec{x} + \vec{b}^{(2)}) + \vec{b}^{(3)})$$

θ is a set of parameters: $w_{ij}^{(l)}, b_i^{(l)}, \cdots$

- Input = vectors, output = vectors
- Neural net = a nested function with a lot of parameters (W, b)
- Parameters (W, b) are determined from data (fitting/training)

Neural network = map between vectors and vectors

Physicists terminology: Variational ansatz

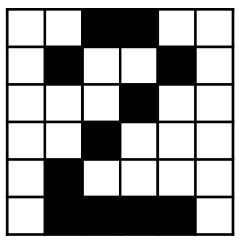
What is the neural networks?

Akio Tomiya

Neural network is a *universal* approximation function

Example: Recognition of hand-written numbers (0-9)

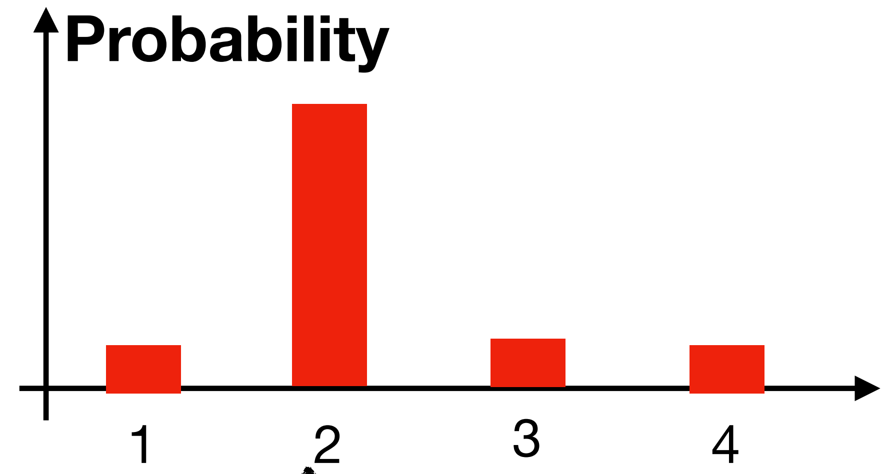
6x6



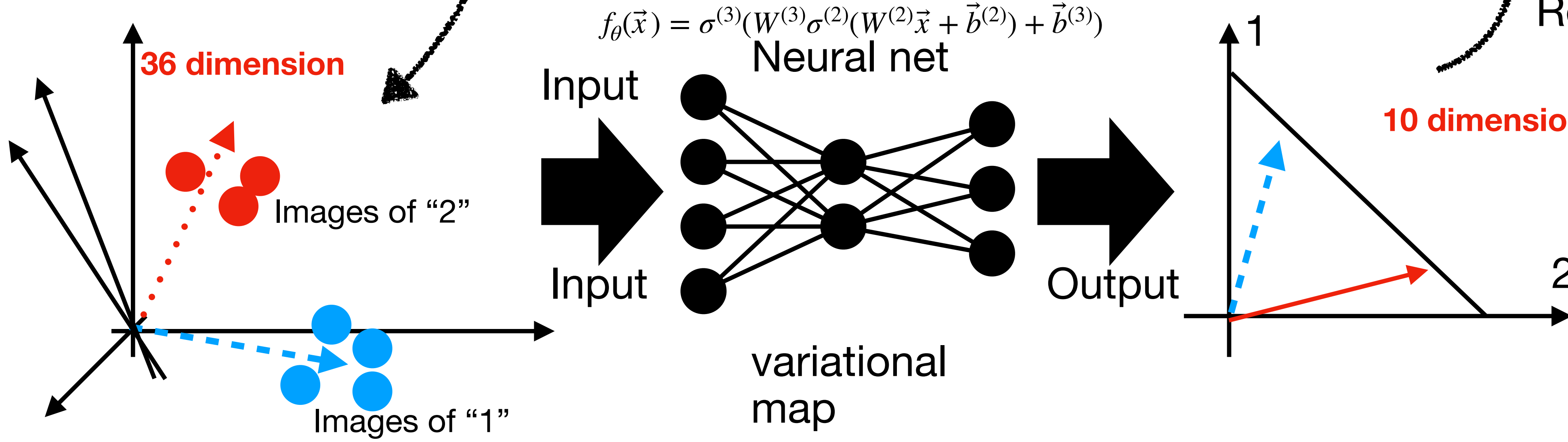
$$= \begin{pmatrix} 0.000 \\ 0.000 \\ 0.8434 \\ 0.756 \\ 0.3456 \\ 0.64 \\ 0.251 \\ \vdots \end{pmatrix} = \vec{x}$$

Image is a vector (6x6=36 dim)

Regarding



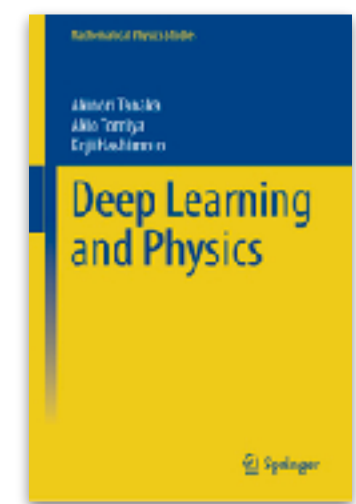
Regard



“0” = (1,0,0,…)
“1” = (0,1,0,…)
“2” = (0,0,1,…)
…
“9” = (0,0,…,1)

Fact: Neural network can mimic any function
= A systematic variational function.

In this example, NN mimics image (36-dim vector) and label (10-dim vector)

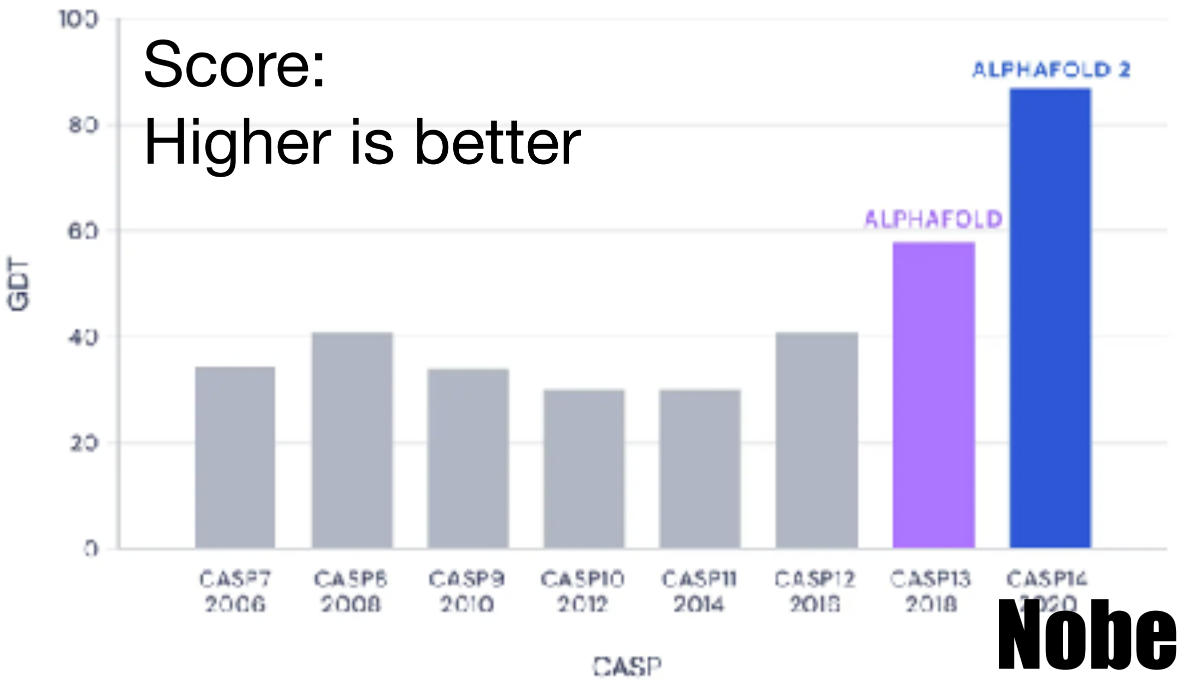


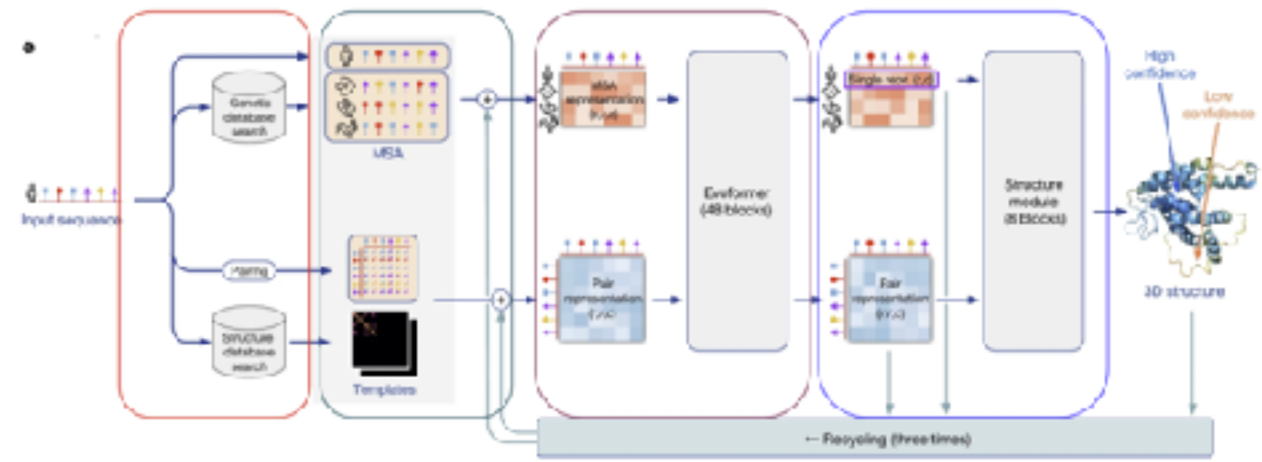
Machine learning

Akio Tomiya

Neural network have done good job

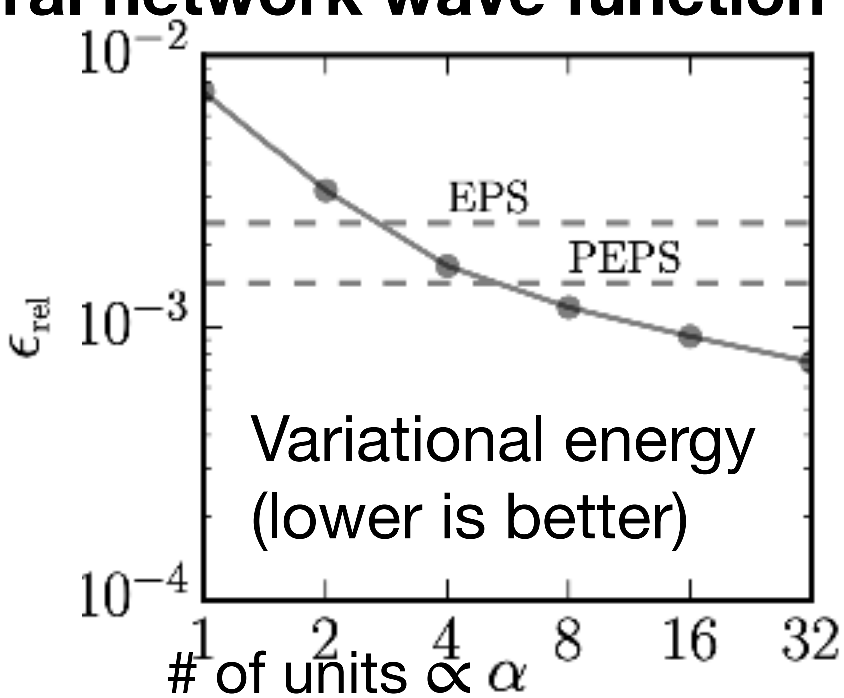
Protein Folding (AlphaFold, John Jumper+, Nature, 2020+), Transformer neural net

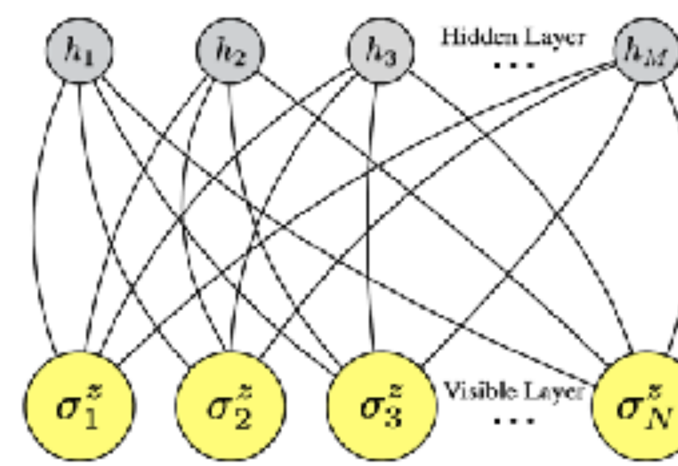




Nobel prize 2024!

Neural network wave function for many body (Carleo Troyer, Science 355, 602 (2017))





Neural net is very useful for science!

Akio Tomiya

Machine learning

Type 1, fully connected neural networks

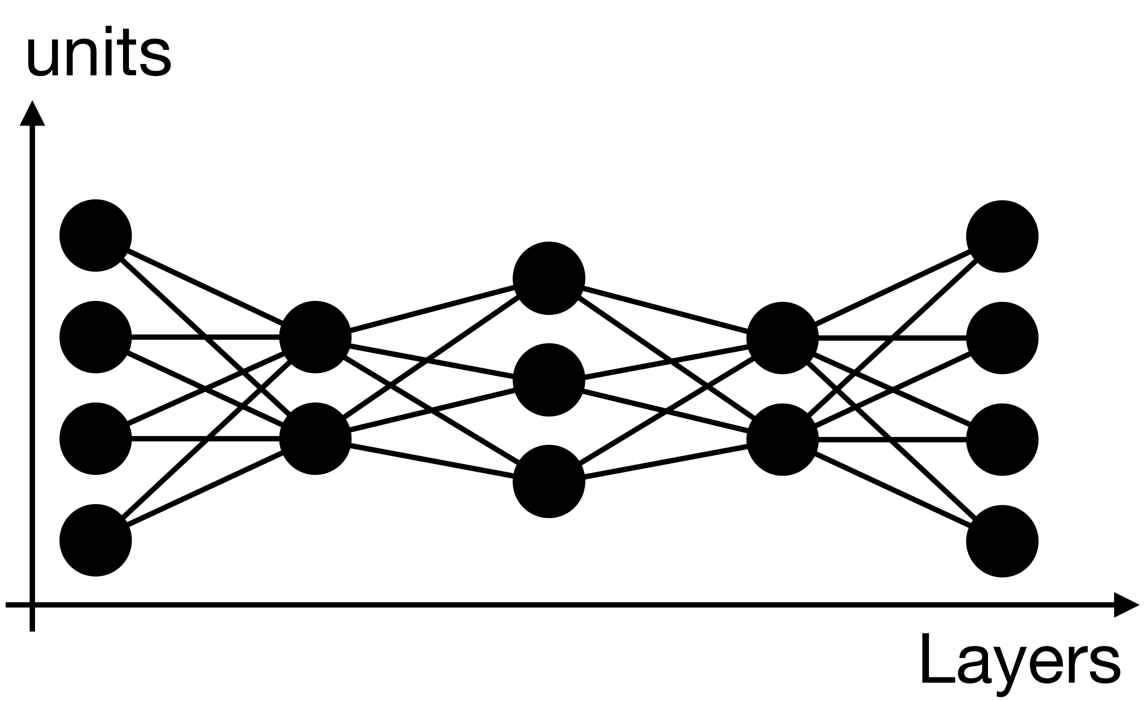
Layers of neural nets $l = 2,3,\cdots, L,\ \vec{u}^{(1)} = \vec{x}$

$W^l, \vec{b}^{(l)}$ are fit parameters

$$\begin{cases} \vec{z}^{(l)} = W^{(l)}\vec{u}^{(l-1)} + \vec{b}^{(l)} \\ u_i^{(l)} = \sigma^{(l)}(z_i^{(l)}) \end{cases}$$

Affine transformation
(b=0 called linear transformation)

Element-wise (local) non-linear.
hyperbolic tangent-ish function



Pros
- Easy to implement
- Good for first trial

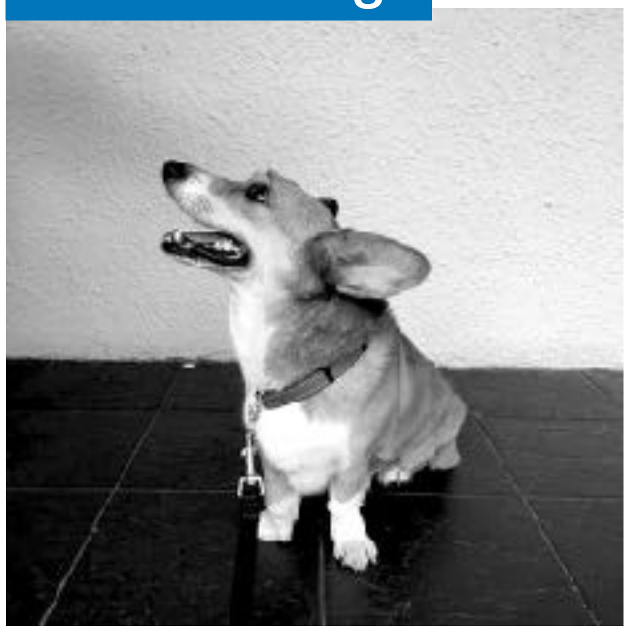
Cons
- Not efficient/performant

Configuration generation in LQCD

Akio Tomiya

Type 2, convolutional neural networks

Filter on image



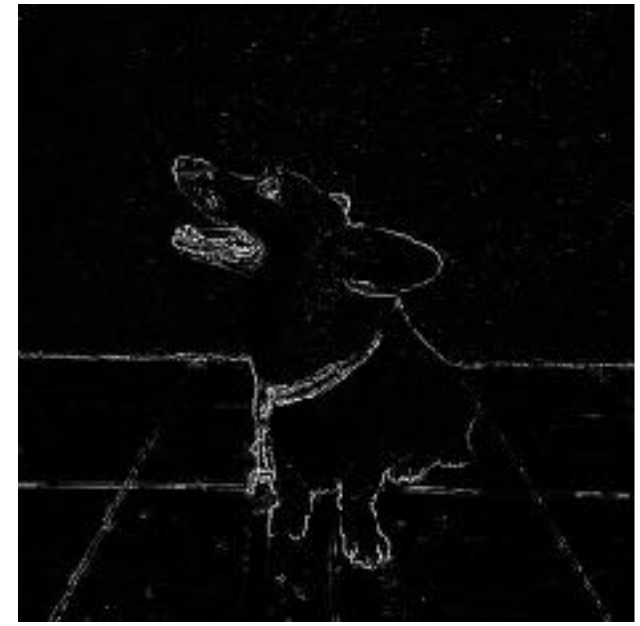
$*$

Laplacian filter

0	1	0
1	-2	1
0	1	0

(Discretization of ∂^2)

$=$



Edge detection

If input is shifted, output is shifted= respects transnational symmetry

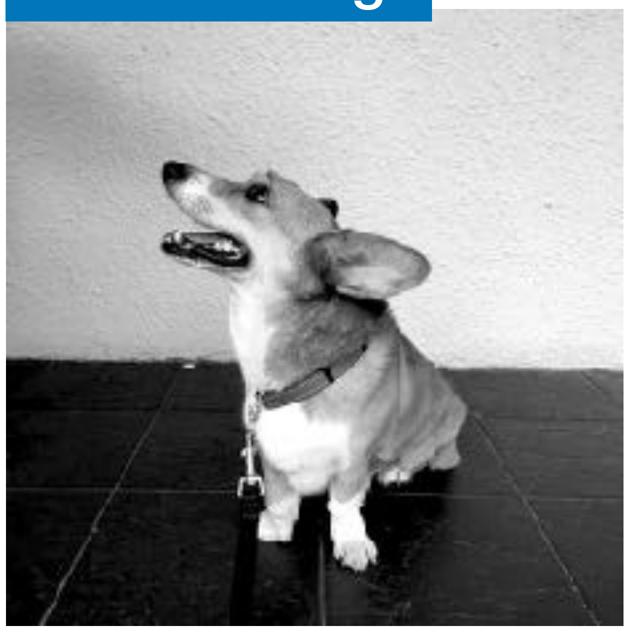
"Equivariant " = Operation and filtering is commutable

Akio Tomiya

Configuration generation in LQCD

Convolution layer = trainable filter

Filter on image



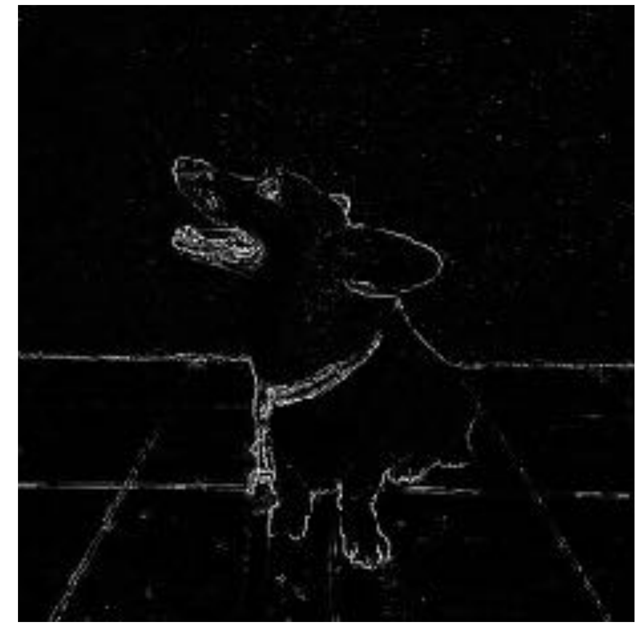
$*$

Laplacian filter

0	1	0
1	-2	1
0	1	0

(Discretization of ∂^2)

=



Edge detection

If input is shifted, output is shifted= respects transnational symmetry

"Equivariant " = Operation and filtering is commutable

Convolution layer

$*$

Trainable filter

W_{11}	W_{12}	W_{13}
W_{21}	W_{22}	W_{23}
W_{31}	W_{32}	W_{33}

=

Edge detection

Smoothing
(Gaussian filter)

...

Fukushima, Kunihiko (1980)
Zhang, Wei (1988) + a lot!

Gaussian filter

$\frac{1}{16}$

1	2	1
2	4	2
1	2	1

(Training and data determines what kind of filter is realized)
Extract features

Convolutional NN (layers) respects transnational symmetry!

Equivariance and convolution

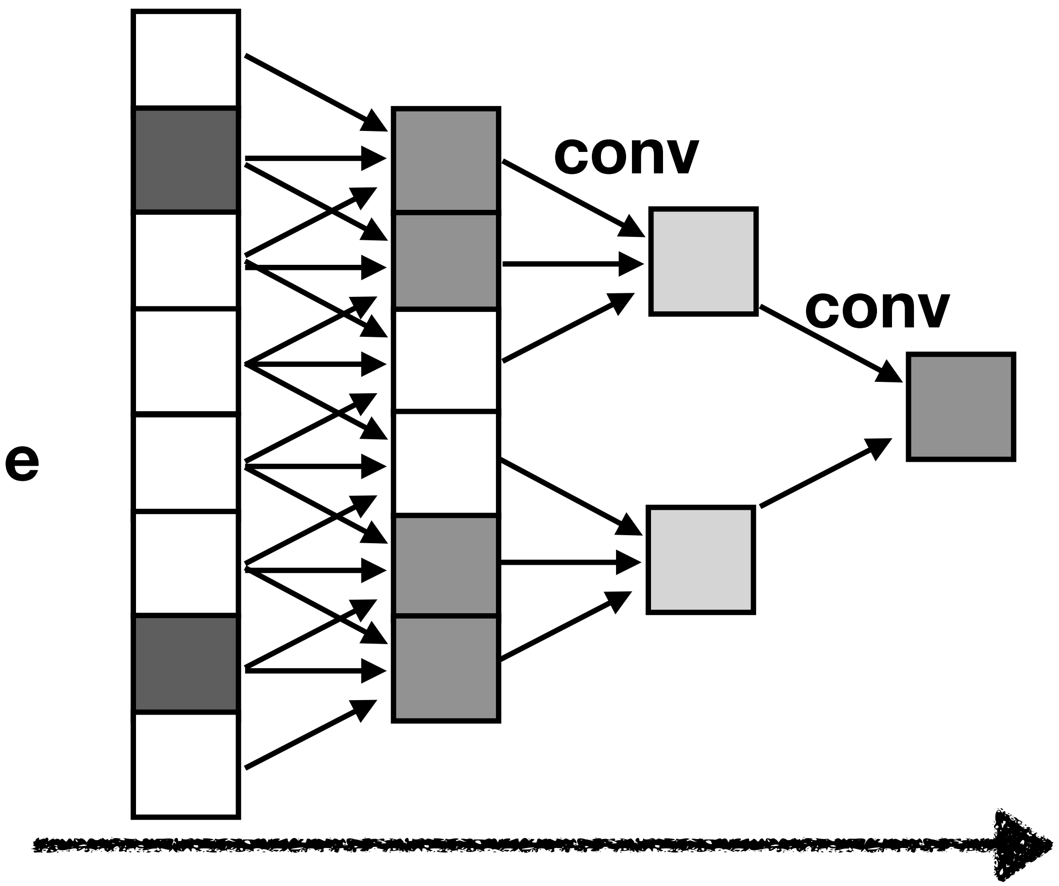
Akio Tomiya

Convolutional Neural network have been good job but local

2106.04554

conv ~ neural net with n-th nearest neighbor connections (local)

e.g.
1d Input image



Long range correlation in input is captured by deep layers since operation is local

However, 1 step of convolutional layer can pick up only local correlation and representability of neural networks is limited. Global correlations are sometimes important.
How can we overcome these difficulties?

Configuration generation in LQCD

Akio Tomiya

Attention layer used in Transformers (GPT etc)

arXiv:1706.03762

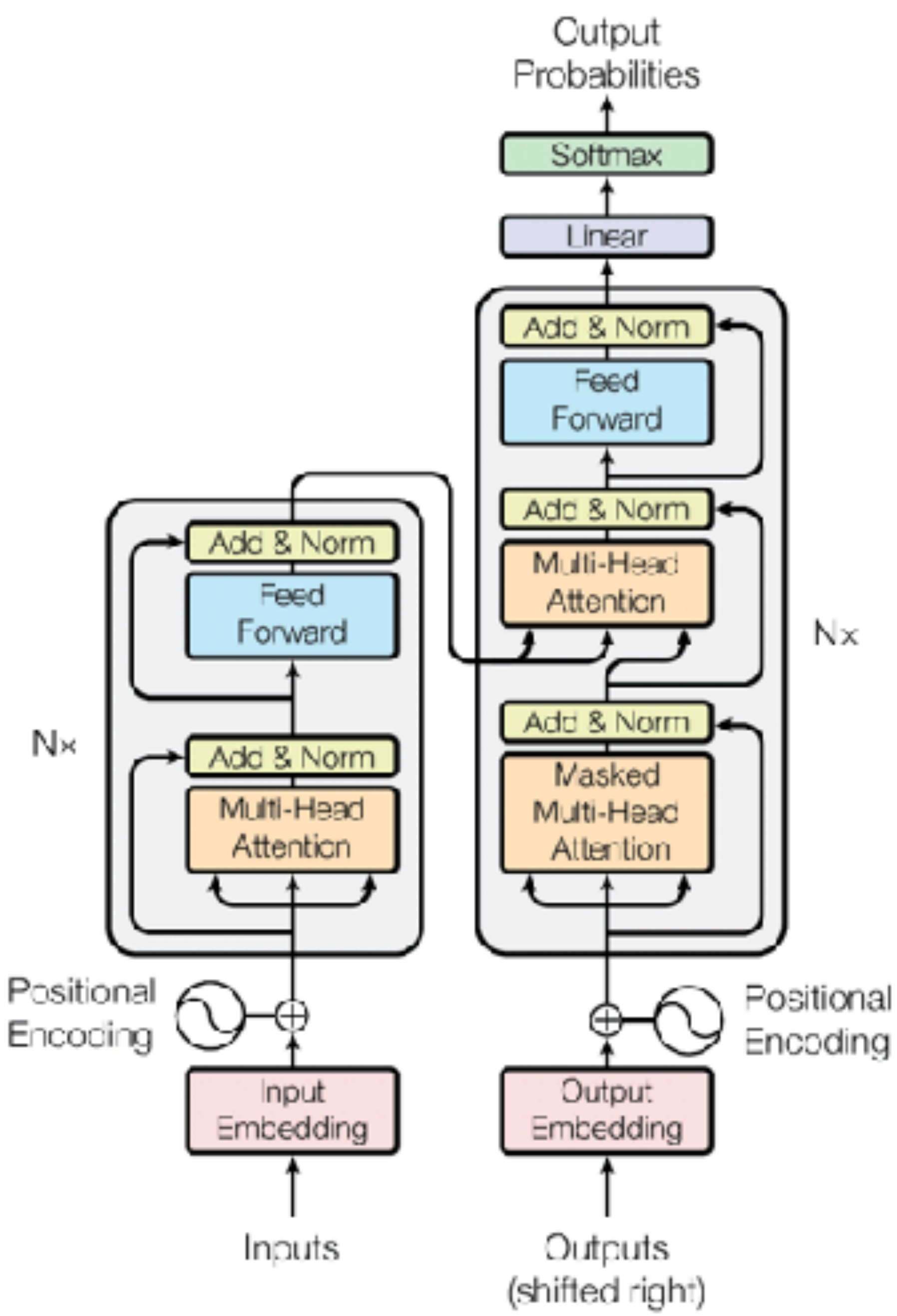


Figure 1: The Transformer - model architecture.

OpenAI ChatGPT

Gemini

DeepL

Attention layer (in transformer model) has been introduced in a paper titled **"Attention is all you need"** (1706.03762)
State of the art architecture of language processing.
Attention layer is essential.

Configuration generation in LQCD

Akio Tomiya

Attention layer can capture non-local correlations

arXiv:1706.03762

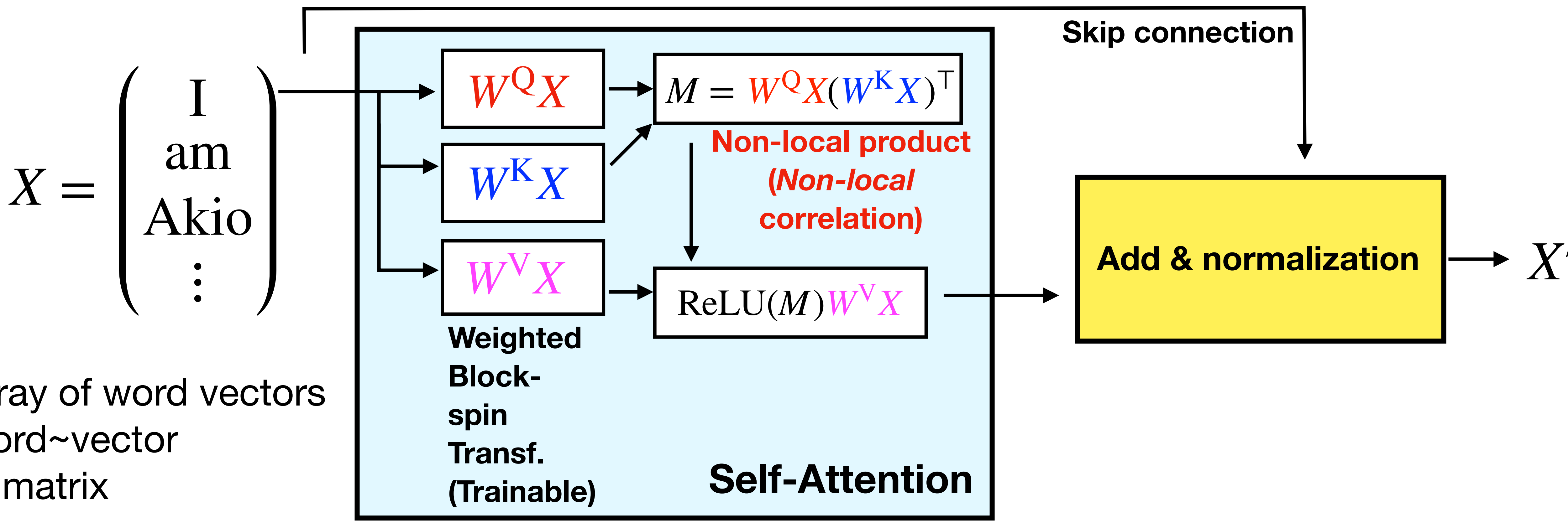
Modifier in language can be non-local

Eg. I am Akio Tomiya living in Japan, who studies machine learning and physics

In physics terminology, this is **non local correlation.**

The attention layer enables us to treat non-local correlation with a neural net!

Simplified version of Attention/Transformer



Array of word vectors
Word~vector
X: matrix

These can be repeated

Configuration generation in LQCD

Akio Tomiya

Transformer shows scaling lows (power law)

arXiv: 2001.08361

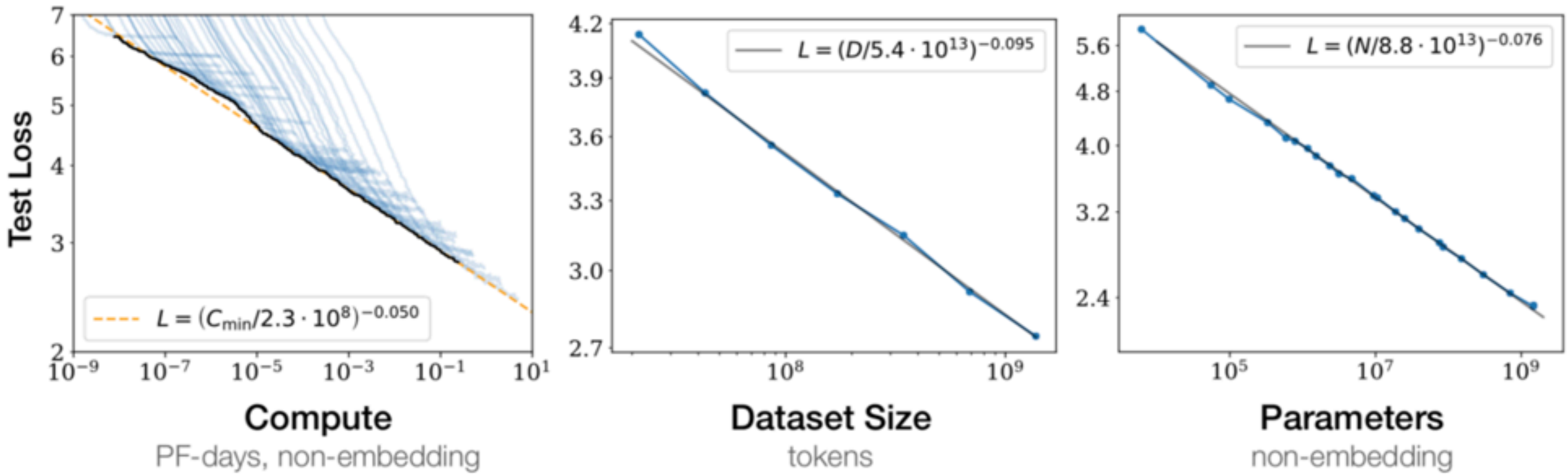


Figure 1 Language modeling performance improves smoothly as we increase the model size, datasetset size, and amount of compute[2] used for training. For optimal performance all three factors must be scaled up in tandem. Empirical performance has a power-law relationship with each individual factor when not bottlenecked by the other two.

- **Transformers requires huge data (e.g. GPT uses all electric books in the world) Because it has few inductive bias (no equivariance)**
- **It can be improved systematically**

Configuration generation in LQCD

Akio Tomiya

Generative models

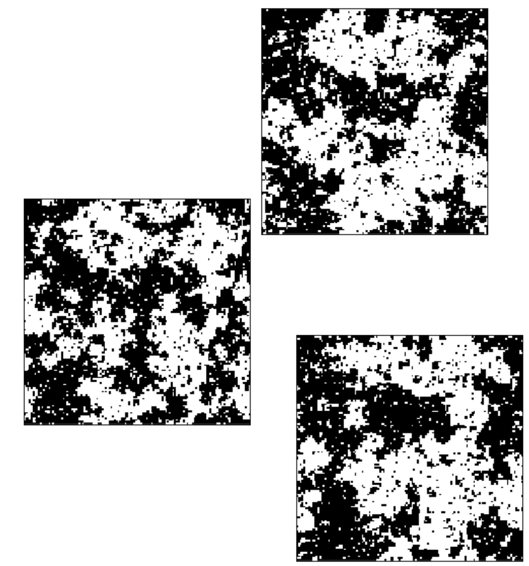
Extract/ modeling

Probability distribution of configurations

$$P[s] = \frac{1}{Z} e^{-H[s]}$$

Written in fields/local variables

Sampling



WIKIPEDIA
The Free Encyclopedia

Extract/ modeling

Probability distribution of images

$$P[s] = ?$$

Written in Neural net

Sampling

OpenAI
ChatGPT

Machine learning

Akio Tomiya

Surrogate model = NN approximated solution/Trajectory

An "effective model" approach for numerical solutions

$$u_{\mathrm{NN}}(x;\theta) \approx u(x)$$

Neural network surrogate models can mimic the effective dynamics: instead of solving PDE/ODE numerically, NN gives an approximate trajectory or solution with reduced computational cost.

Machine learning

Akio Tomiya

Physics informed neural net (PINN),

The physical law (PDE operator F) and boundary conditions (BCs) are encoded directly into the loss function.
Training forces the NN to satisfy physics, not only data.

$$L = \sum_{i=1}^{N_{\text{data}}} \left| u_{\text{NN}}\left(x_i; \theta\right) - u_i^{\text{data}} \right|^2 + \lambda \sum_{j=1}^{N_{\text{phys}}} \left| \mathscr{F}\left[u_{\text{NN}}\left(x_j; \theta \right) \right] \right|^2$$

F=0 is the physics law

Machine learning
Equivariance

Akio Tomiya

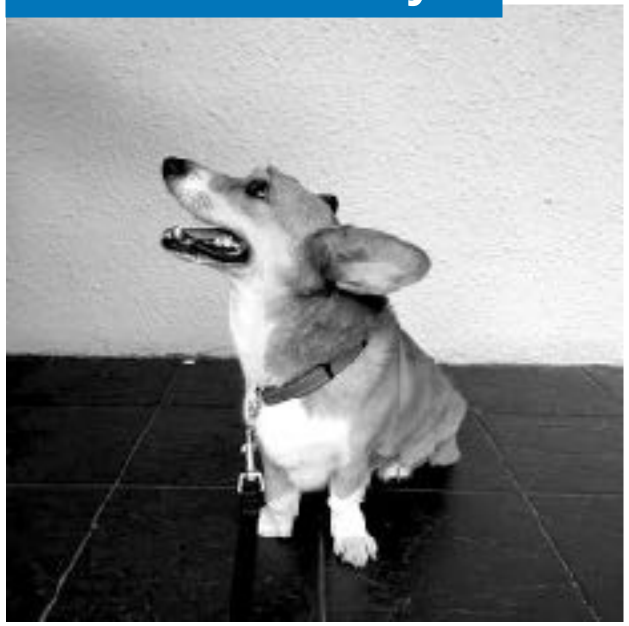
Let $f_\theta(\vec{x})$ be a neural net, (θ is a set of parameters) which takes a value in the same domain of $\vec{x}$.

If a transformation $\vec{x} \to T\vec{x}$ acts as

$$f_\theta(T\vec{x}) \to Tf_\theta(\vec{x}),$$

this neural net is *equivariant*.
Same concept of covariant in particle physics.
Commutativity with f_θ and T

Convolution layer



$*$

Trainable filter

w_{11}	w_{12}	w_{13}
w_{21}	w_{22}	w_{23}
w_{31}	w_{32}	w_{33}

$=$

Edge detection

Smoothing
(Gaussian filter)

...

(Training and data determines what kind of filter is realized)
Extract features

Fukushima, Kunihiko (1980)
Zhang, Wei (1988) + a lot!

Gaussian filter

$$\frac{1}{16}\begin{pmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{pmatrix}$$

Configuration generation in LQCD

Akio Tomiya

Smearing = Smoothing of gauge fields

Eg.

Coarse image

Gaussian filter

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 1 \\ 1 & 2 & 1 \end{bmatrix}$$

Smoothened image

We want to smoothen *gauge* field configurations with keeping *gauge* symmetry

Two types:

APE-type smearing

Stout-type smearing

M. Albanese+ 1987
R. Hoffmann+ 2007
C. Morningster+ 2003

Smearing

Smoothing with gauge symmetry, APE type

APE-type smearing

M. Albanese+ 1987
R. Hoffmann+ 2007

$$U_\mu(n) \to U_\mu^{\rm fat}(n) = \mathcal{N}\left[(1-\alpha)U_\mu(n) + \frac{\alpha}{6}V_\mu^\dagger[U](n)\right]$$

Normalization

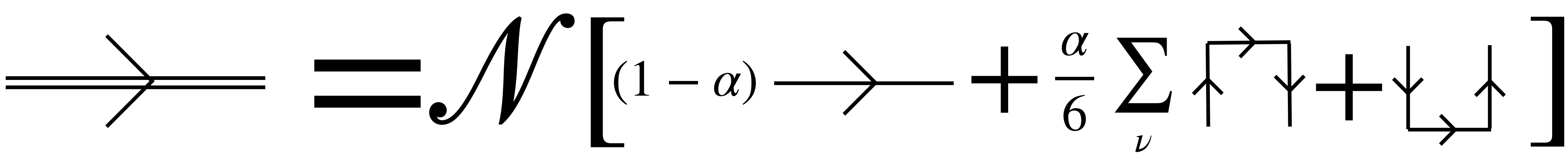
$$\mathcal{N}[M] = \frac{M}{\sqrt{M^\dagger M}}$$

Or projection

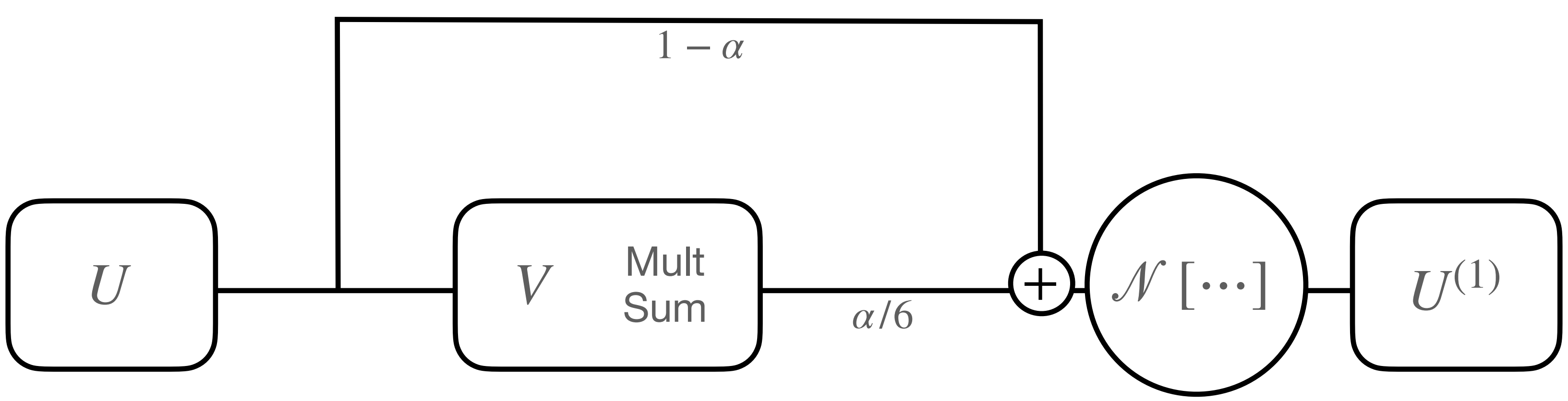
$$V_\mu^\dagger[U](n) = \sum_{\mu\neq\nu} U_\nu(n)U_\mu(n+\hat{\nu})U_\nu^\dagger(n+\hat{\mu}) + \cdots$$

$V_\mu^\dagger[U](n)$& $U_\mu(n)$ shows same transformation
→$U_\mu^{\rm fat}[U](n)$ is as well

Schematically,



In the calculation graph,



This smoothing is commutable with gauge transformation

Configuration generation in LQCD

Akio Tomiya

Smearing ∼ neural network with fixed parameter!

AT Y. Nagai arXiv: 2103.11965

General form of smearing (~smoothing, averaging in space)

$$\begin{cases} z_\mu(n) = w_1 U_\mu(n) + w_2 \mathcal{G}[U] \\ U_\mu^{\mathrm{fat}}(n) = \mathcal{N}(z_\mu(n)) \end{cases}$$

Summation with gauge sym

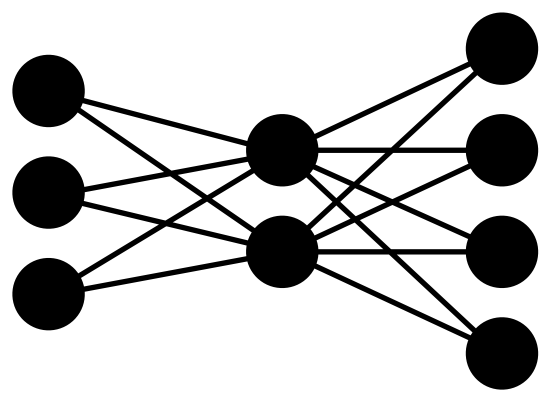
A local function
(Projecting on the gauge group)

(Index i in the neural net corresponds to n & μ in smearing. Information processing with NN is evolution of scalar field)

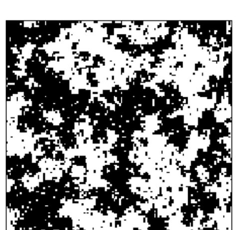
Multi-level smearing = Deep learning (with given parameters)

As same as the convolution, we can train weights.

Image

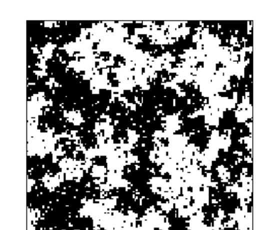


Image



Configuration

$$z_\mu(n) = w_1 U_\mu(n) + w_2 \mathcal{G}[U]$$



Configuration

Self-learning Monte-Carlo

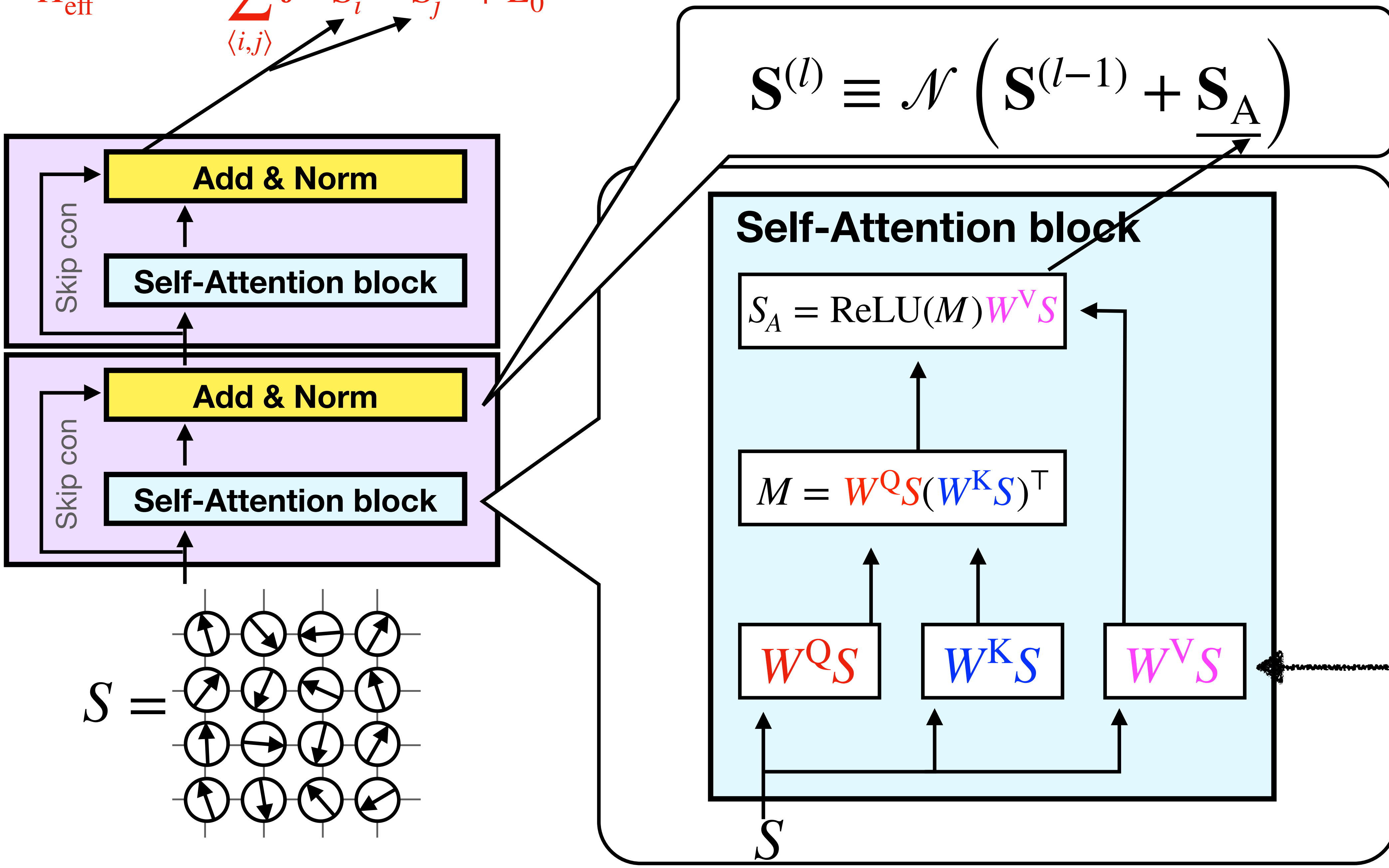
Equivariant Attention layer

Akio Tomiya

arXiv: 2306.11527.

We can construct effective hamiltonian with output of Attention layer because "output of Attention = smeared fields with non-local correlation"

$$H_{\rm eff}^{\rm Linear} = -\sum_{\langle i,j \rangle} J^{\rm eff} \mathbf{S}_i^{\rm eff} \cdot \mathbf{S}_j^{\rm eff} + E_0$$



Smeared fields
Rot. equivariant
Trsl. equivariant
trainable!

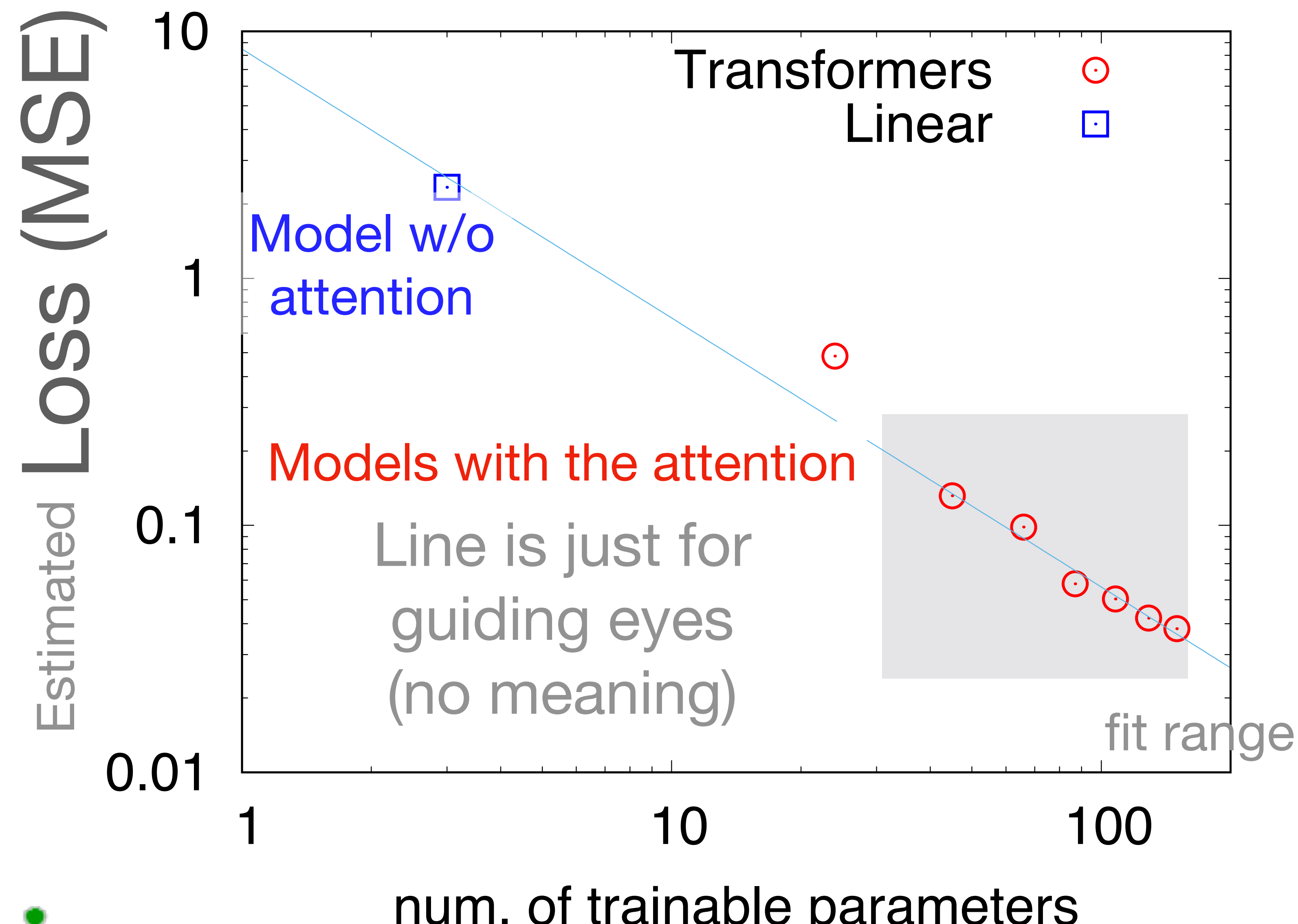
Smearing
Rot. equivariant
Trsl. equivariant
trainable!

CASK: Covariant Transformer

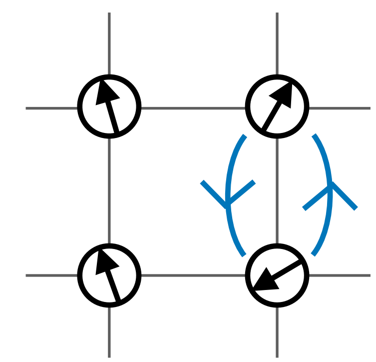
Previous work for a spin system

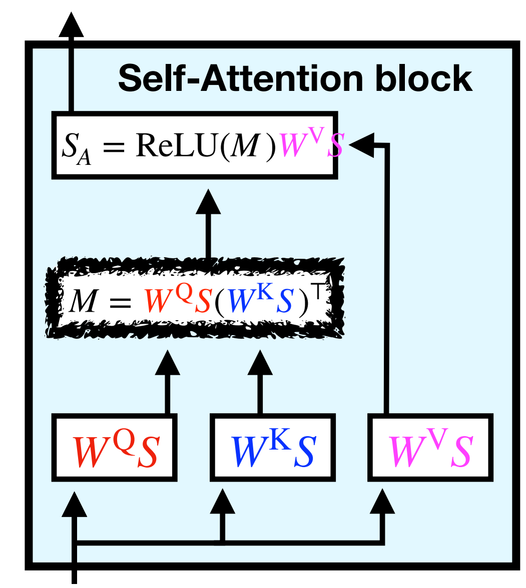
arXiv: 2306.11527 + update

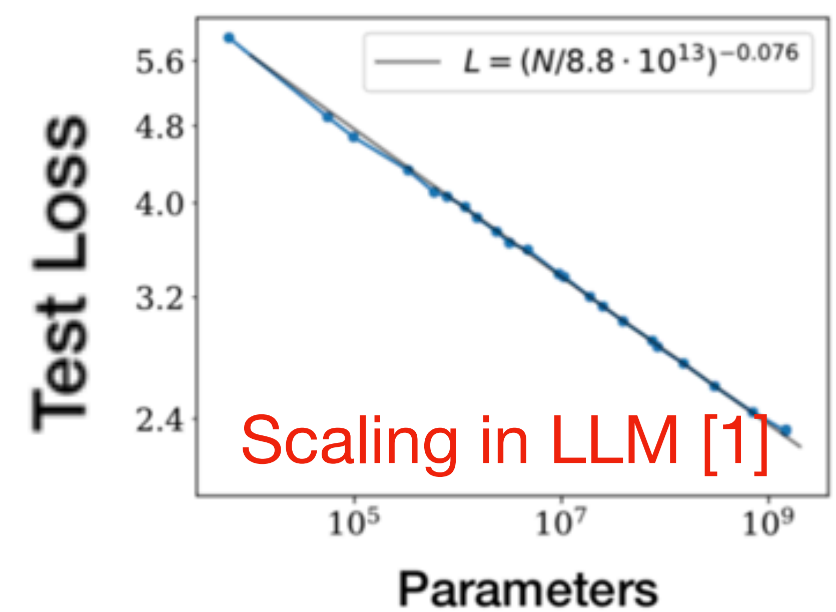
We simulate a classical spin-electron system in 2d (~ Kondo system) as a toy model. It is *not* gauge theory but good for a testbed. We utilize self-learning MC with a *covariant transformer* as a surrogate [AT, Y Nagai 2024]. We see that **scaling law**! How about lattice QCD?



(1 layer ~ 30 parameters)







Scaling in LLM [1]

CASK: Covariant Transformer

Lattice Gauge covariant attention

Akio Tomiya

arXiv: 2501.16955

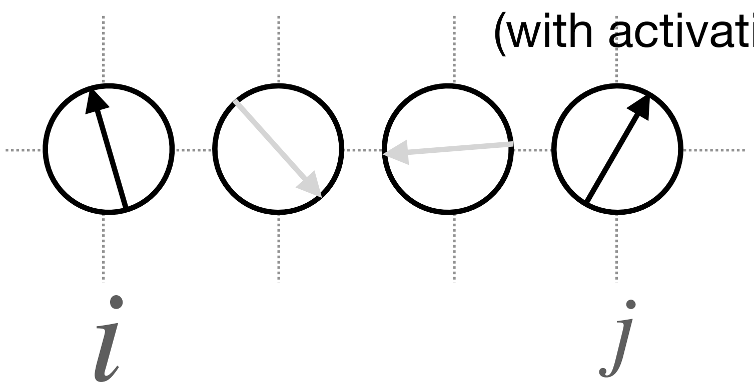
Attention matrix in transformer ~ correlation function (with block-spin transformed spin)

-> we replace it with “correlation function of links” in a **covariant** way

Attention for Kondo spins

$$a_{ij} \sim \vec{S}_i \cdot \vec{S}_j$$

(with activation)



invariant
under global O(3)

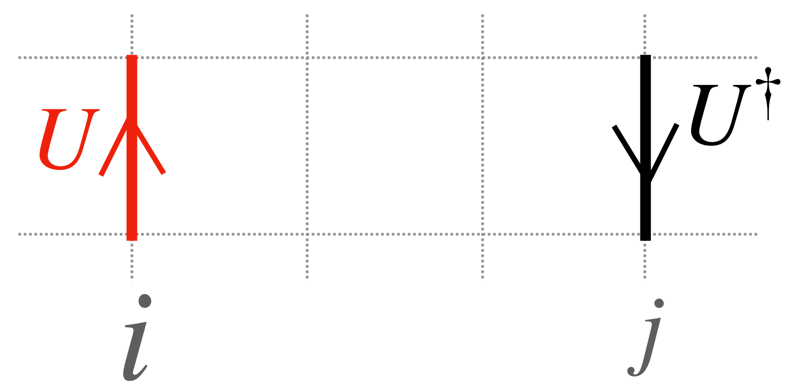
$$a_{ij} \sim (R\vec{S})_i^\top R\vec{S}_j = \vec{S}_i^\top \vec{S}_j$$

In total, output is covariant

2310.13222 AT+,2306.11527 AT+

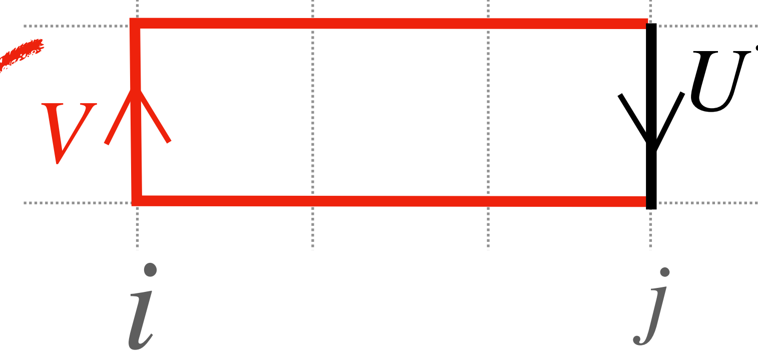
Gauging

$$a_{i\mu,j\mu} \sim \mathrm{Re\,tr}\, U_\mu(i) U_\mu^\dagger(j)$$



not invariant
(cannot be used)

Lattice gauge covariant attention



invariant under
local SU(N)

$$a_{i\mu,j\mu} \sim \mathrm{Re\,tr}\, V_\mu(i) U_\mu^\dagger(j)$$

(with activation)

In total, output is covariant

CASK: Covariant Transformer

CASK for SU(2), SU(3) gauge theory + fermions

Akio Tomiya

JuliaQCD

arXiv: 2501.16955

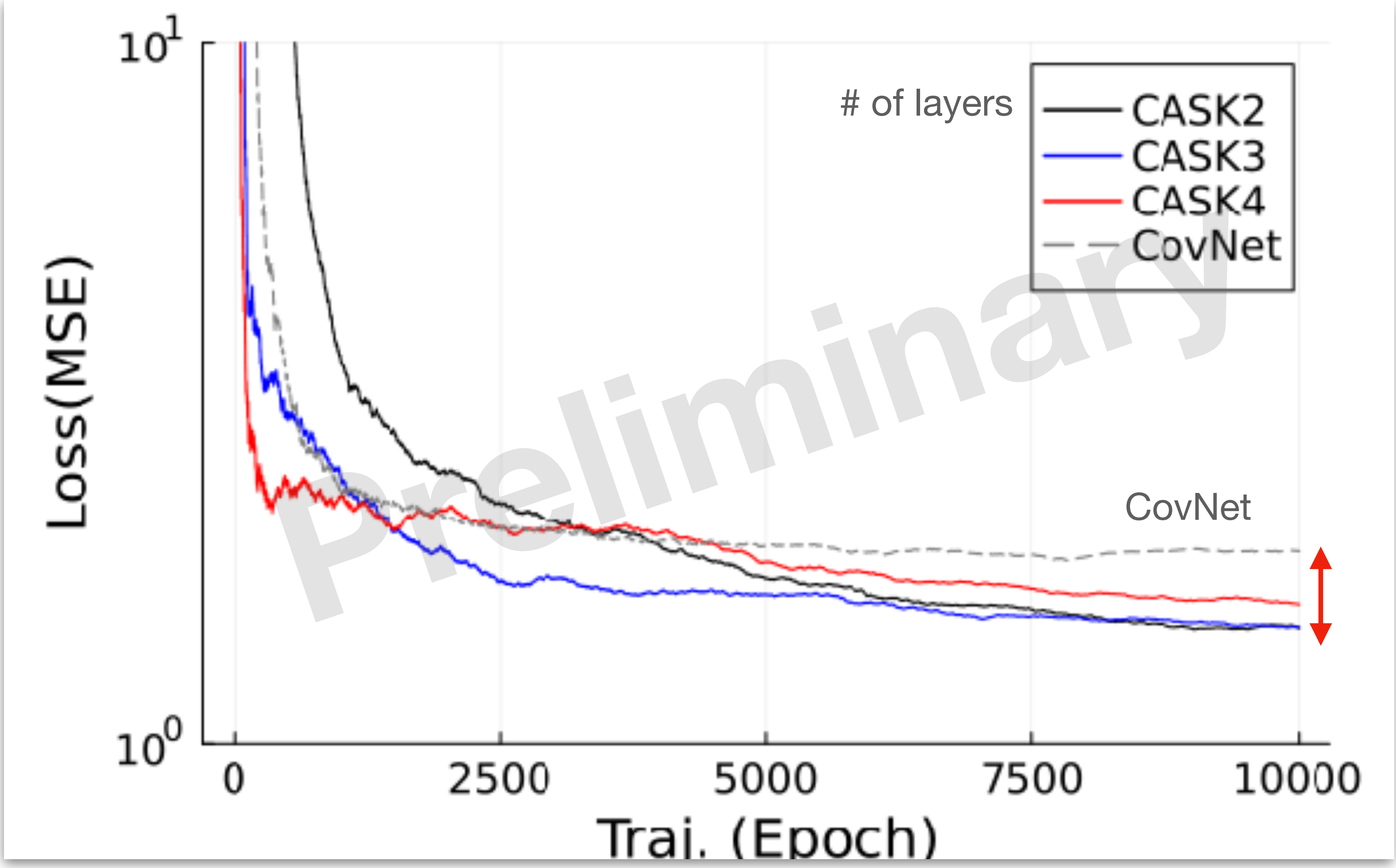
Comparison Covariant convolution (CovNet) and Covariant transformer (CASK)

$$U^{(\mathrm{NN-out})} = \begin{cases} U_\mu^{(\mathrm{CovNet})} = g_\theta^{(\mathrm{CovNet})} U_\mu^{(\mathrm{in})} & \text{CovNet (convolution based, baseline)} \\ U_\mu^{(\mathrm{CASK})} = g_\theta^{(\mathrm{CASK})} U_\mu^{(\mathrm{in})} & \text{CASK (transformer based)} \end{cases}$$

$$\mathrm{Loss} = \sum_{\mathrm{data}} \left| S^{(\mathrm{quark})}[U^{(\mathrm{NN-out})}; m = 0.4] - S^{(\mathrm{quark})}[U; m = 0.3] \right|^2$$

$$S^{(\mathrm{quark})}[U; m] = \sum_n \phi^\dagger (D[U] + m)^{-1} \phi$$

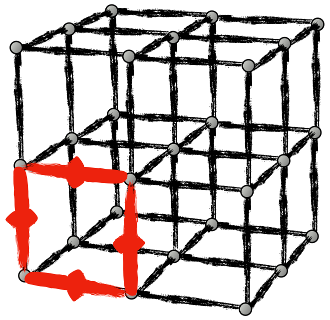
Energy function



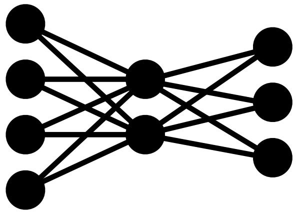
- Dynamical simulation
- $L^4 = 4^4$, SU(2), ma = 0.3
  33% larger mass in MD
- CASK has better expressibility than CovNet (Covariant CNN)
- SU(3) works as well

Outline of my talk

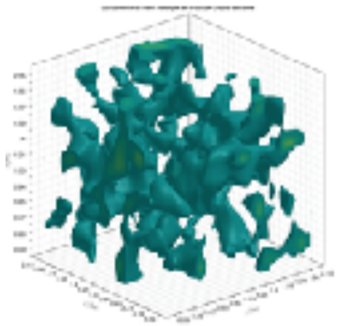
This talk is based on
JPSJ 94 (2025) 3, 031006



Lattice QCD?



Machine learning



Production of configurations

Slide

Lattice QCD?

Monte-Carlo integration is available

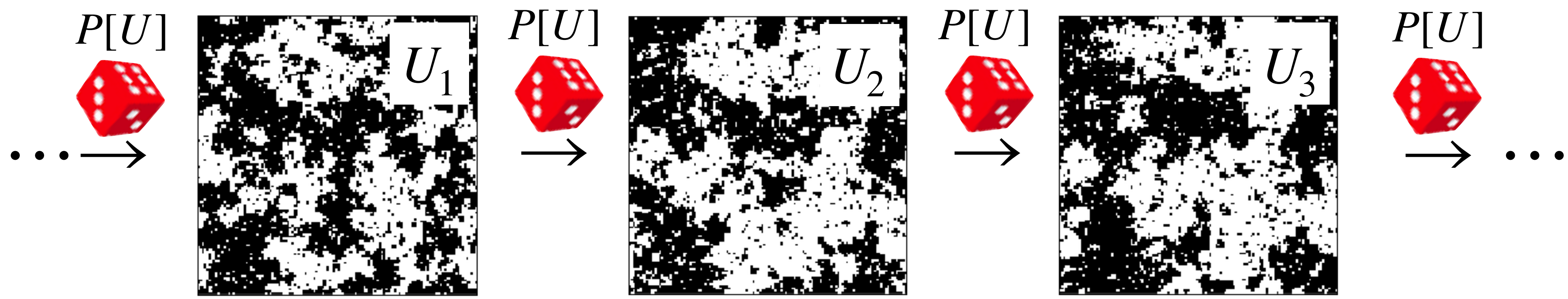
Akio Tomiya

HMC: Simon Duane, Anthony Kennedy, Brian Pendleton and Duncan Roweth1987

Quantum expectation value
$$\langle \mathcal{O} \rangle = \frac{1}{Z}\int \underbrace{DU}_{10^{11}\text{ dim. integral}} e^{-S_{\text{QCD}}[U]}\mathcal{O}(U)$$

$$S_{\text{QCD}}[U] = S_{\text{gauge}}[U] - \log\det(\not{D}[U]+m)$$

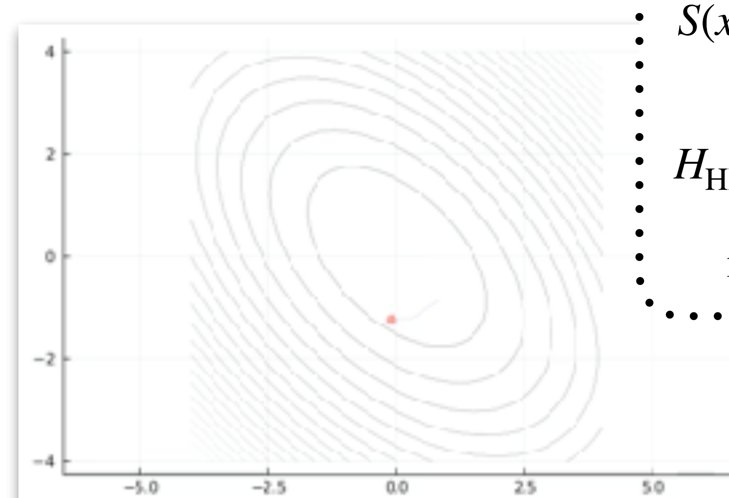
Monte-Carlo: Generate field configurations with "$P[U] \propto e^{-S_{\text{eff}}[U]}$". Stochastically estimate $\langle \mathcal{O} \rangle$




= Hybrid/Hamiltonian Monte-Carlo (HMC)
(De-facto standard Exact algorithm)

= Random momentum + EOM
Here we *regard* S_{QCD} as a potential for U

≈Molecular dynamics with random p & given U



$$S(x,y) = \frac{1}{2}(x^2+y^2+xy)$$
$$H_{\text{HMC}} = \frac{p_x^2}{2} + \frac{p_y^2}{2} + S(x,y)$$
init p_x, p_y = random

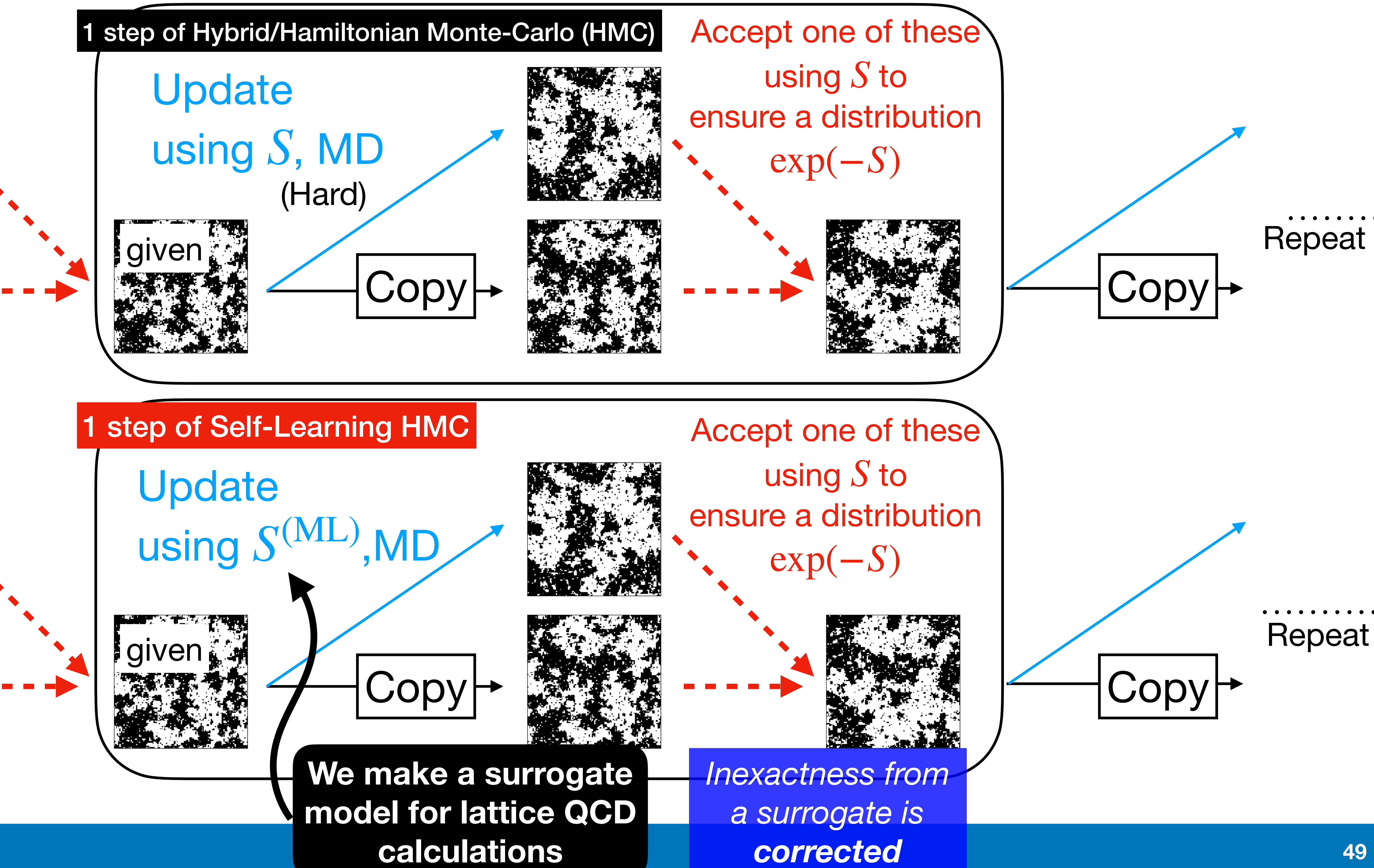
$$\langle \mathcal{O} \rangle = \frac{1}{N_{\text{sample}}}\sum_{k=1}^{N_{\text{sample}}} \mathcal{O}[U_k] \quad \left(N_{\text{sample}} \to \infty\right)$$

CASK: Covariant Transformer

Akio Tomiya

Self-learning HMC = Surrogate + Correction = Exact

arXiv: 2306.11527
and ref. therein



Self-learning MC

Linear effective action/Hamiltonian

arXiv:1610.03137+

We want expectation values with $W[\phi] \propto \exp[-\beta S[\phi]]$

using $W_{\text{eff}}[\phi] \propto \exp[-\beta S_{\text{eff}}[\phi]]$ **How can we design?**

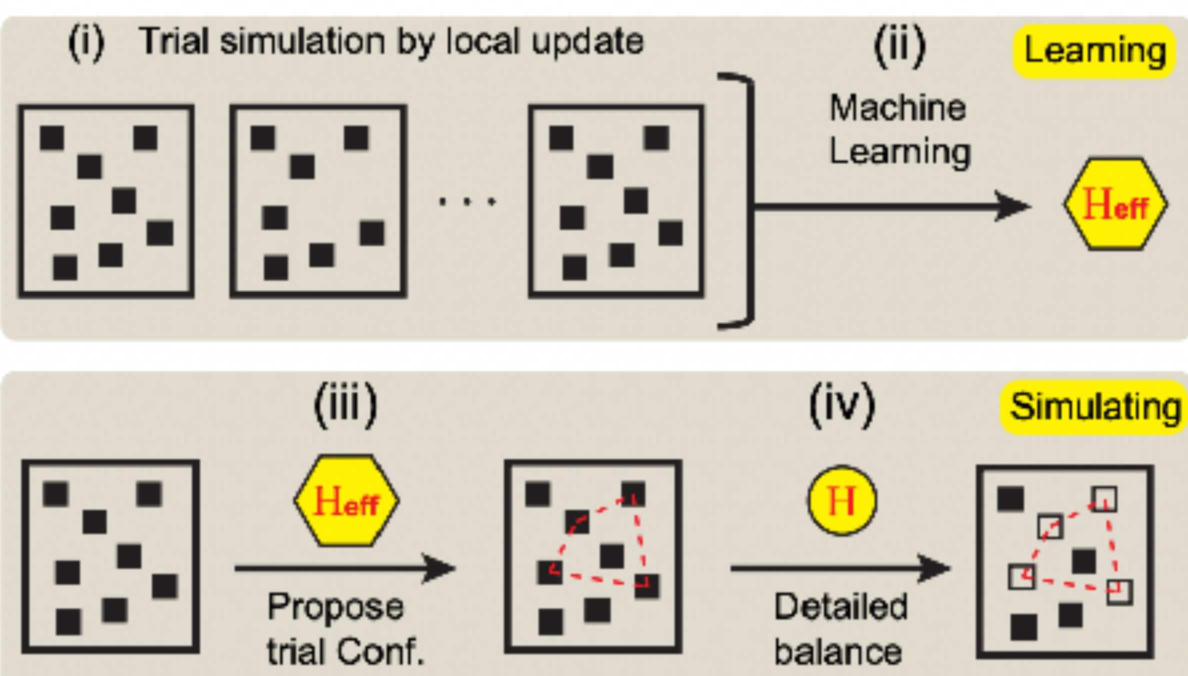
Example in the first paper, classical magnet (Ising like theory)

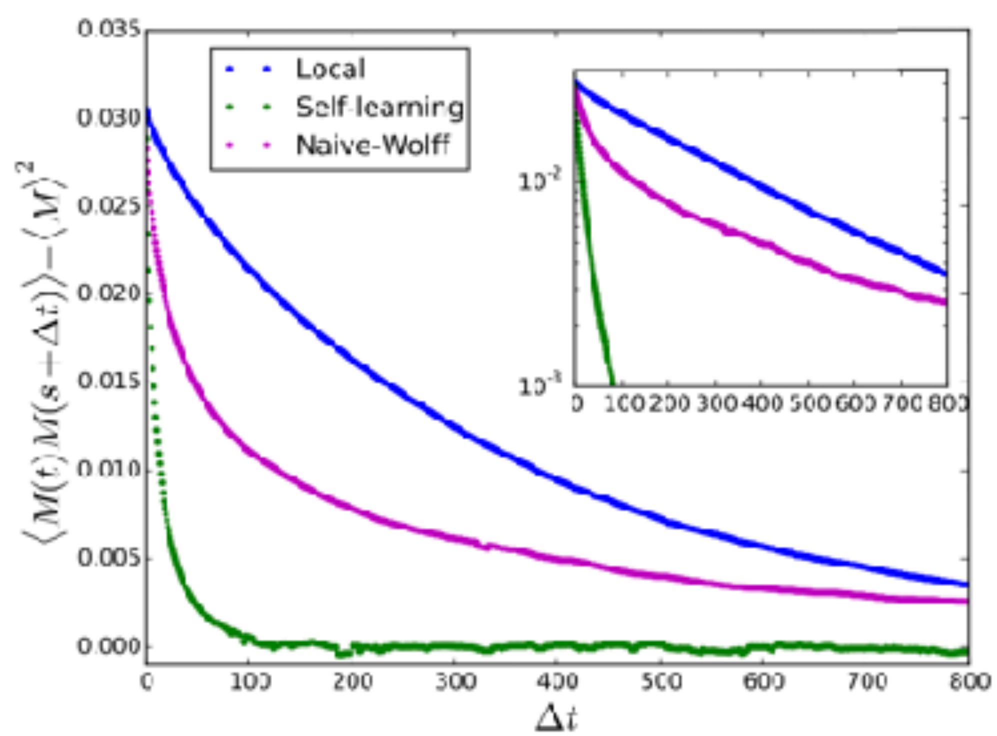
$S_i = \pm 1$ $\quad H = -J \sum_{\langle ij \rangle} S_i S_j - K \sum_{ijkl \in \square} S_i S_j S_k S_l$

2nd term prevents global update algorithm

$$H_{\text{eff}} = E_0 - \tilde{J}_1 \sum_{\langle ij \rangle_1} S_i S_j \quad \text{(linear)}$$

$E_0, \tilde{J}_1$ are determined by fit. Minimizing $(H - H_{\text{eff}})^2$

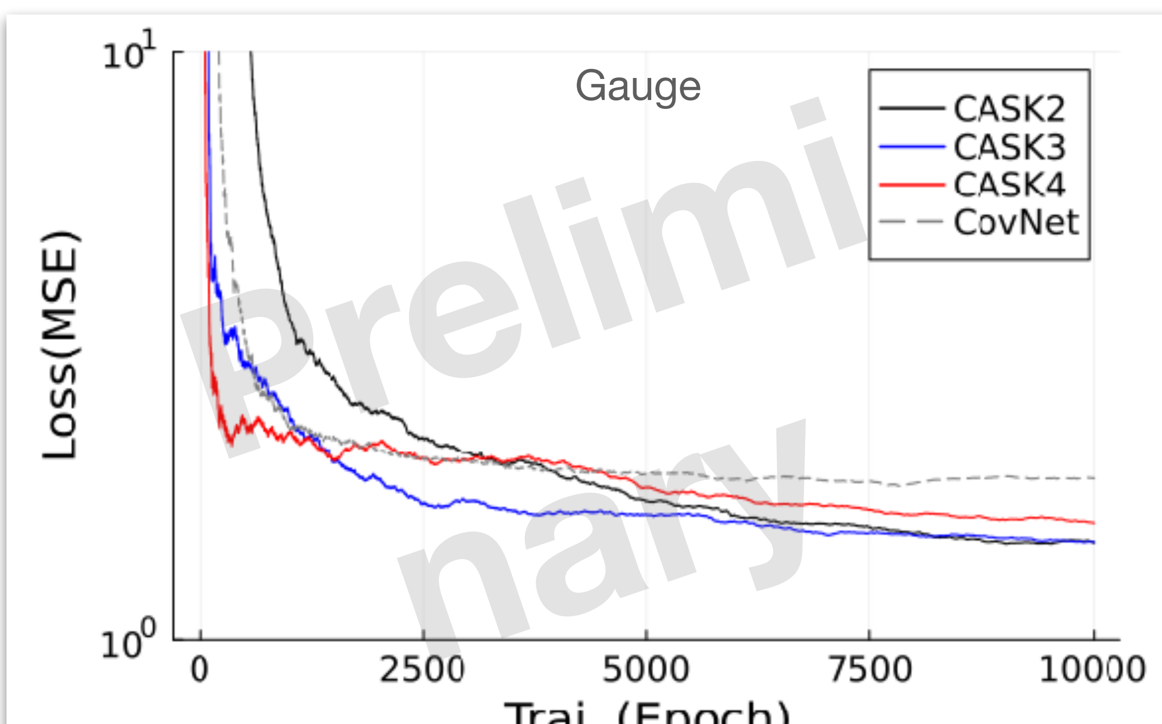


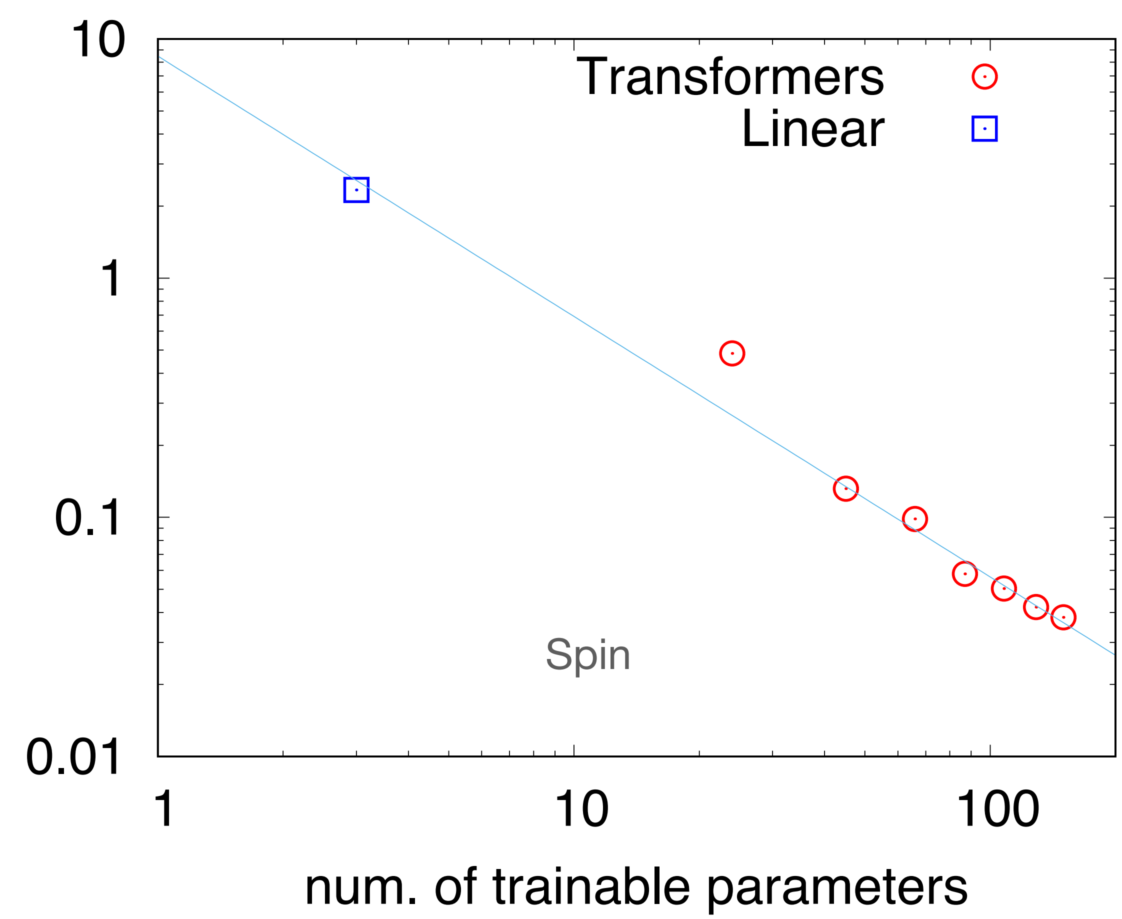


Self-learning MC
CASK & spin transformer

Akio Tomiya
JuliaQCD
arXiv: 2501.16955

Comparison Covariant convolution (CovNet) and Covariant transformer (CASK) and a spin transformer simulations have been done with SL(H)MC





Flow based sampling algorithm

Change of variables makes problem easy

$$\int D\phi e^{-S[\phi]} O[\phi] = \int Dz \underbrace{\left| \det \frac{\partial \phi}{\partial z} \right|}_{=\text{Jacobian}=J} e^{-S[\phi[z]]} O[\phi[z]]$$

$$S_{\text{eff}}[z] = S[\phi[z]] - \log J[z]$$

$$= \int Dz e^{-S_{\text{eff}}[z]} O[\phi[z]]$$

***If* this is easy** **to sample (or integrate), like flat measure/Gaussian, we are happy**

arxiv 1904.12072, 2003.06413, 2008.05456 and more.

Flow based sampling algorithm

Akio Tomiya

Viewpoint: Change of variables makes problem easy

Simplest example: Box Muller

$$\begin{cases} z = e^{-\frac{1}{2}(x^2+y^2)} \\ \tan\theta = y/x \end{cases}$$

Change of variables

$$\int_{-\infty}^{\infty} dx \int_{-\infty}^{\infty} dy\; e^{-\frac{1}{2}x^2 - \frac{1}{2}y^2} = \frac{1}{2}\int_0^{2\pi} d\theta \int_0^1 dz$$

Target integral: hard

Easy

Change of variables sometimes problem easier (this case, it makes the measure flat)

RHS is flat measure
→We can sample like right eq.
(uniform)

$$\begin{cases} \xi_1 \sim (0,2\pi) \\ \xi_2 \sim (0,1) \end{cases}$$

We can reconstruct
a field config x, y
for original theory
like right eq.

$$\begin{cases} x = r\cos\theta & \theta = \xi_1 \\ y = r\sin\theta & r = \sqrt{-2\log\xi_2} \end{cases}$$

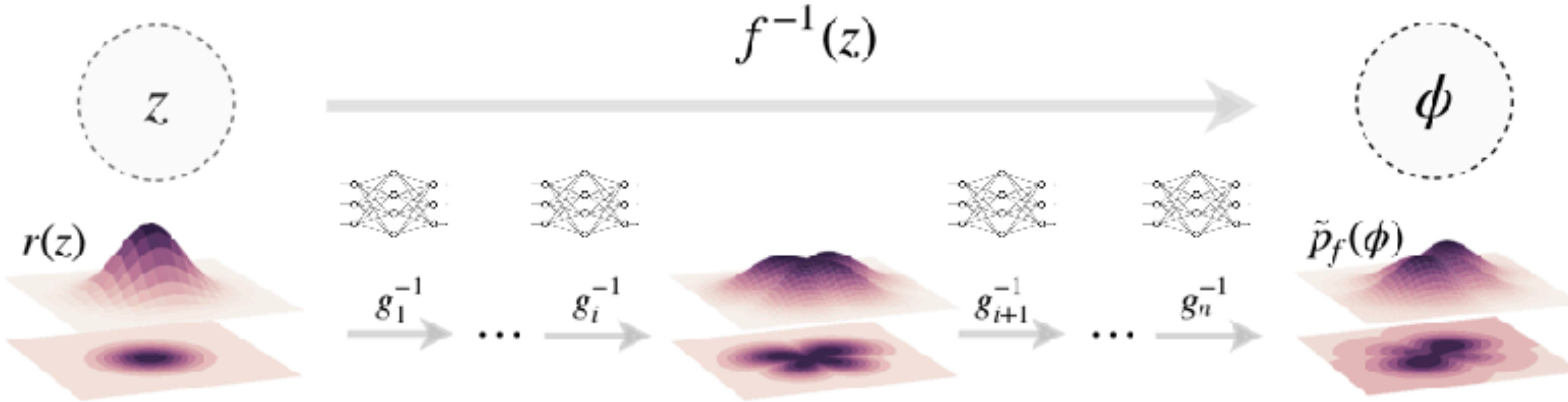
arxiv 1904.12072, 2003.06413, 2008.05456 and more.

Flow based sampling algorithm

Trivializing map realized using neural network

Normalizing flow? = Change of variable with **neural nets**

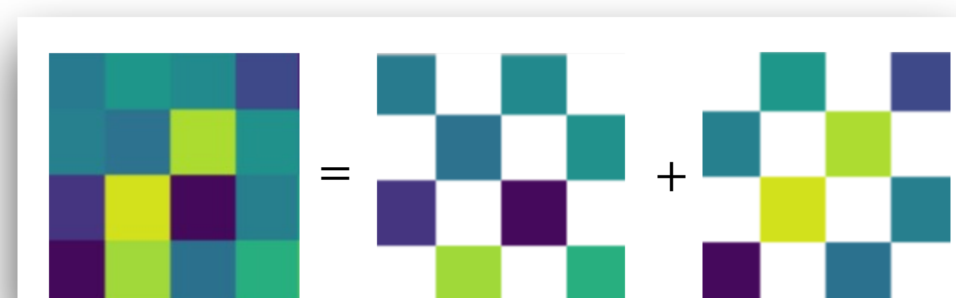
Tractable Jacobian is realized by checker-board technique



(a) Normalizing flow between prior and output distributions

$$\prod_i \int d\varphi_i e^{-V(\varphi_i)} \underline{J[\varphi]} O[F[\varphi]] \approx \int D\phi e^{-S[\phi]} O[\phi]$$

Problem: Jacobian is difficult = O(V^3)
-> Introduce checker-board decomposition



arxiv 1904.12072, 2003.06413, 2008.05456 and more.

Credit: Daniel Hacket (MIT)

Akio Tomiya

Flow based sampling algorithm

Flow based ML for QFT

MIT + Deepmind + …

$$\int D\phi e^{-S[\phi]} O[\phi] \approx \prod_i \int d\varphi_i e^{-V(\varphi_i)} J[\varphi] O[F[\varphi]]$$

Original integral: hard

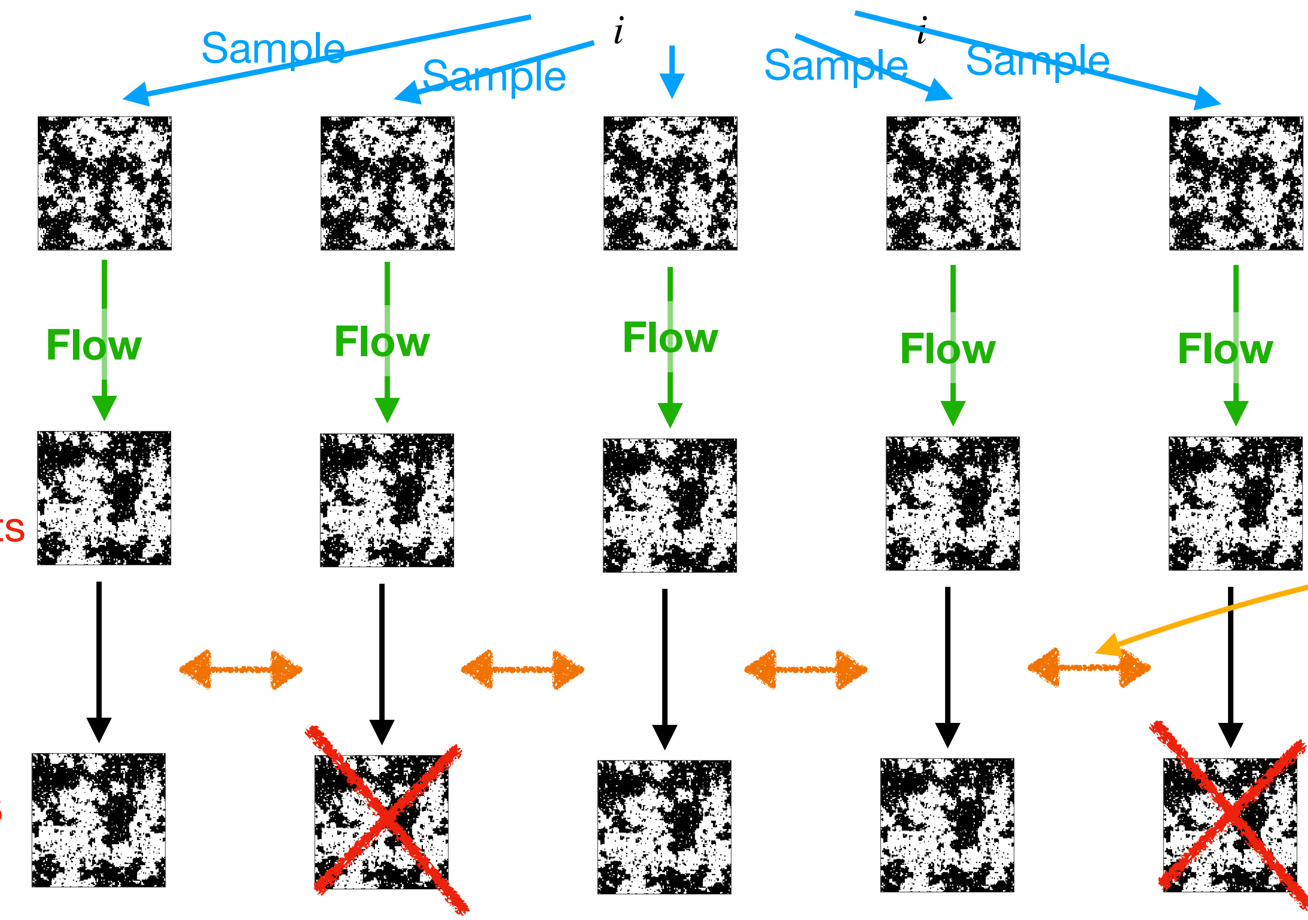
Easy

Flow-based sampling algorithm

$$\prod_i^{Vol} e^{-V(\varphi_i)} = \prod_i^{Vol} r(\varphi_i)$$

①Embarrassedly parallel sampling
from trivial theory
(no kinetic term, no topology)

No auto-correlation
No correlation for points



Reject
(Use left conf.)

Reject
(Use left conf.)

No auto-correlation
Approx.correlation for points

Correct correlations
Small auto-correlation

②"un-trivializing map"

"Cooling = change of variable"
via trained neural net

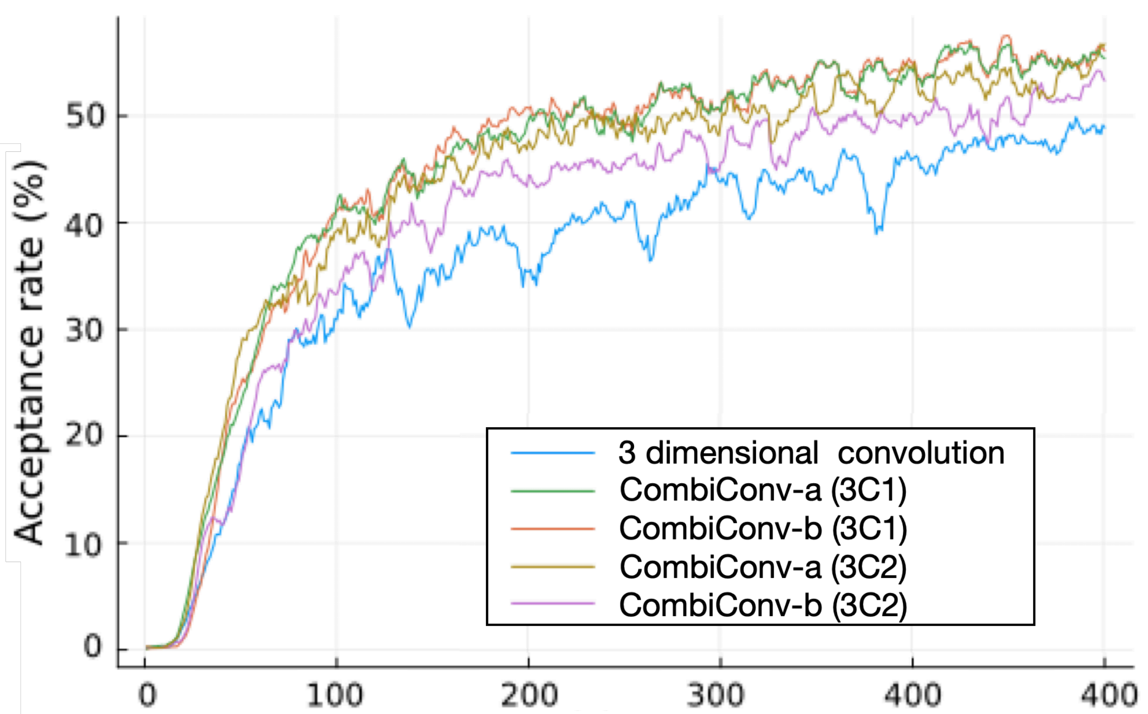
③Metropolis-*Hastings* with
$e^{-S}/e^{-V(\varphi_i)}J^{-1}[\varphi]$
Only sequential

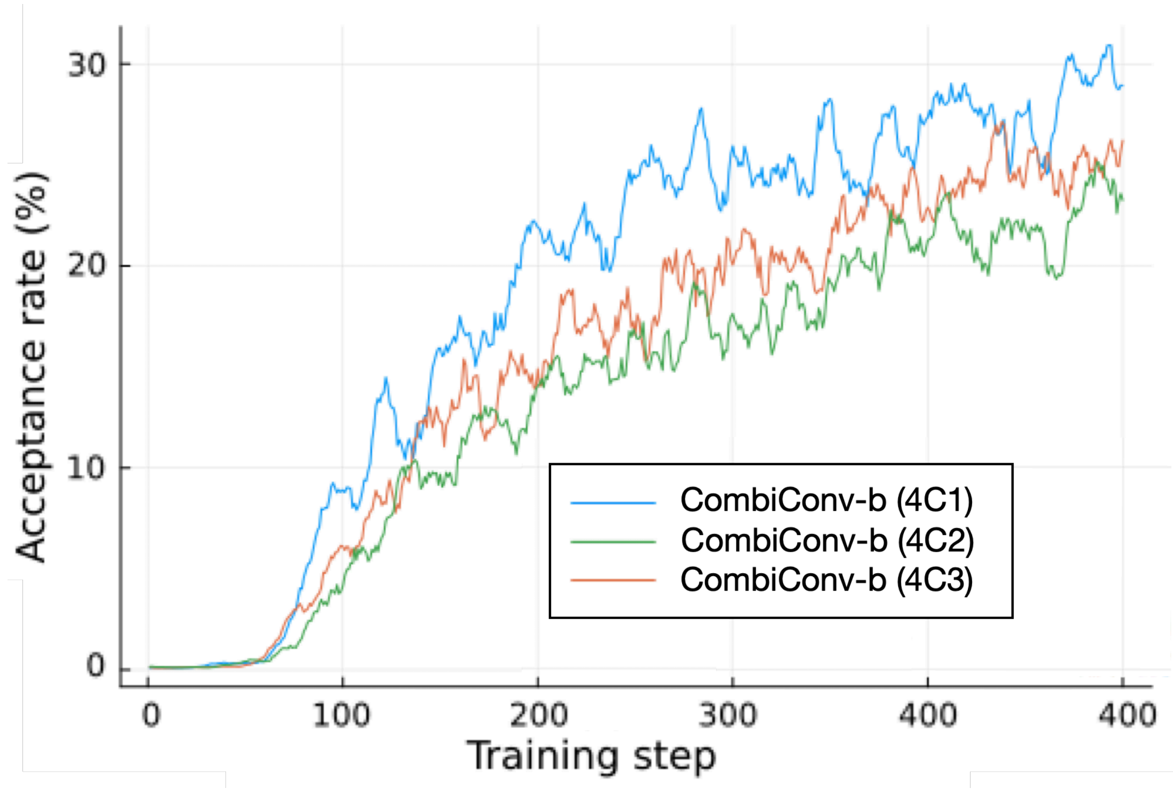
Auto-correlation
~ Rejection rate

Flow based sampling algorithm

Akio Tomiya

We make new convolutional layer for QFT in d-dim



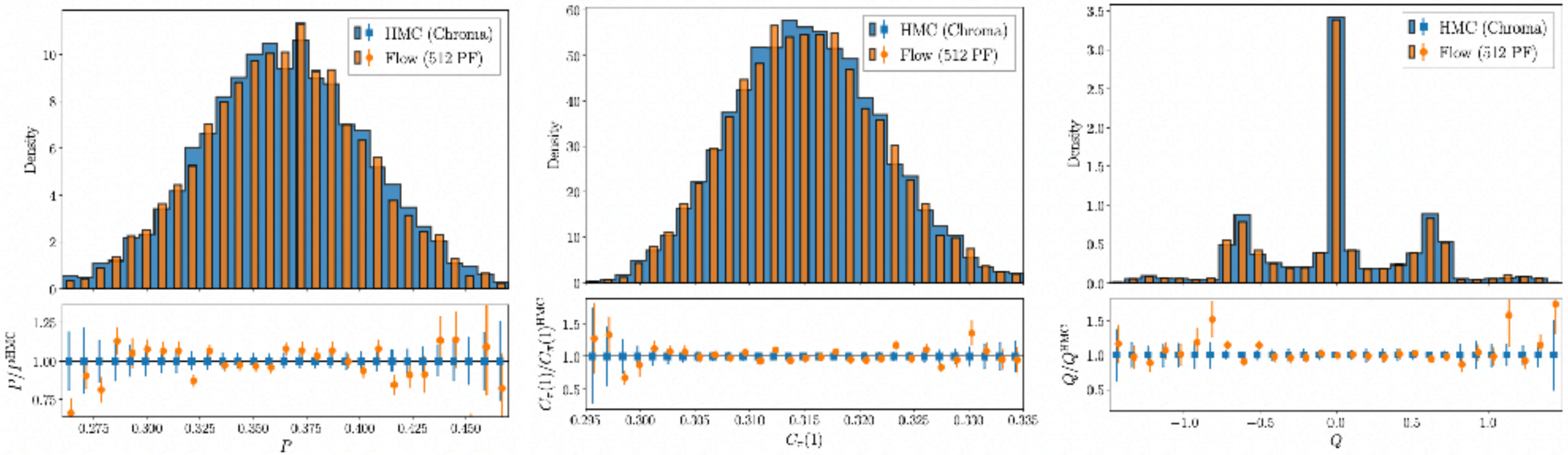


- We implement CombiConv for flow-based sampling algorithm for d-dimensional scalar field theory on the lattice
- 3d convolution is available on GomalizingFlow.jl [1], open source implementation of flow-based sampling algorithm
- nCk $\equiv \dfrac{n!}{k!(n-k)!}$
- In 3d, the acceptance rate is improved for CombiConv compared with the conventional 3d convolution
- In 4d, it works well for any combination of lower dimensional convolution
- This works in any number of dimensions.

Flow based sampling algorithm

Akio Tomiya

Full QCD in L=4



(a) Plaquette

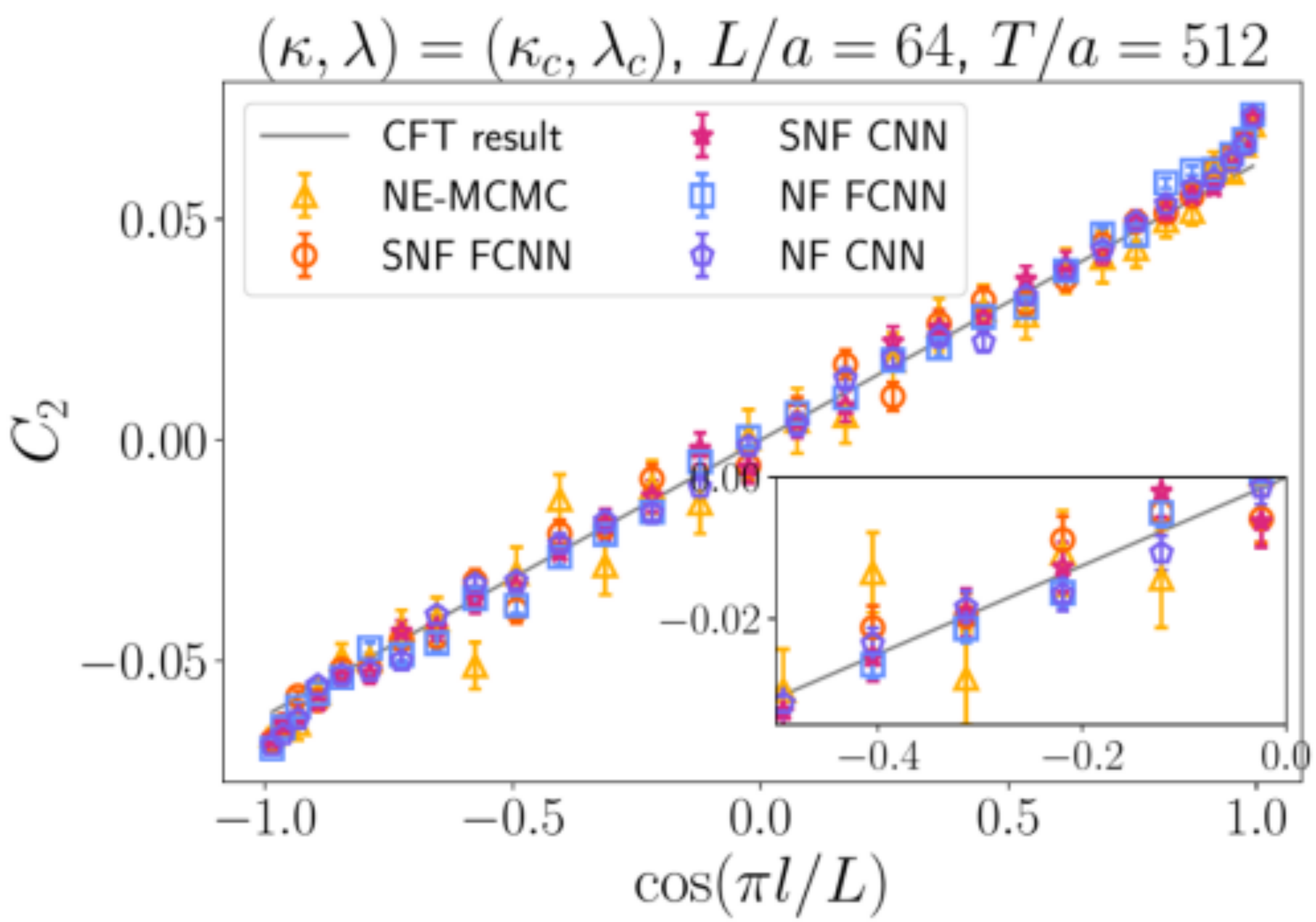
(c) Pion correlation function at $x_0 = 1$

(d) Topological charge at $t/a^2 = 4$

- Very heavy pion but dynamical QCD

Production of configurations

Monte-Carlo integration is available



$$C = \frac{l^{D-1}}{|\partial A|} \frac{\partial S_A}{\partial l},$$

$$S_A = -\mathrm{Tr}(\rho_A \ln \rho_A), \quad \rho_A = \mathrm{Tr}_B\, \rho,$$

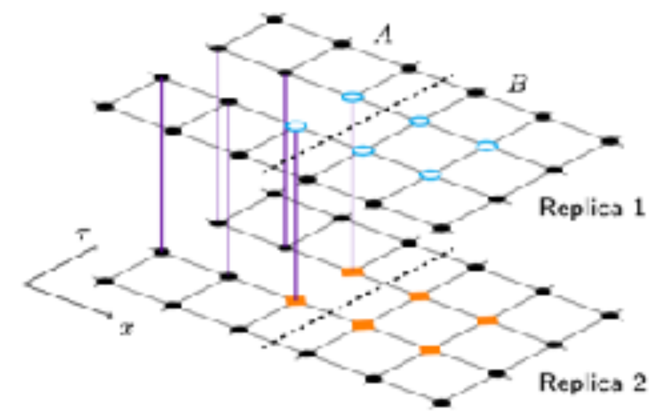


FIG. 1. $(1+1)$-dimensional lattice with two replicas (τ is the Euclidean-time direction). Purple links connect different replicas; dashed lines separate A and B. When defect coupling layers act on the configuration, the lattice is divided in three parts: the environment (black sites), which does not enter the coupling layer; frozen sites (empty cyan circles), that are the neural network input; active sites (orange diamonds), which are transformed by the layer.

Flow based model
mimics the partition function
-> one can calculate the entanglement entropy

https://arxiv.org/abs/2201.08862 https://arxiv.org/abs/2412.00200 https://arxiv.org/abs/2410.14466

Akio Tomiya

Production of configurations

Diffusion model as the stochastic quantization

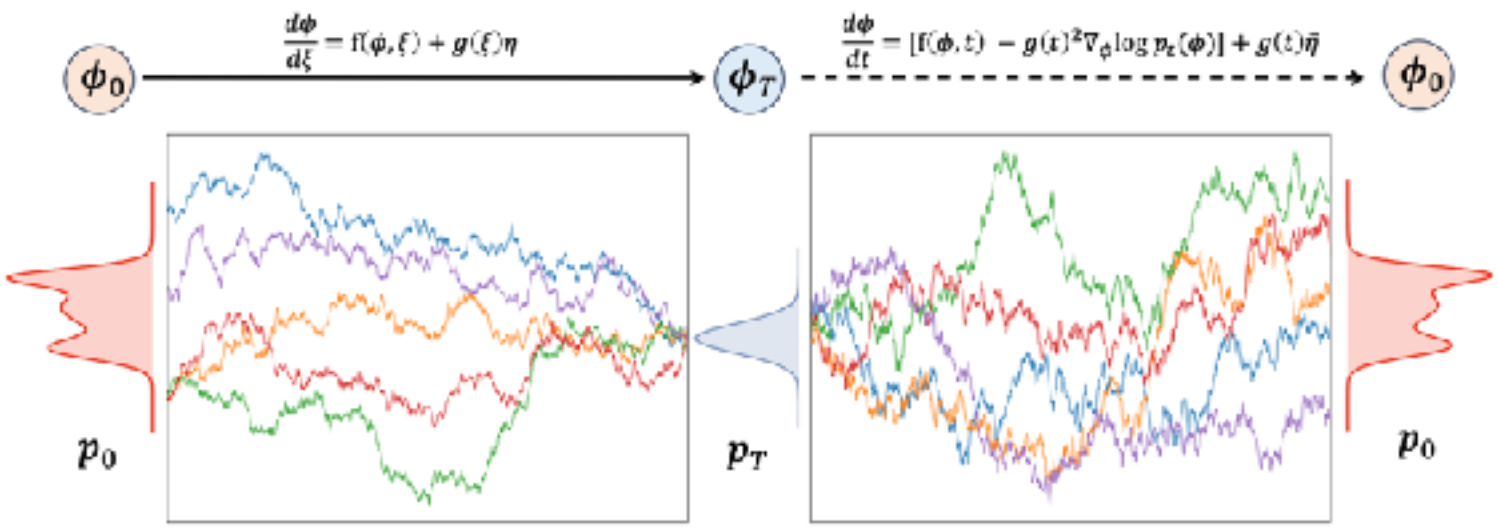


Figure 2: A sketch of the forward diffusion process (left panel) and the reverse denoising process (right panel). The two stochastic processes are described by two stochastic differential equations. The target distribution is typically unknown but learnt from the training data.

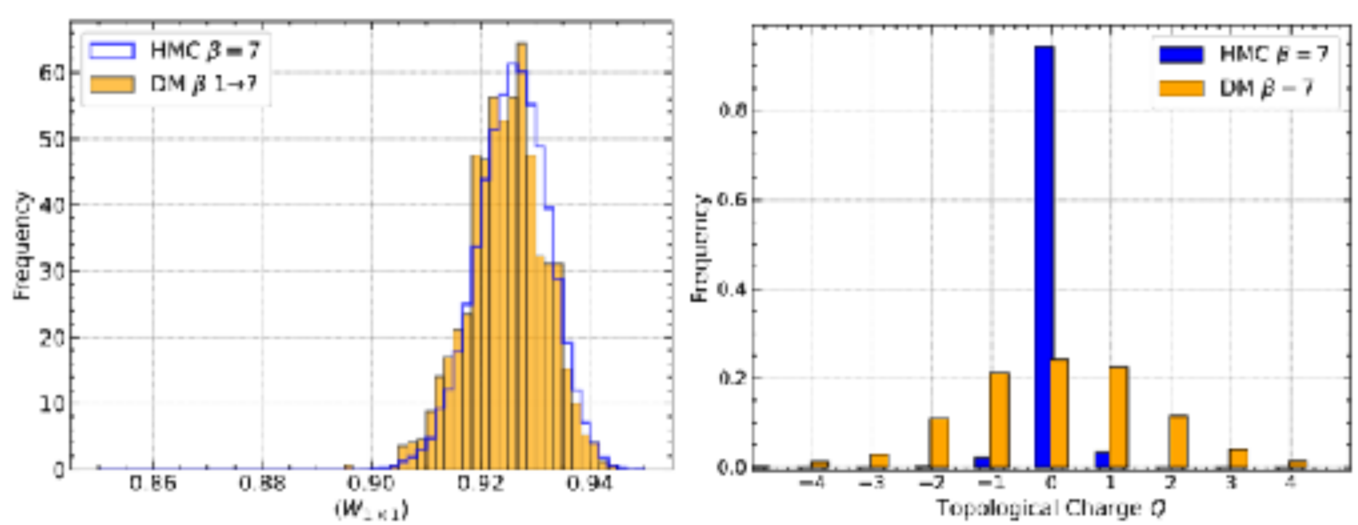


Figure 2: Comparison of distributions for the Wilson loop (left) and the topological charge (right) at $\beta = 7$ from the test data-set (HMC) and from the DM trained at $\beta = 1$ but conditioned at $\beta = 7$. The number of independent configurations is 1,024 in both cases.

As same as diffusion model
for generative AI,
we can sample
gauge configuration
using backward Langevin
with Metropolis test

1+1 U(1)
Lattice gauge theory

https://arxiv.org/abs/2309.17082, L. Wang , G. Aarts and K. Zhou

https://arxiv.org/abs/2410.19602

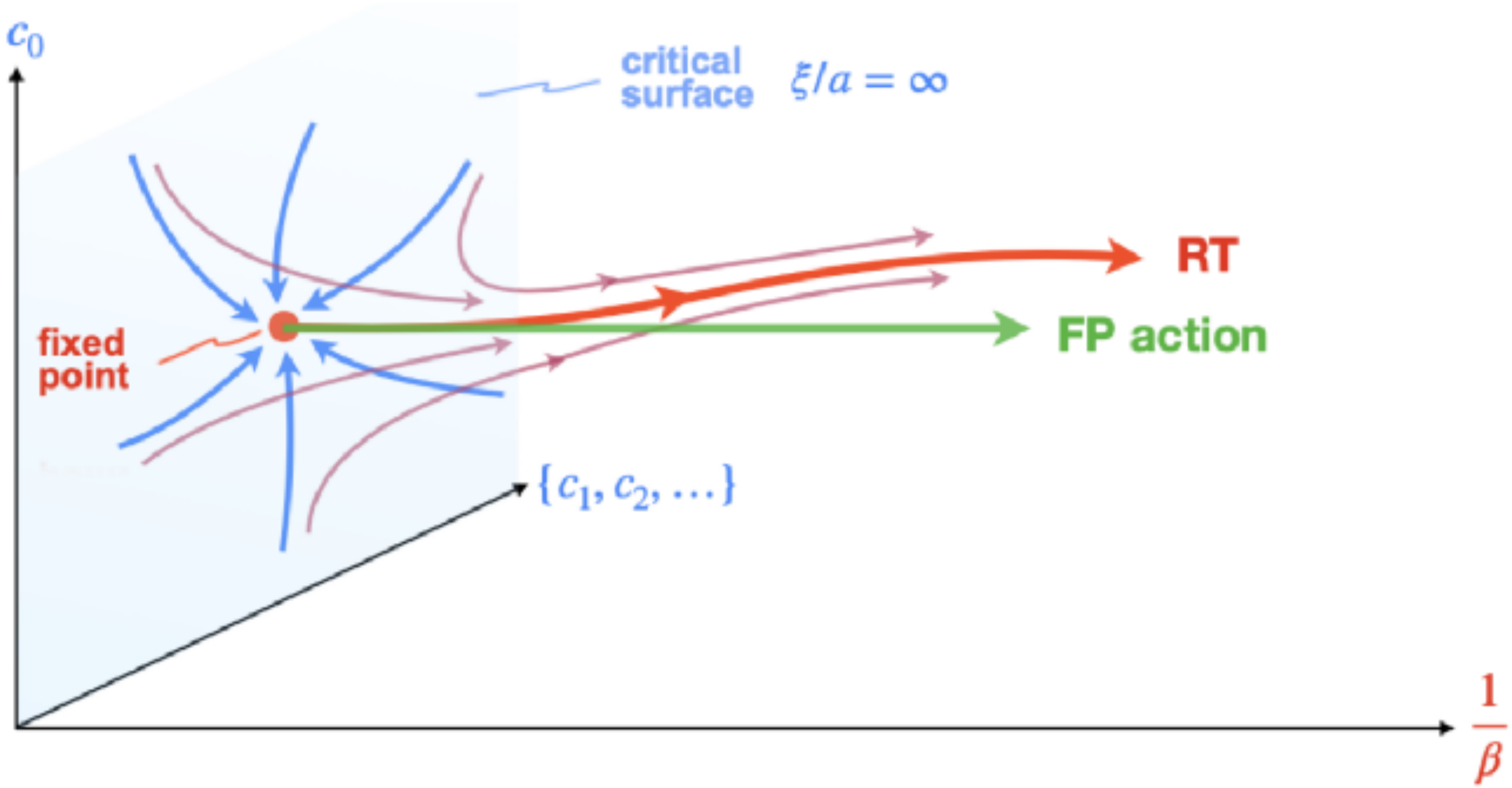
Production of configurations

Misc

Production of configurations

Perfect action

If a lattice action is close to a fixed point,
it has *no* discretization effects



According to P. Hasenfratz's proposal,
we can realize a (classical) "perfect action"

A saddle point solution of the Boltzmann weight

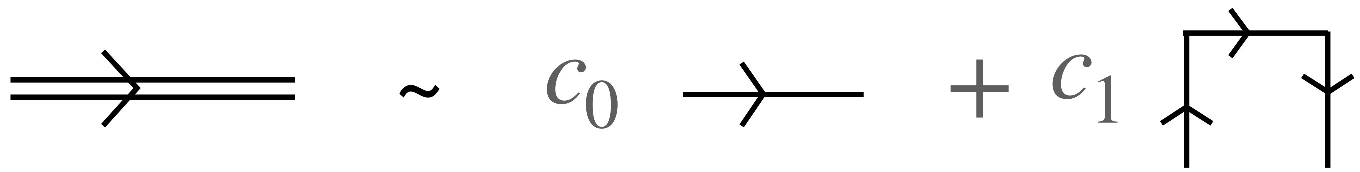
Production of configurations

Akio Tomiya

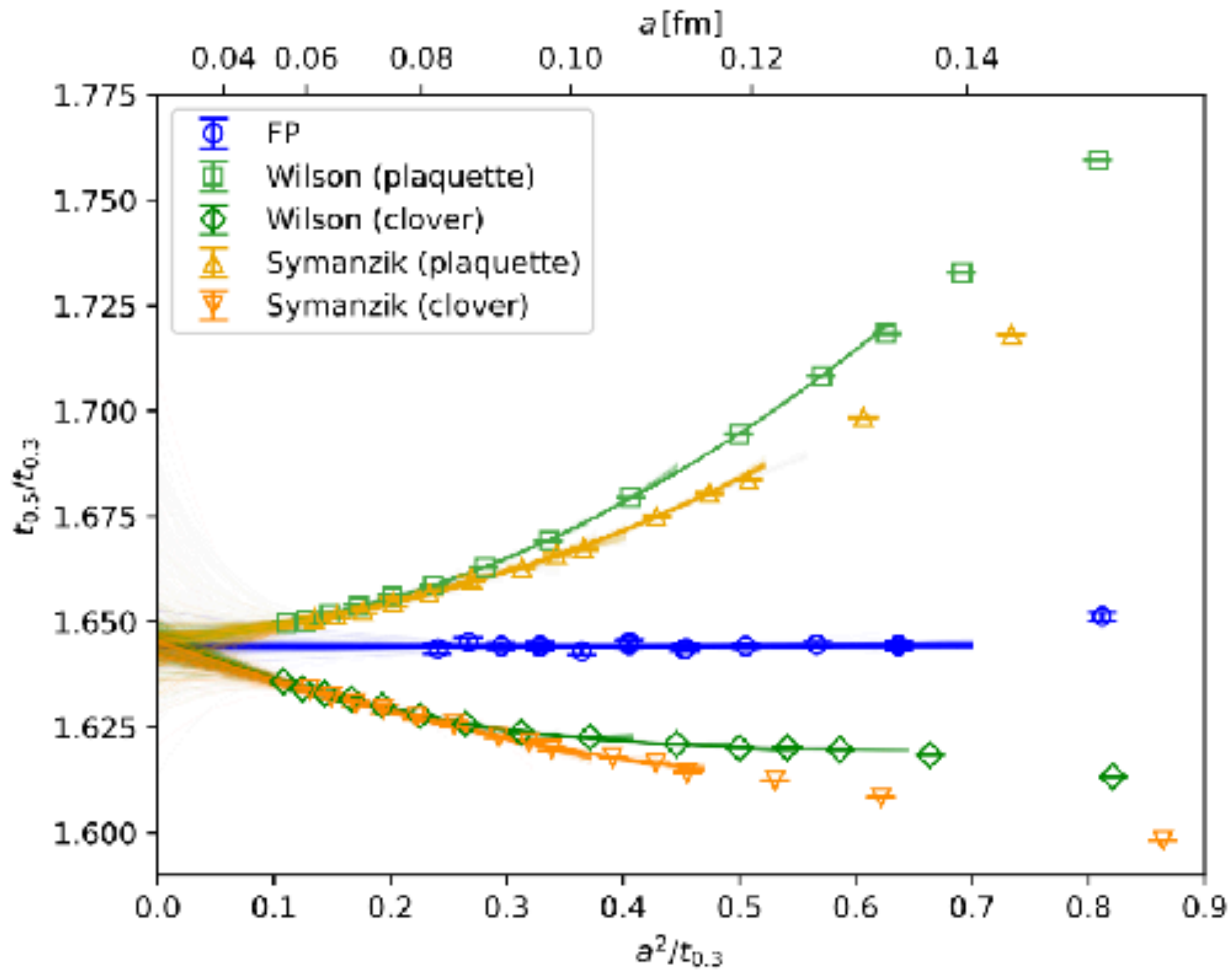
Perfect action

$$\exp\left(-\beta' S'[V]\right) = \int DU \exp(-\beta\{S[U] + T[U,V]\})$$

Blocking kernel



(and highly parametrized via gauge cov net)



$$t^2\langle E(t)\rangle\Big|_{t=t_c} = c$$

$$\frac{t_{0.5}(a)}{t_{0.3}(a)} = \left(\frac{t_{0.5}}{t_{0.3}}\right)_{a=0}\left[1 + b\frac{a^2}{t_{0.3}} + O\left(\frac{a^4}{t_{0.3}^2}\right)\right]$$

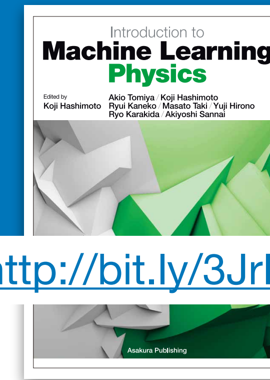
FIG. 1. Continuum-limit extrapolations for the ratios $t_{0.3}/w_{0.3}^2$ and $t_{0.5}/t_{0.3}$. Results from Wilson and Symanzik MC simulations are shown using plaquette and clover discretizations of the action density.

Summary

M + lattice field theory

Akio Tomiya

Code



http://bit.ly/3JrEjls

This talk is based on JPSJ 94 (2025) 3, 031006

- Production and measurement need numerical cost
- Machine learning is useful for natural science/physics/Lattice QCD
  - to reduce cost in different ways
  - Supervised learning requires data ahead of training
  - Self-learning does not require it (SLHMC&Flow).
- Now, machine learning techniques are bias free
  - Gauge case, architectures are gauge covariant!
  - We can remove bias from ML
- Some results show better than existing algorithms (not all)

MLPhys Foundation of "Machine Learning Physics"
Program for Promoting Researches on the Supercomputer Fugaku
Large-scale lattice QCD simulation and development of AI technology
KAKENHI: 20K14479, 22H05112, 22H05111, 22K03539

Thanks!