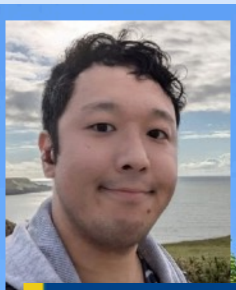
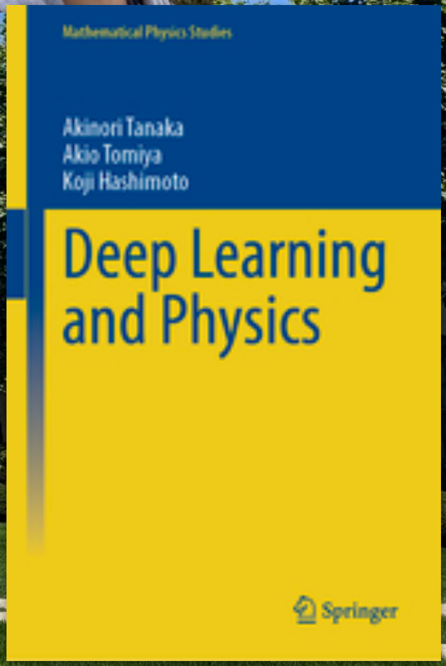
German-Japan Seminar
2024/9/27 (Fri) 11:00-11:30
Mainz

https://indico.zdv.uni-mainz.de/event/16/
In person

Machine learning in lattice gauge theory

Akio Tomiya (Lecturer/Jr Associate prof)
Tokyo Woman's Christian University





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

akio_at_yukawa.kyoto-u.ac.jp

Tsukuba Direction

https://www.twcu.ac.jp/main/english/index.html

My team: LQCD + ML

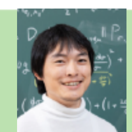
"Machine Learning Physics Initiative"

MLPhYs

2022-2027, 10M USD, 70 researchers

Director : K. Hashimoto

B01 A.Tanaka: Math and Application of DL



B02 Y.Kabashima: Statistical data ML

B03 K.Fukushima: Topology and Geometry of ML

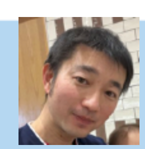
+...

A01 A.Tomiya: Computational physics

+...

A02 M.Nojiri: High Energy Physics

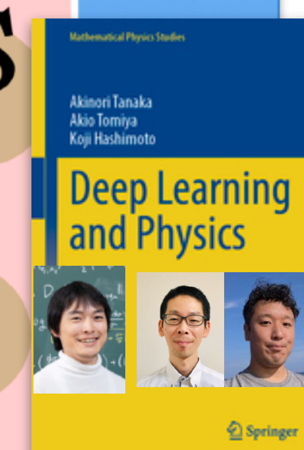
A03 T.Ohtsuki: Condensed Matter Physics



A04 K.Hashimoto: Quantum and Gravity Physics

+...

ML Phys



2021

FY2022-2026 MEXT -KAKENHI- Grant-in-Aid for Transformative Research Areas (A)

科研費 KAKENHI

My team (A01): LQCD + ML

MLPhYs

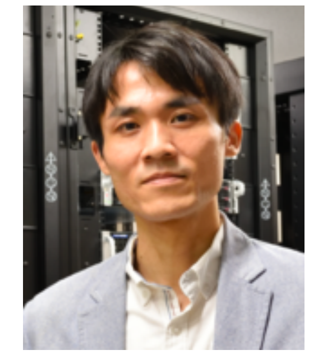
PI: Akio Tomiya (Me)
TWCU
LQCD, ML

Kouji Kashiwa
Fukuoka Institute of Technology
LQCD, ML

Hiroshi Ohno
U. of Tsukuba
LQCD

Tetsuya Sakurai
U. of Tsukuba
Computation

Yasunori Futamura
U. of Tsukuba
Computation



post-docs
& external members

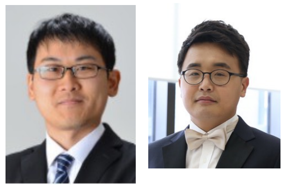
B. J. Choi
U. of Tsukuba

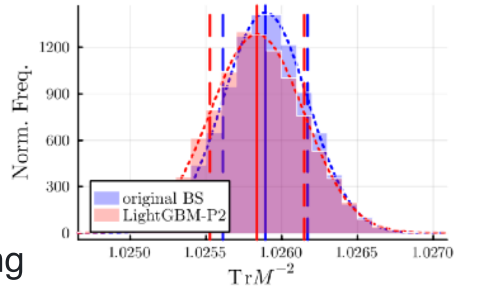
J. Takahashi
Meteorological College

Y. Nagai
U. of Tokyo

- **Apply machine learning techniques on LQCD (To increase what we can do)**
- **Find physics-oriented ML architecture**
- **Making codes for LQCD + ML**

measurement with BDT





On going

LatticeQCD.jl

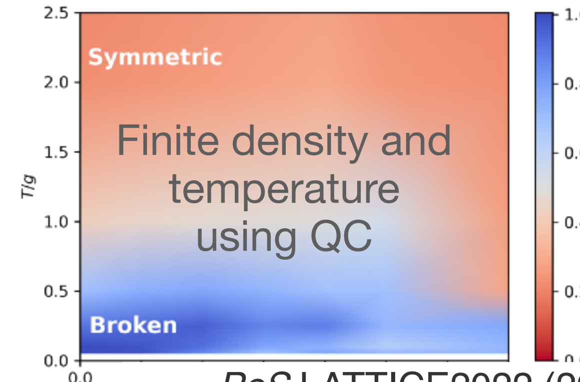
Open source

LQCD (+ML) with julia

This covers most of modern tech

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

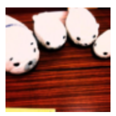
(and associated sub-libraries)

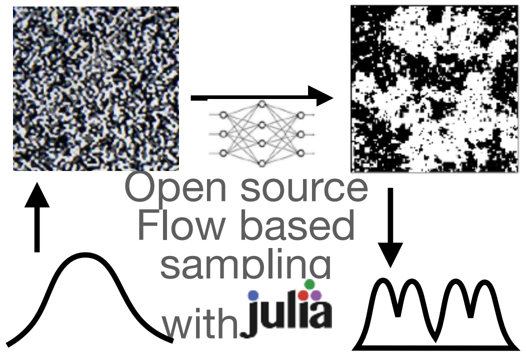


Finite density and temperature using QC

PoS LATTICE2022 (2023) 039

ML + QC: Quantum thermodynamics using Density matrix and MADE





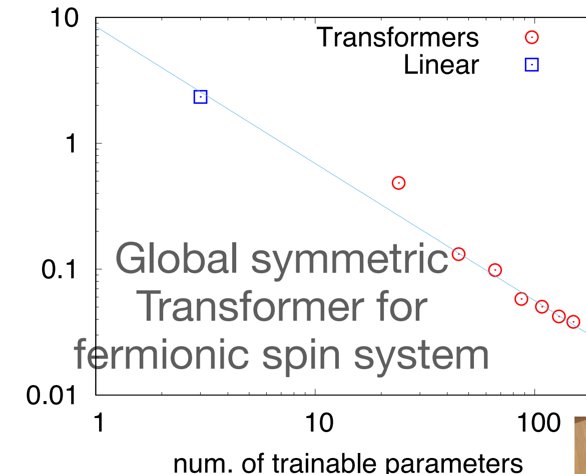
Open source Flow based sampling with julia

arXiv: 2208.08903

ML Phys A01

$$\frac{dU_\mu^{(t)}(n)}{dt} = \mathcal{G}^{\bar{\theta}}(U_\mu^{(t)}(n))$$

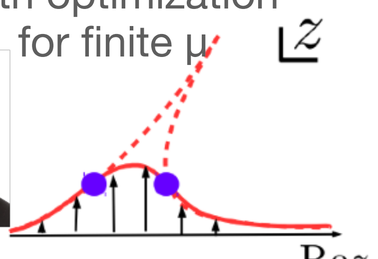
Gauge covariant neural net
arXiv: 2103.11965



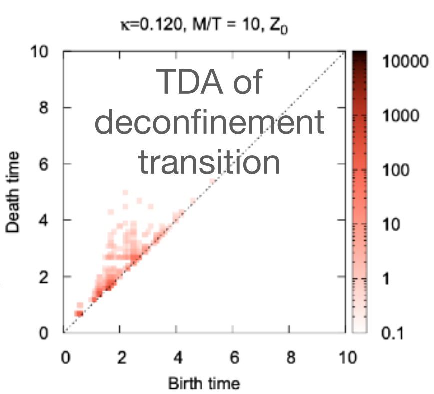
Global symmetric Transformer for fermionic spin system

2310.13222
2306.11527

Path optimization for finite μ



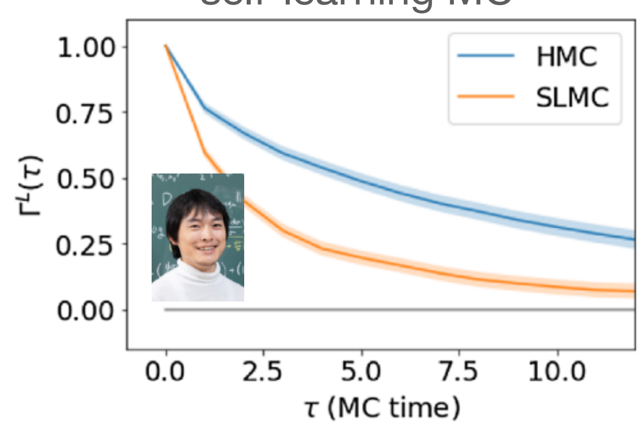
Phys. Rev. D 108, 094504
(Figure from 1812.11506)



TDA of deconfinement transition

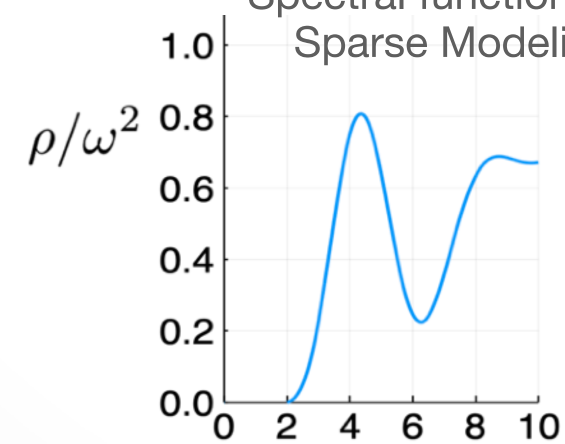
1810.07635

Gauge invariant self-learning MC



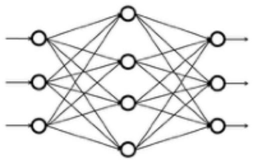
Phys. Rev. D 107, 054501

Spectral function with Sparse Modeling



(arXiv: 2311.15233)
+ on going

Outline of my talk



Machine learning?

Machine learning
for Lattice QCD

1. Transformer for O(3) spin model
2. CASK: Gauge symmetric transformer

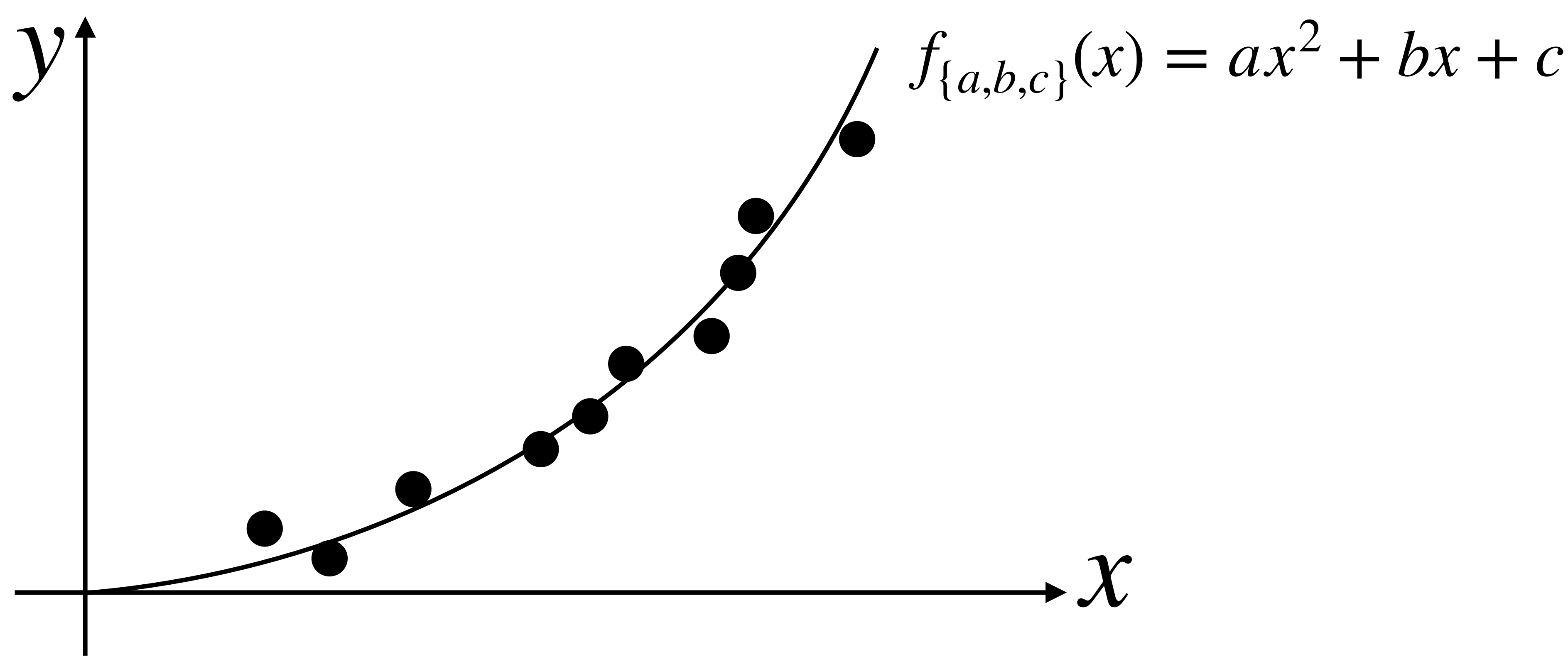
(Measurement -> Hiroshi's talk)

Machine learning?

What is machine learning?

E.g. Linear regression $\in$ Supervised learning

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



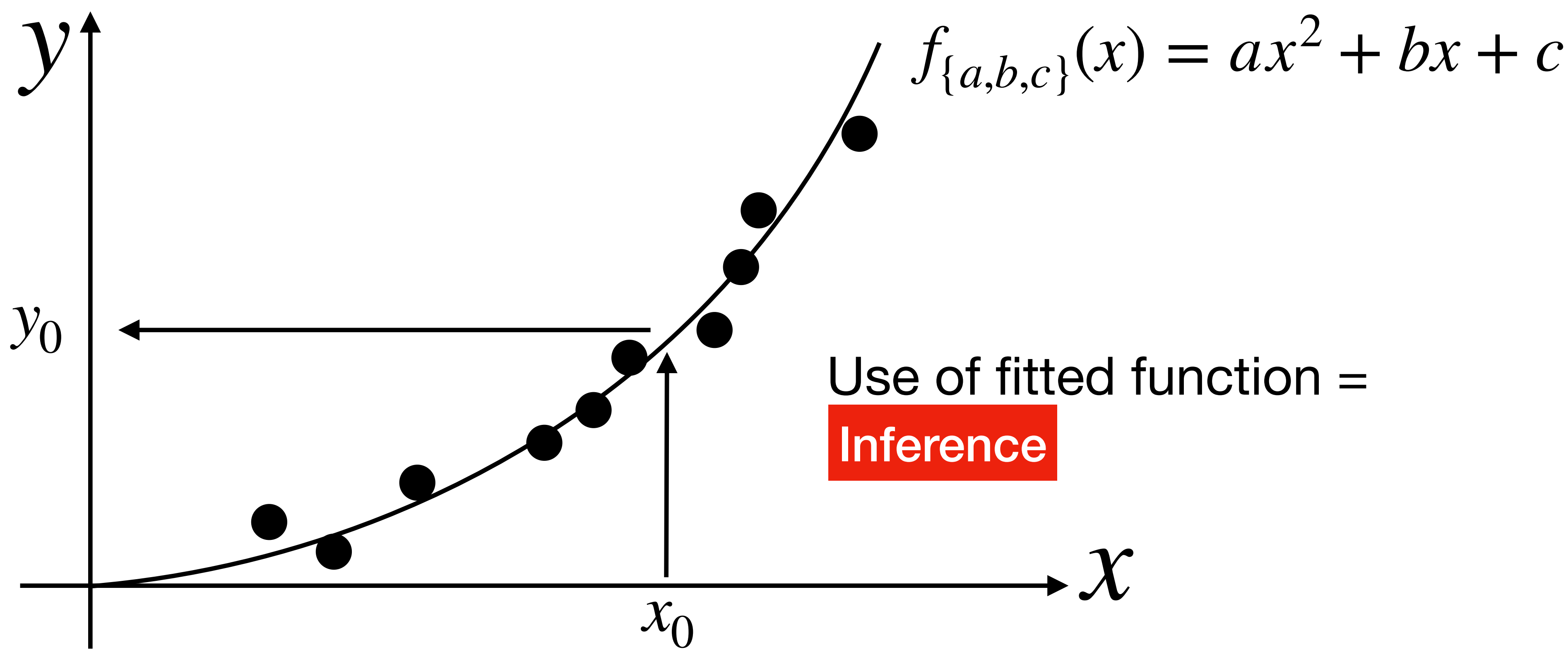
$$f_{\{a,b,c\}}(x) = ax^2 + bx + c \qquad E = \frac{1}{2} \sum_d \left| f_{\{a,b,c\}}(x^{(d)}) - y^{(d)} \right|^2$$

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

What is machine learning?

E.g. Linear regression ∈ Supervised learning

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



Use of fitted function = **Inference**

Now we can predict y value which not in the data

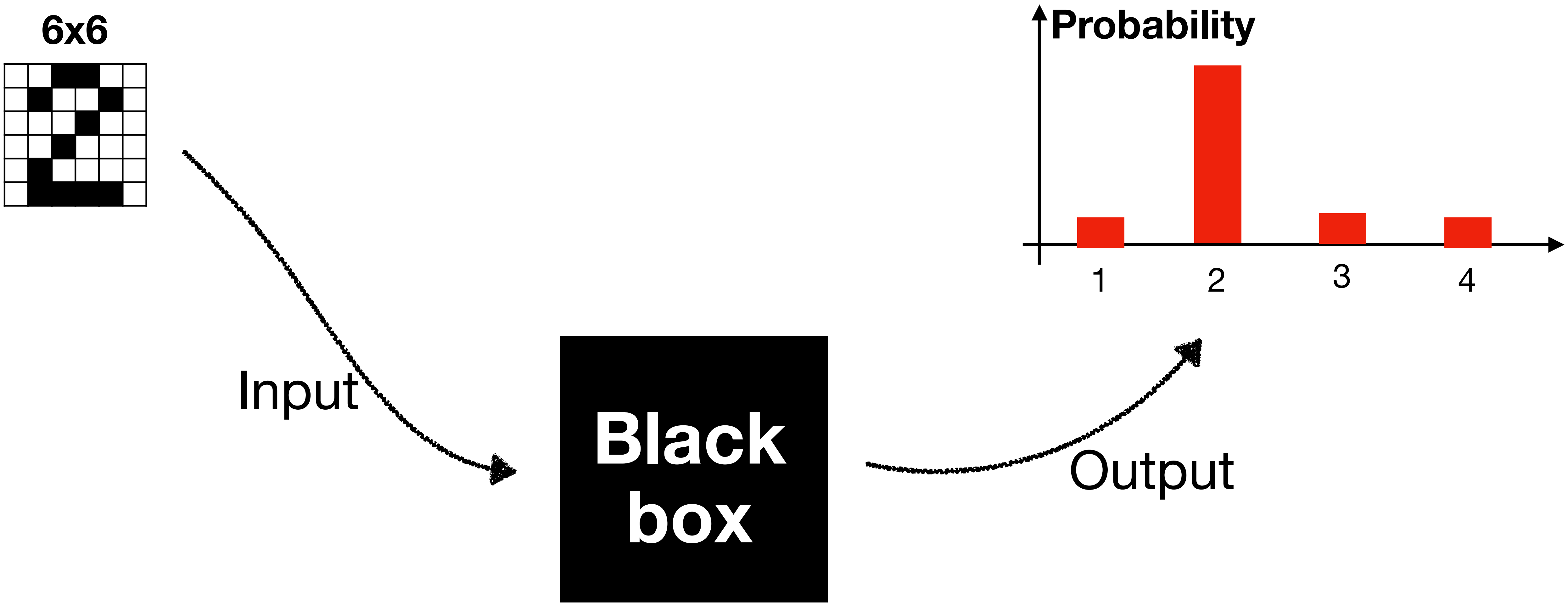
In physics language, variational method

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?

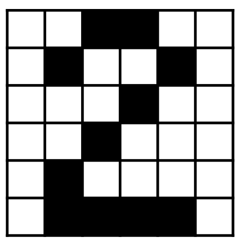
What is the neural networks?

Akio Tomiya

Neural network is a *universal* approximation function

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

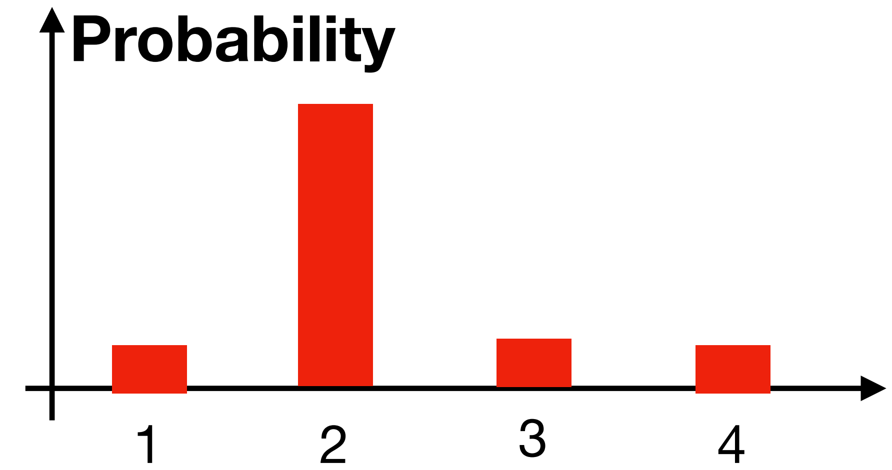
6x6



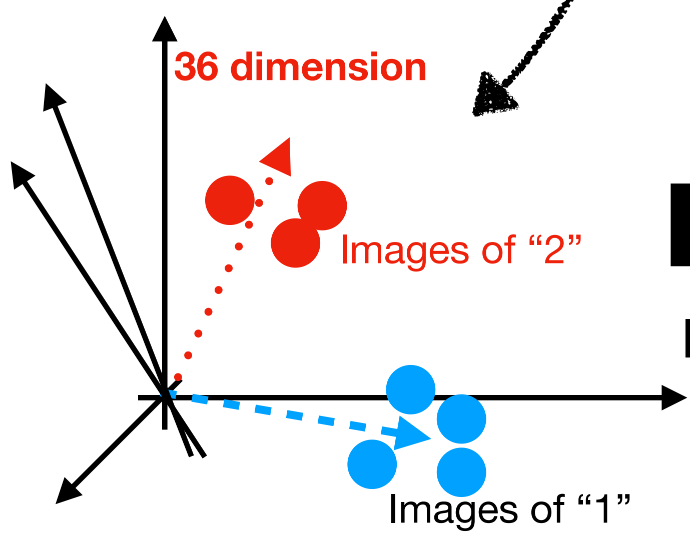
$$= \begin{pmatrix}0.000\\0.000\\0.8434\\0.756\\0.3456\\0.64\\0.251\\\vdots\end{pmatrix} = \vec{x}$$

Image is a vector (6x6=36 dim)

Regard



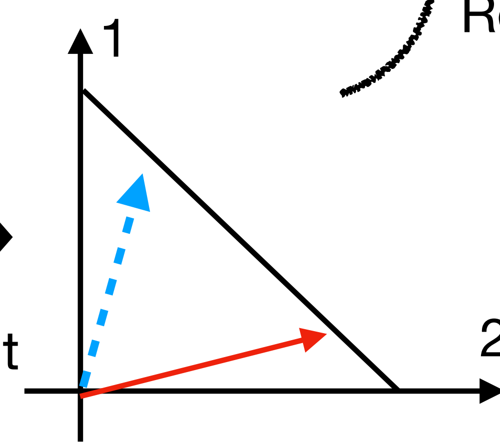
Regard



Input

Black box

Output



10 dimension

Image recognition = Find a map between two vector spaces

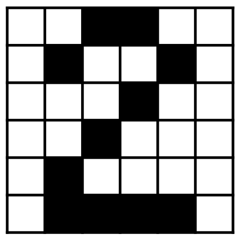
What is the neural networks?

Akio Tomiya

Neural network is a *universal* approximation function

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

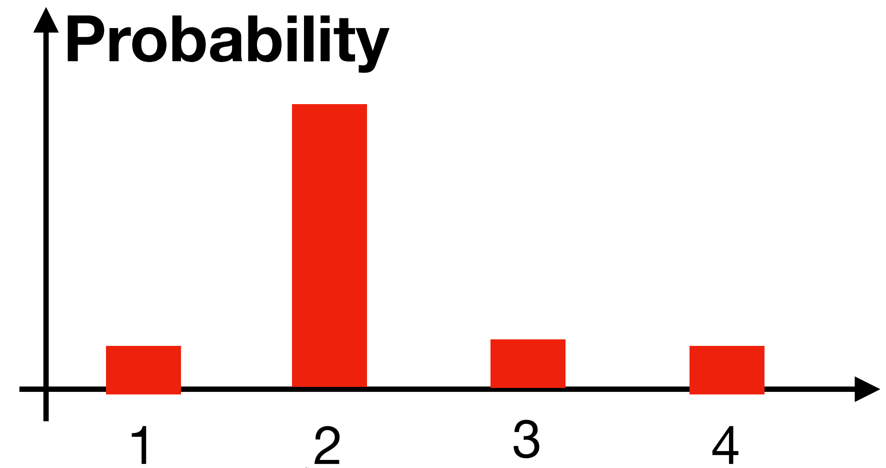
6x6



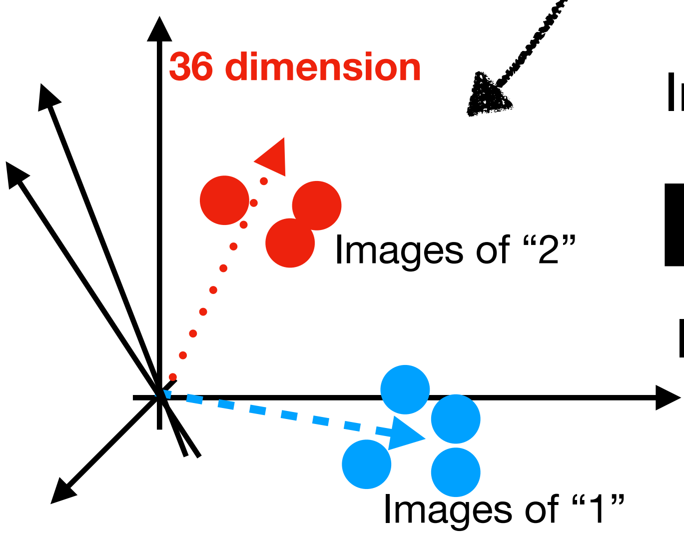
$$= \begin{pmatrix} 0.000 \\ 0.000 \\ 0.8434 \\ 0.756 \\ 0.3456 \\ 0.64 \\ 0.251 \\ \vdots \end{pmatrix} = \vec{x}$$

Image is a vector (6x6=36 dim)

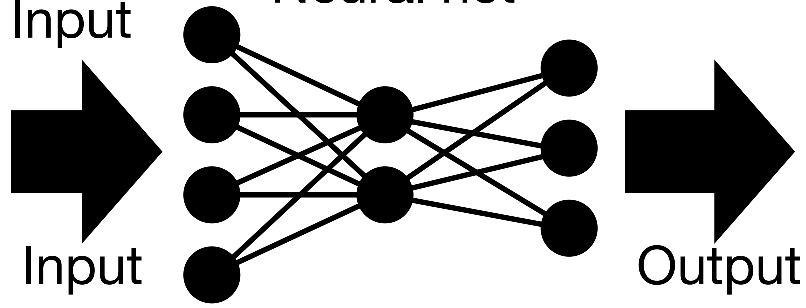
Regard



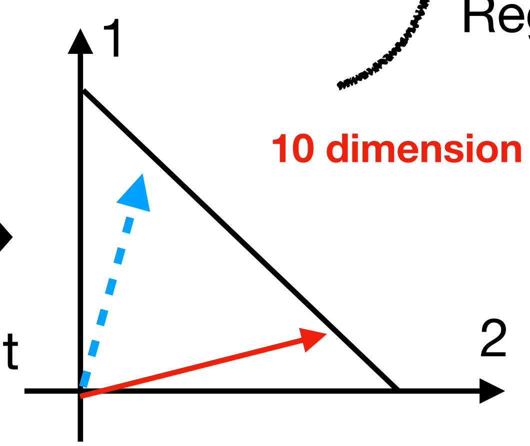
Regard



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



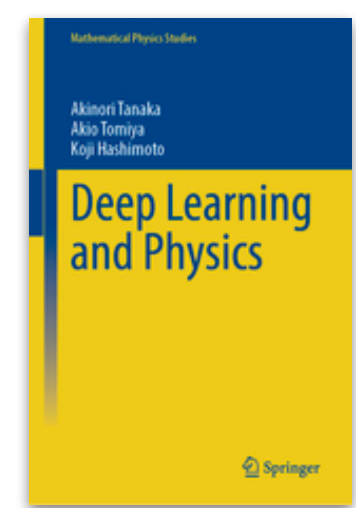
=variational function



“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.

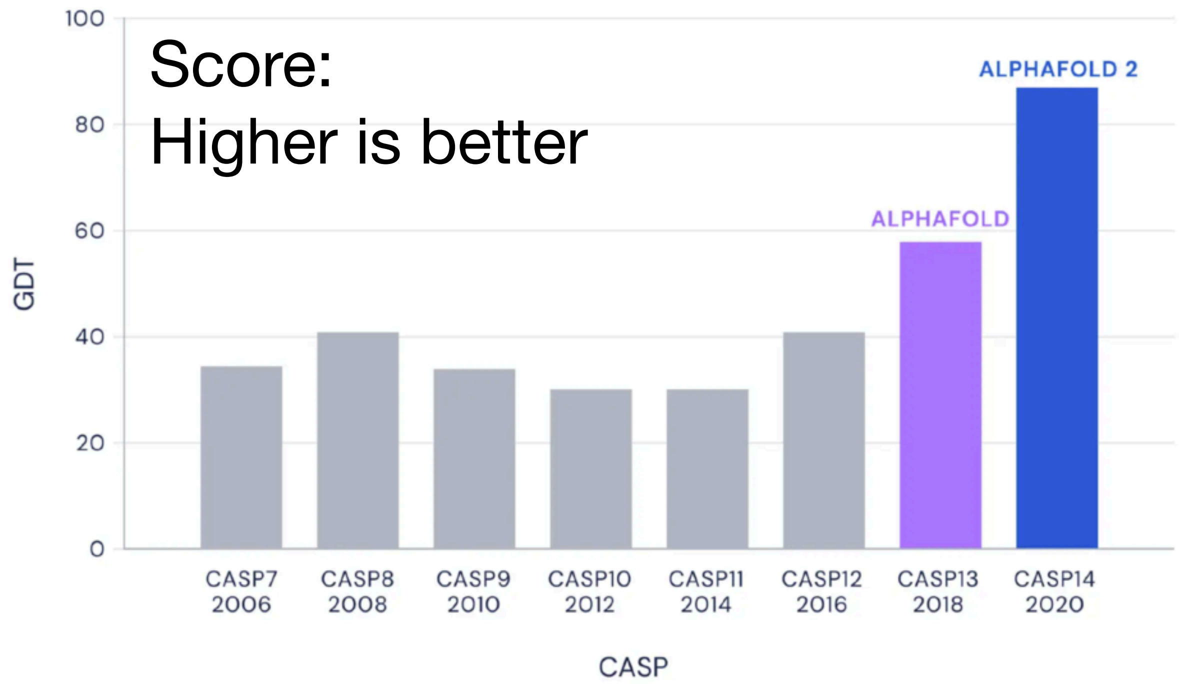
In this example, NN mimics image (36-dim vector) and label (10-dim vector)

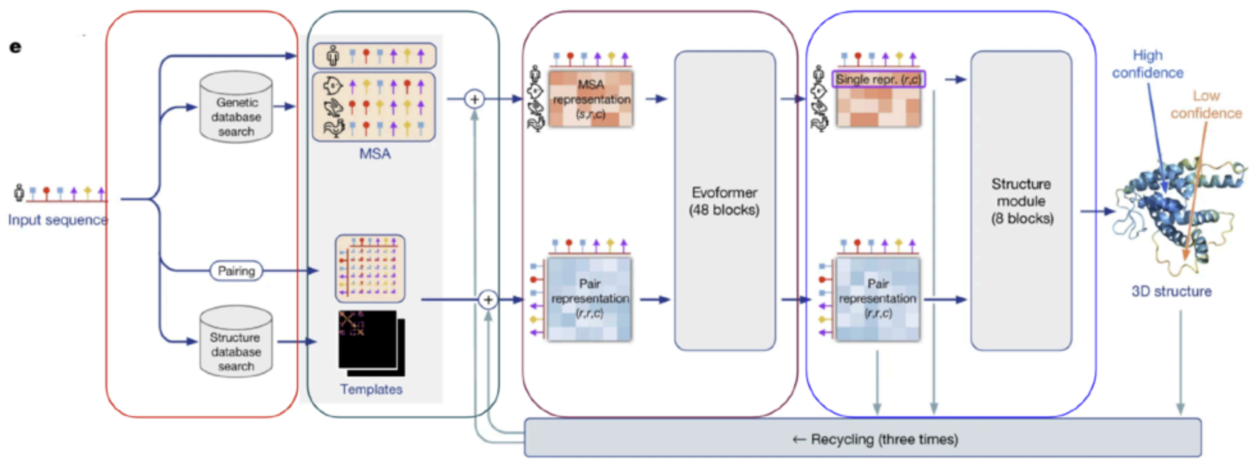


What is the neural networks?

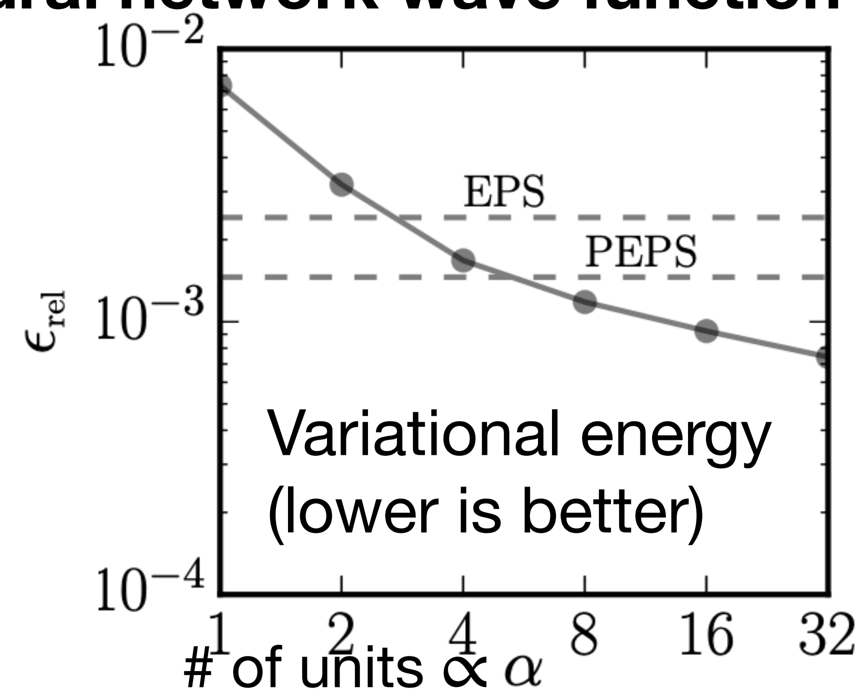
Neural network have been good job

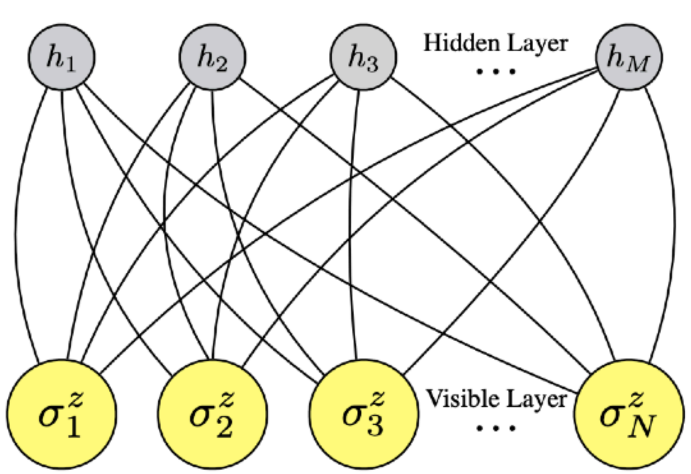
Protein Folding (AlphaFold2, John Jumper+, Nature, 2020+), Transformer neural net





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





Neural net + "Expert knowledge" → Best performance

https://horomary.hatenablog.com/entry/2021/10/01/194825
https://www.aist.go.jp/aist_j/press_release/pr2020/pr20201111/pr20201111.html

Transformer and Attention

Akio Tomiya

Attention layer used in Transformers (GPT, Gemini)

arXiv:1706.03762

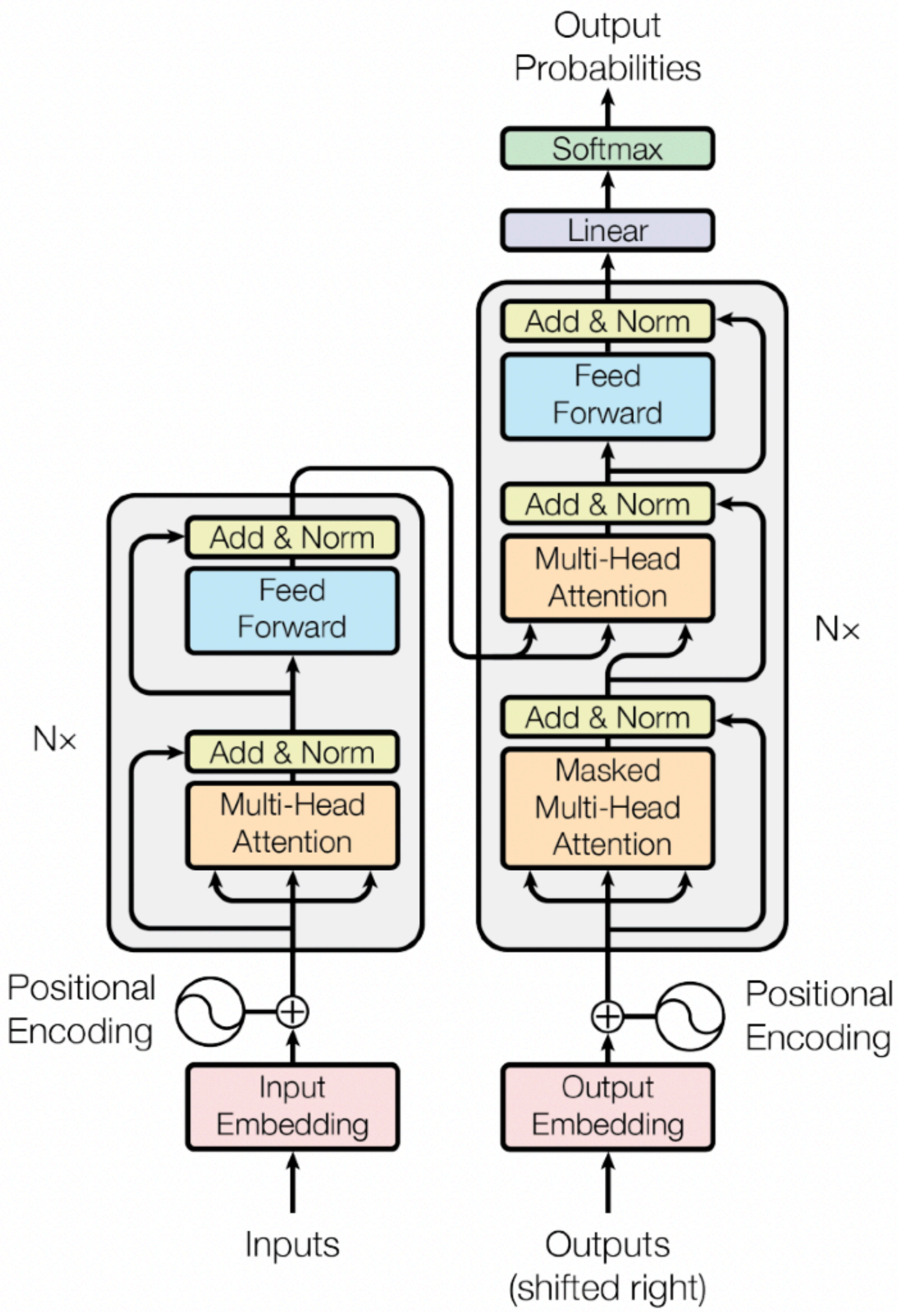


Figure 1: The Transformer - model architecture.

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.

Transformer and Attention

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!

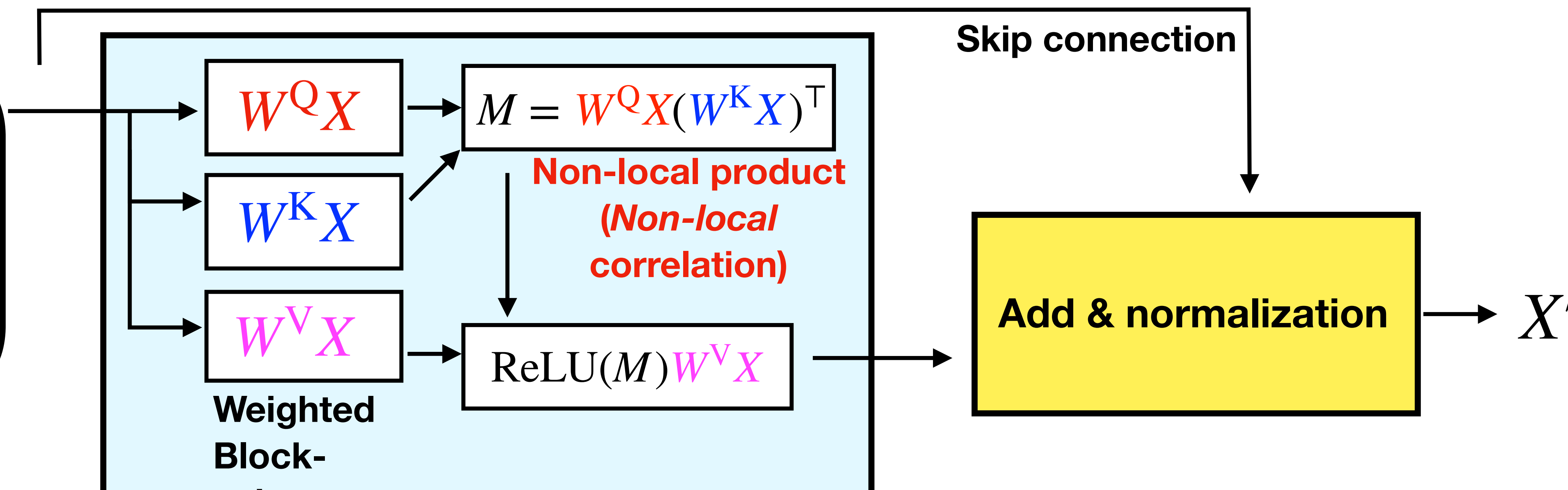
Simplified version of Attention/Transformer

$$X = \begin{pmatrix} \mathrm{I} \\ \mathrm{am} \\ \mathrm{Akio} \\ \vdots \end{pmatrix}$$

Array of word vectors

Word~vector

X: matrix



These blocks can be repeatedly applied

Transformer and Attention

Akio Tomiya

Transformer shows scaling lows (power law)

arXiv: 2001.08361

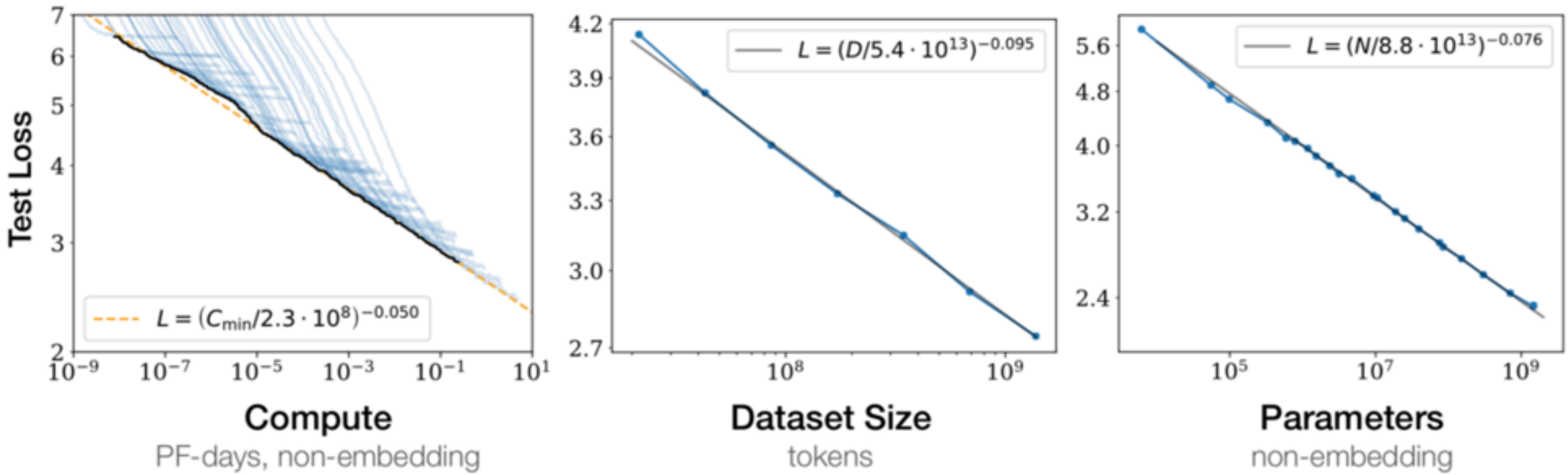


Figure 1 Language modeling performance improves smoothly as we increase the model size, datasetset size, and amount of compute[2] used for training. For optimal performance all three factors must be scaled up in tandem. Empirical performance has a power-law relationship with each individual factor when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

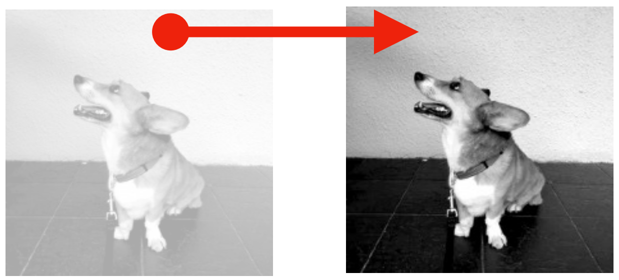
- **It can be improved systematically**
- **Transformers requires huge data**
  (e.g. GPT uses all electric books in the world)
  Because it has few inductive bias (no equivariance)

Equivariance and convolution

Akio Tomiya

Knowledge ∋ Convolution layer = trainable filter, Equivariant

Filter on image



shift to right

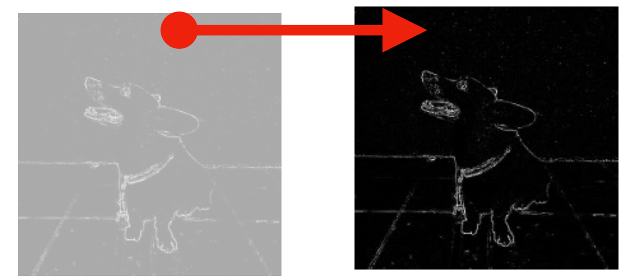
$*$

Laplacian filter

0	1	0
1	-2	1
0	1	0

(Discretization of ∂^2)

$=$

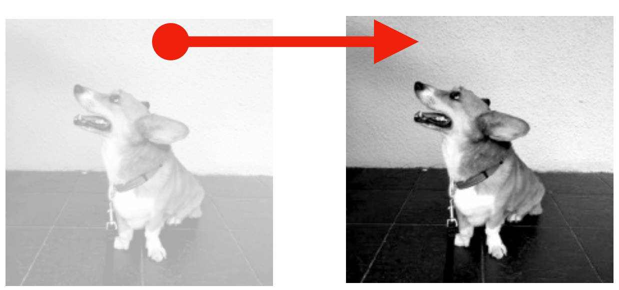


shift to right

Edge detection

Translational operation is *commutable* with filtering (equivariant~covariant)

Convolution layer



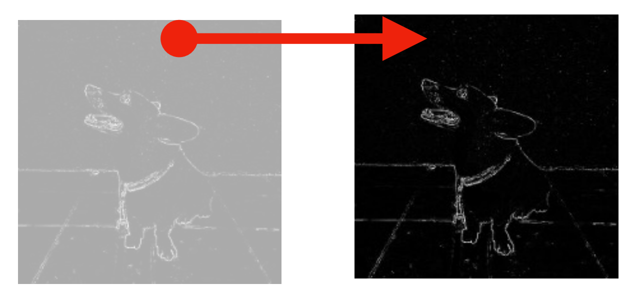
shift to right

$*$

Trainable filter

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

$=$



shift to right

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

Translational operation is *commutable* with convolutional neurons (equivariant)

This can be any filter which helps feature extraction (minimizing loss)

Equivariance reduces data demands. Ensuring symmetry (plausible Inference)

Many of convolution are needed to capture global structures

Machine learning + LQCD?

Background of this work

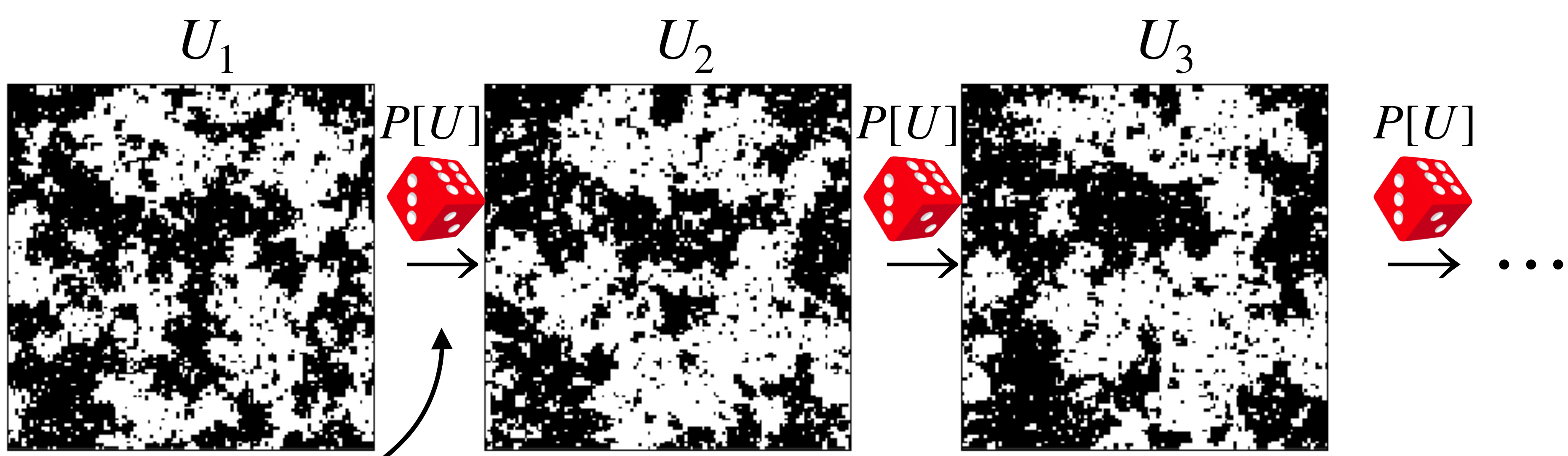
Monte-Carlo integration is available

M. Creutz 1980

Target integration
= expectation value $\langle \mathcal{O} \rangle = \frac{1}{Z} \int \mathcal{D}U e^{-S_{\mathrm{eff}}[U]} \mathcal{O}(U)$

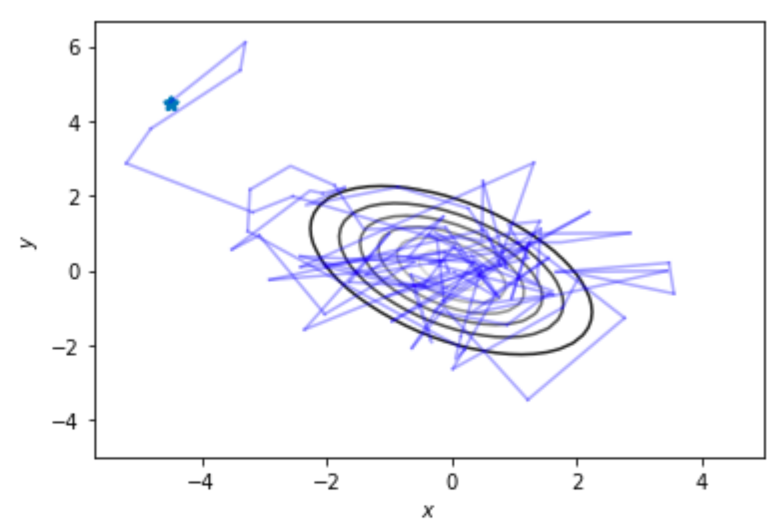
$S_{\mathrm{eff}}[U] = S_{\mathrm{gauge}}[U] - \log \det(\not{D}[U] + m)$

Monte-Carlo: Generate field configurations with "$P[U] \propto e^{-S_{\mathrm{eff}}[U]}$". It gives expectation value



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

Random momentum + EOM
= Random walk like algorithm



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

Background of this work

Akio Tomiya

Monte-Carlo integration is available, but still expensive!

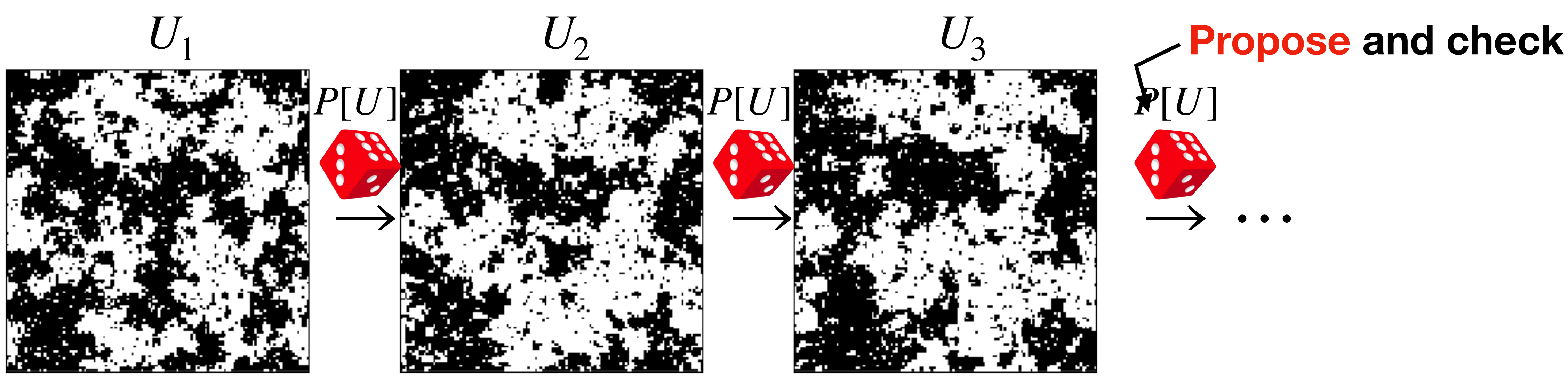
M. Creutz 1980

Target integration
= expectation value $\langle \mathcal{O} \rangle = \frac{1}{Z} \int \mathcal{D}U e^{-S_{\mathrm{eff}}[U]} \mathcal{O}(U)$

$S_{\mathrm{eff}}[U] = S_{\mathrm{gauge}}[U] - \log\det(D\!\!\!/[U] + m)$

Monte-Carlo: Generate field configurations with “$P[U] \propto e^{-S_{\mathrm{eff}}[U]}$” . It gives expectation value

Markov-Chain



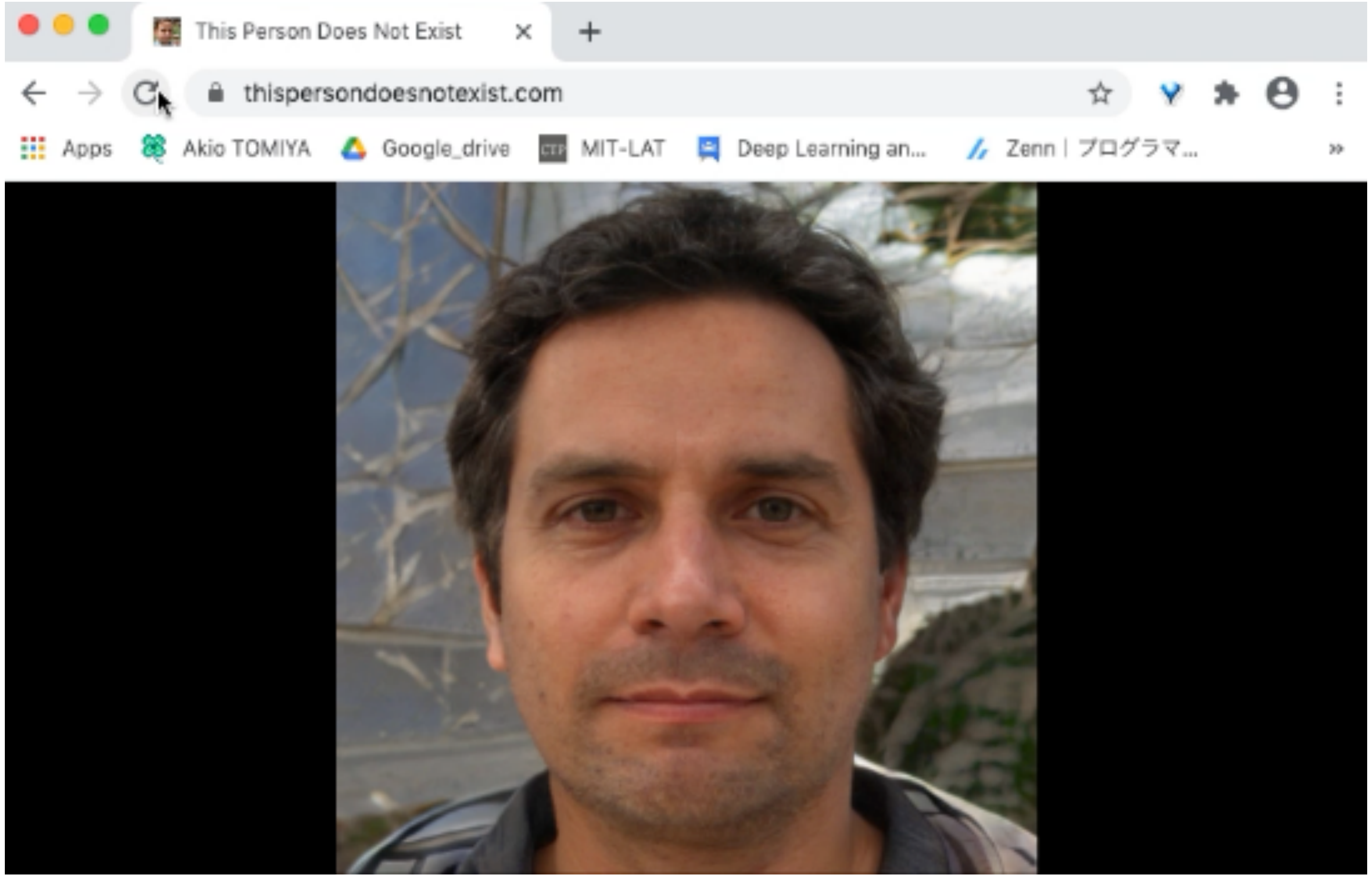
$$\langle \mathcal{O} \rangle \approx \frac{1}{N_{\mathrm{sample}}} \sum_{k=1}^{N_{\mathrm{sample}}} \mathcal{O}[U_k]$$

Production with is numerically expensive
and how can we accelerate it? We use machine learning!

Background of this work

Generative neural net can make human face images

Neural nets can generate realistic human faces (Style GAN2)



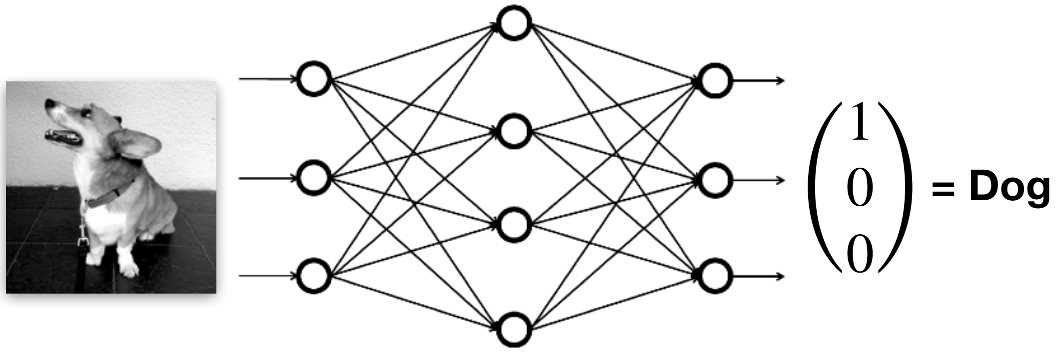
Realistic Images can be generated by machine learning!
Configurations as well? (proposals ~ images?)

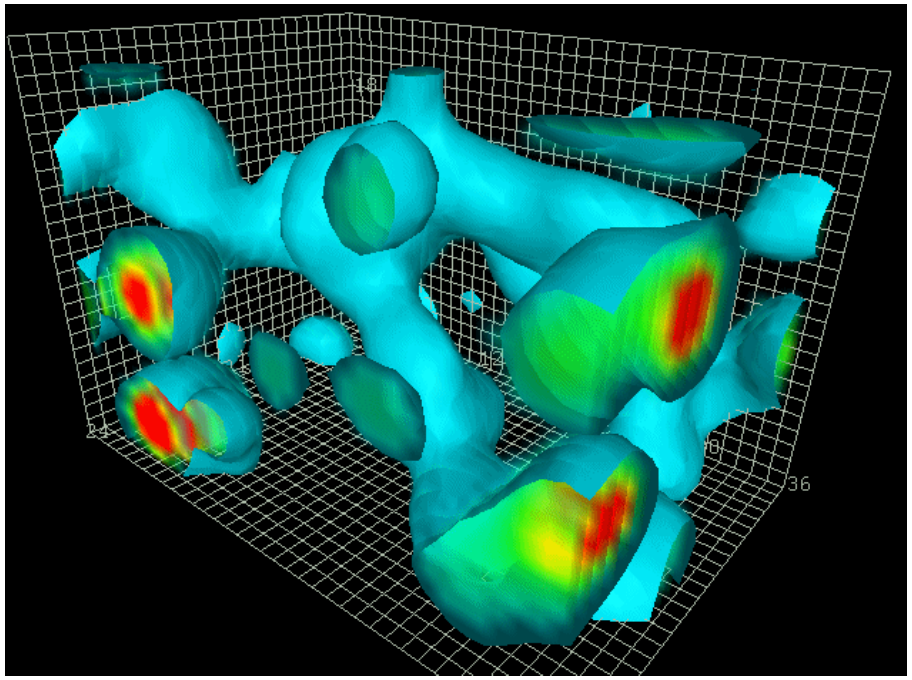
Background of this work

Akio Tomiya

Machine learning for LQCD, LQCD with machine learning

- Neural networks
    - data processing techniques for 2d/3d data in the real world (pictures)
    - (Variational) Approximation ($\sim$ fitting)
    - Generative NN can generate images/pictures
- Lattice QCD is more complicated than pictures
    - 4 dimension/relativistic
    - Non-abelian **gauge symmetry** (difficult)
    - Fermions (anti-commuting/fully quantum)
      -> Non-local effective correlation in gauge field
    - **Exactness** in MCMC is necessary!
- Q. How can we deal with?





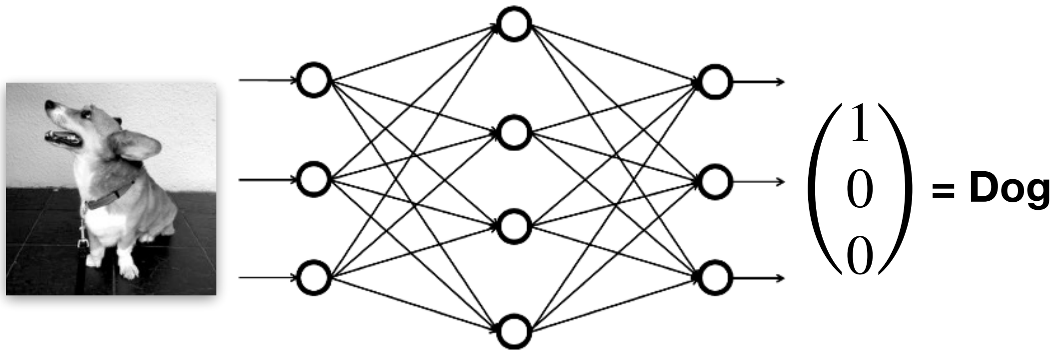
http://www.physics.adelaide.edu.au/theory/staff/leinweber/VisualQCD/QCDvacuum/

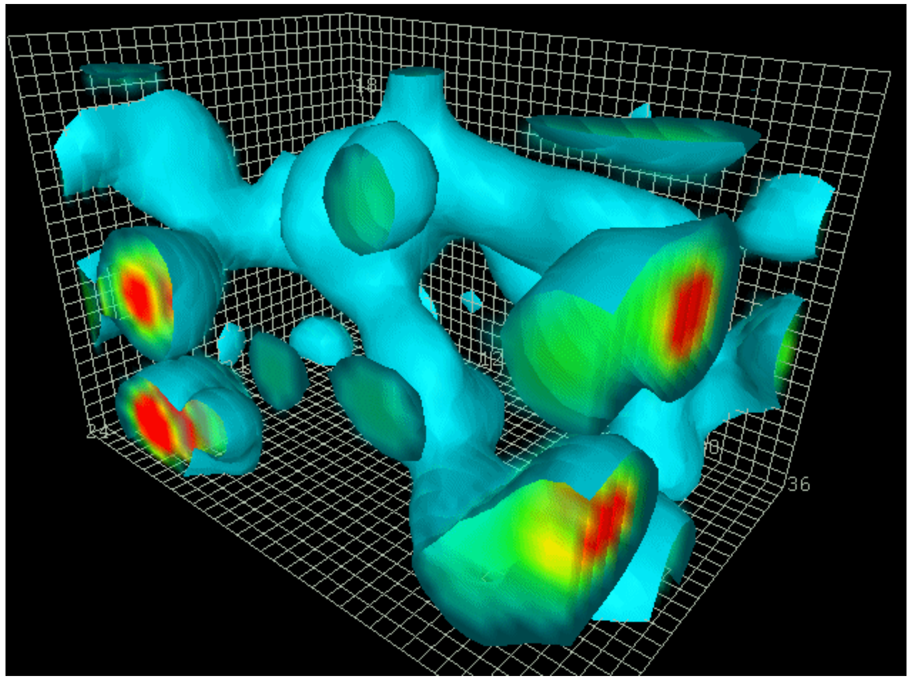
Background of this work

Akio Tomiya

Machine learning for LQCD, LQCD with machine learning

- Our purpose of here is, realizing neural network with gauge/globally-symmetric covariance
  - improvement of efficiency is not current goal
- In this talk, we apply our method to generating configurations AS A WORKING EXAMPLE
- Here we introduce two Transformers for spin-system and gauge theory
- No physics but algorithm to realize symmetry covariant neural nets



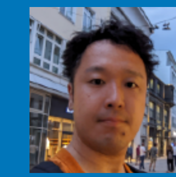


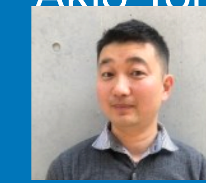
http://www.physics.adelaide.edu.au/theory/staff/leinweber/VisualQCD/QCDvacuum/

Lattice QCD code for generic purpose

Open source LQCD code in Julia Language

Akio Tomiya
AT & Y. Nagai (A01, A03)





LatticeQCD.jl

Open source (Julia Official package), Easy as Python and Fast as a fortran code -> Best for R&D purpose

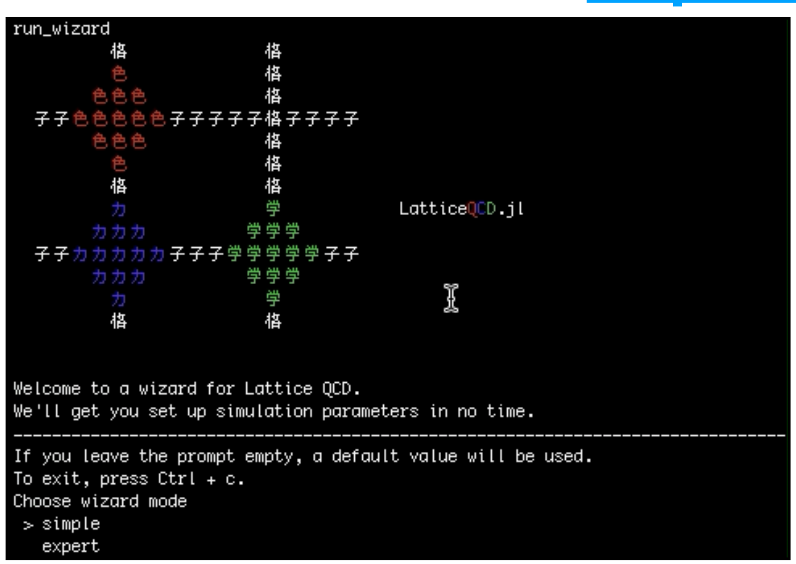
Machines: Laptop/desktop/Jupyter/Supercomputers

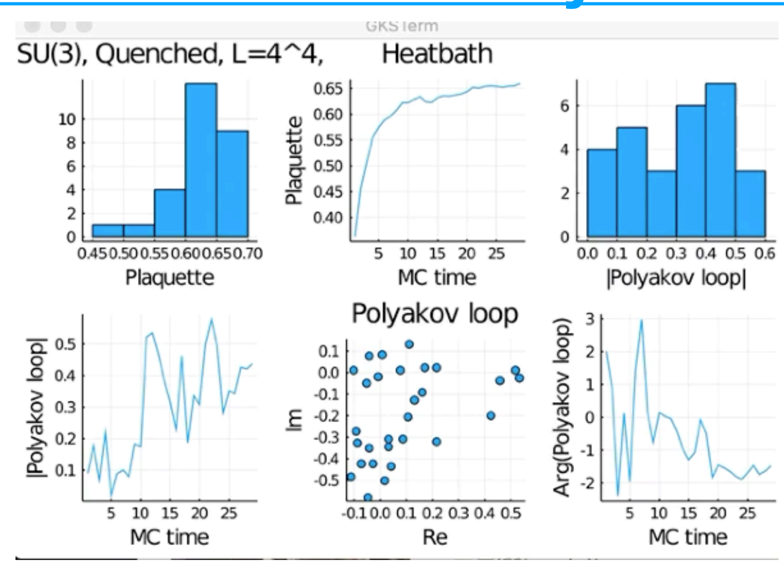
Functions: SU(Nc)-heatbath, RHMC, Self-learning HMC, SU(Nc) Stout
Dynamical Staggered, Dynamical Wilson, Dynamical Domain-wall
Measurements

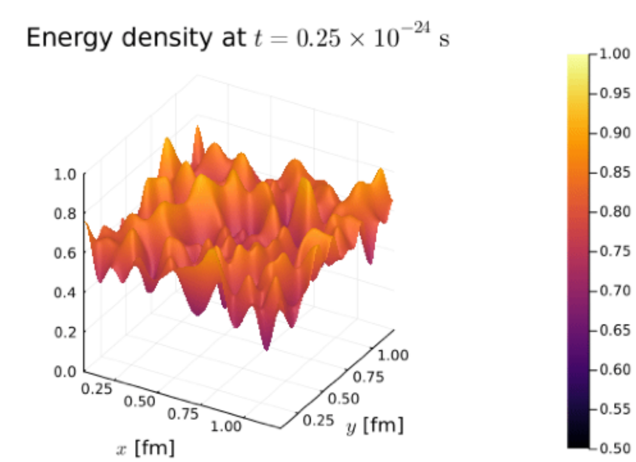
Start LQCD in **5 min**

1. Download Julia binary
2. Add the package through Julia package manager
3. Execute! (without explicit compiling)

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



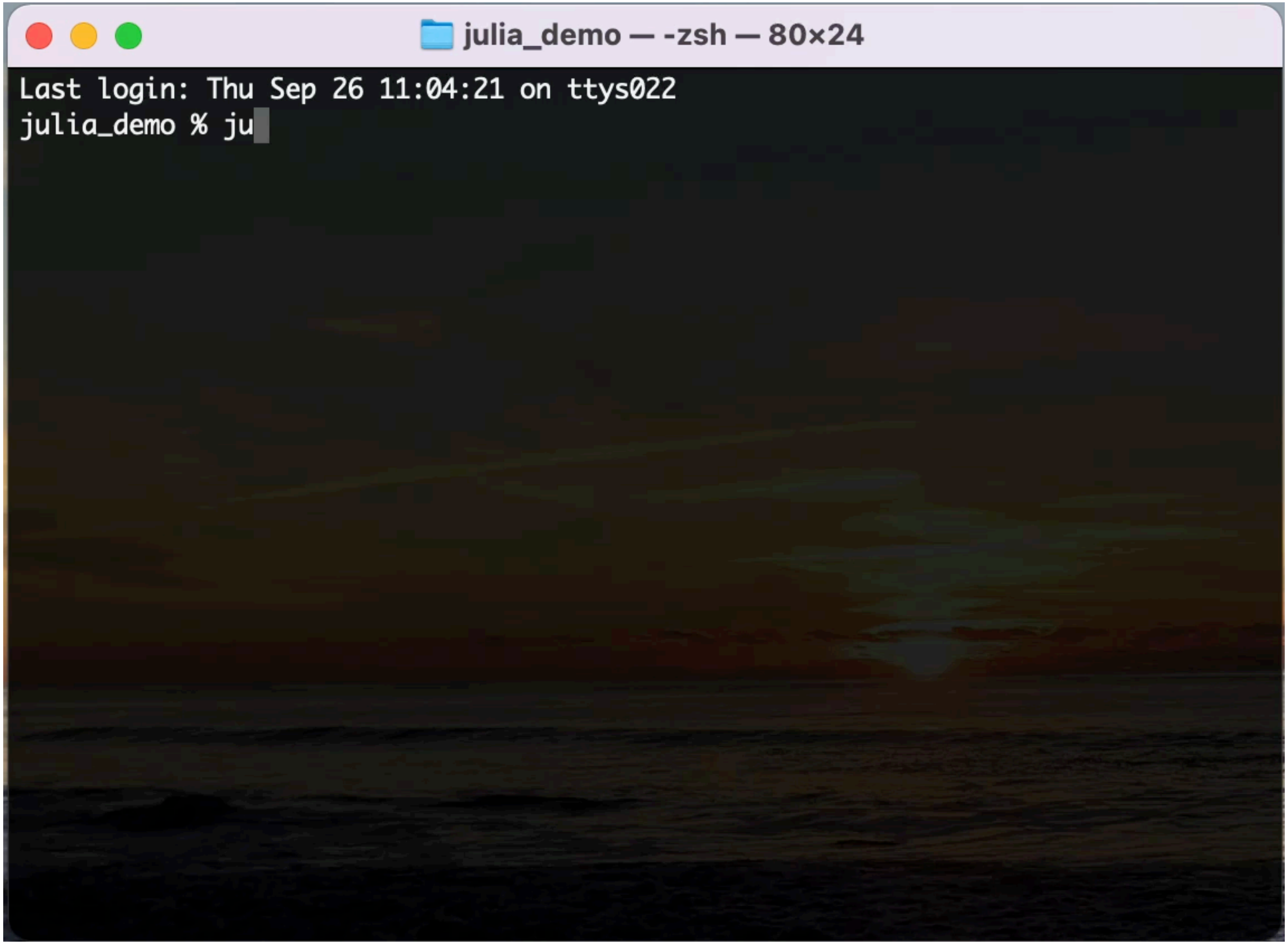




Minimize time for code development + actual calculations

Demo

Video: https://youtu.be/Z-CT8A2R_-w



Install, parameter file wizard
run Full QCD(Wilson) HMC, pion correlator

Lattice QCD code

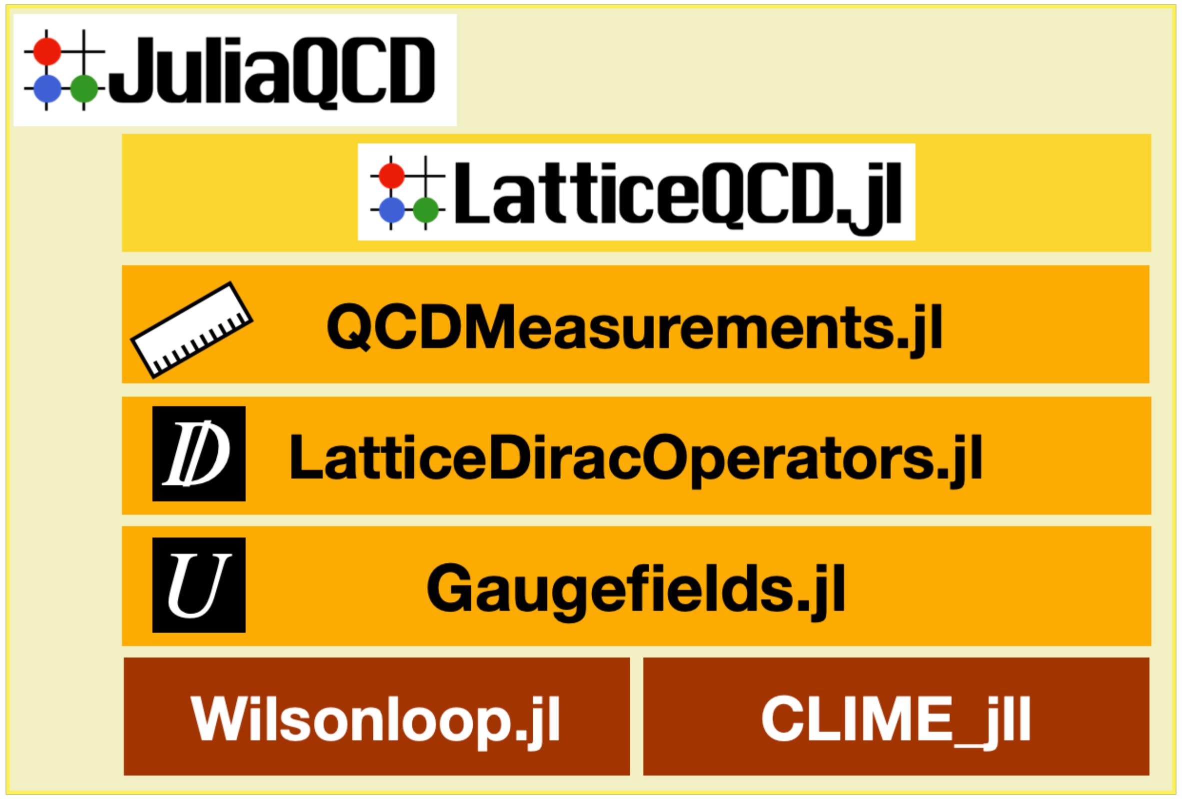
JuliaQCD: Open source LQCD code project

Akio Tomiya

AT & Y. Nagai

https://github.com/JuliaQCD

https://arxiv.org/abs/2409.03030

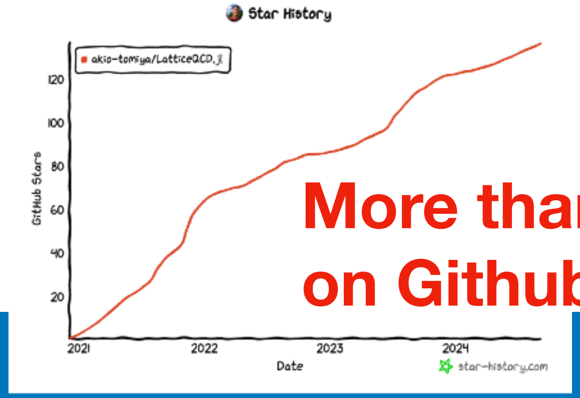


- **LatticeQCD.jl:** Wrapper of following package (※)
  - Easy to start, Suite
- **QCDMeasurements.jl:** Chiral Cond., Pion-propagator, Wilson loop etc MPI
- **LatticeDiracOperators.jl:** Lattice fermions (Wilson、Staggered, DW) and solvers

  MPI
- **Gaugefields.jl:** SU(N) gauge fields and action, gradient flow. Zn gauge fields are now supported (※※). Auto-grad (automatic derivartive for force and ML) MPI
- **Wilsonloop.jl:** Symbolic definition of Wilson loops and lines. They are converted to product of links
- **CLIME:** wrapping Clime. To treat ILDG format conf

※ LatticeDiracOperators.jl & Gaugefields.jl can be executed without LatticeQCD.jl. See github page

※※ Thanks to O. Morikawa



More than 100 stars on Github

Contributions are very welcome!

Applications on LQCD

Machine learning for lattice QCD

1. Transformer for O(3) spin model
2. CASK: Gauge symmetric transformer

Transformer for O(3) spin model

Akio Tomiya

Target: Double exchange model

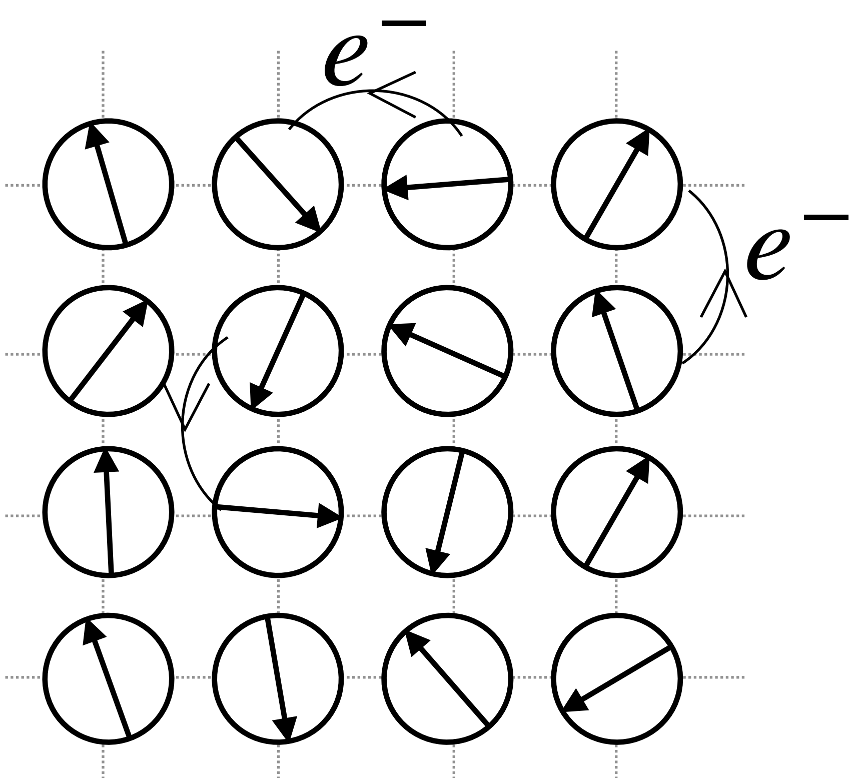
Target system: Classical Heisenberg spin $\mathbf{S}_i$+ Fermion on 2d lattice

$$H = -t \sum_{\alpha,\langle i,j\rangle} (\hat{c}^\dagger_{i\alpha}\hat{c}_{j\alpha} + \mathrm{h.c.}) + \frac{J}{2}\sum_i \mathbf{S}_i \cdot \hat{\sigma}_i \qquad \text{(Kondo model)}$$

Fermion

Current of c

Hensenberg spin



3d vectors on 2d lattice
Anti-ferro magnet

Two different phases
- **Anti-ferromagnet (~staggered mag)**
- **Paramagnet (~normal metal)**

(This system is similar to lattice QCD but easier)

Transformer for O(3) spin model

Akio Tomiya

Previous work

https://arxiv.org/abs/2005.06992

Target system: Classical Heisenberg spin $\mathbf{S}_i$ + Fermion on 2d lattice

$$H = -t \sum_{\alpha,\langle i,j\rangle} (\hat{c}^\dagger_{i\alpha}\hat{c}_{j\alpha} + \mathrm{h.c.}) + \frac{J}{2}\sum_i \mathbf{S}_i \cdot \hat{\sigma}_i$$

(Kondo model)

Naive effective model:

$$H^{\mathrm{Linear}}_{\mathrm{eff}} = -\sum_{\langle i,j\rangle_n} J^{\mathrm{eff}}_n \mathbf{S}_i \cdot \mathbf{S}_j + E_0$$

J^{eff}_n: n-th nearest neighbor

J^{eff}_n is determined by regression (training) to improve approximation

Self-learning Monte-Carlo:

Update with H_{eff} and Metropolis-Hastings with H & H_{eff}

H_{eff} has tunable parameters (couplings), which will be tuned.

Cancel in-exactness by MH-test,.This is an exact algorithms

Self-learning Monte-Carlo

Akio Tomiya

SLMC = MCMC with an effective model

arXiv:1610.03137+

For statistical spin system, we want to calculate expectation value with

$$W(\{\mathbf{S}\}) \propto \exp[-\beta H(\{\mathbf{S}\})]$$

On the other hand, an effective model $H_{\text{eff}}(\{\mathbf{S}\})$ can compose in MCMC,

$\{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\}$ this distributes $W_{\text{eff}}(\{\mathbf{S}\}) \propto \exp[-\beta H_{\text{eff}}(\{\mathbf{S}\})]$

if the update「→」 satisfies the detailed balance. We can employ Metropolis test like

$$A_{\text{eff}}(\{\mathbf{S}'\},\{\mathbf{S}\}) = \min\left(1, W_{\text{eff}}(\{\mathbf{S}'\})/W_{\text{eff}}(\{\mathbf{S}\})\right).$$

SLMC: Self-learning Monte-Carlo

We can construct *double* MCMC with $H(\{\mathbf{S}\})$ and $H_{\text{eff}}(\{\mathbf{S}\})$

$$\{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\} \longrightarrow \{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\} \longrightarrow$$

with Metropolis-*Hastings* test: $A(\{\mathbf{S}'\},\{\mathbf{S}\}) = \min\left(1, \frac{W(\{\mathbf{S}'\})}{W(\{\mathbf{S}\})}\frac{W_{\text{eff}}(\{\mathbf{S}\})}{W_{\text{eff}}(\{\mathbf{S}'\})}\right).$

- **Effective model can have fit parameters**
- **Exact! It satisfies detailed balance with** $W(\{\mathbf{S}\})$
- **It has been used for full QCD too (arXiv: 2010.11900, 2103.11965)**

Transformer for O(3) spin model

Akio Tomiya

Block spin transformation using neural net

Target system: Classical Heisenberg spin S_i+ Fermion on 2d lattice

$$H = -t \sum_{\alpha,\langle i,j \rangle} (\hat{c}^\dagger_{i\alpha} \hat{c}_{j\alpha} + \mathrm{h.c.}) + \frac{J}{2} \sum_i \mathbf{S}_i \cdot \hat{\sigma}_i$$

(Kondo model)

Naive effective model:

$$H^{\mathrm{Linear}}_{\mathrm{eff}} = -\sum_{\langle i,j \rangle_n} J^{\mathrm{eff}}_n \mathbf{S}_i \cdot \mathbf{S}_j + E_0$$

J^{eff}_n: n-th nearest neighbor

We replace this by "translated" spin S^{NN}_i with a transformer and used in self-learning MC

$$H_{\mathrm{eff}} = -\sum_{\langle i,j \rangle_n} J^{\mathrm{eff}}_n \mathbf{S}^{\mathrm{NN}}_i \cdot \mathbf{S}^{\mathrm{NN}}_j + E_0$$

arXiv: 2306.11527.

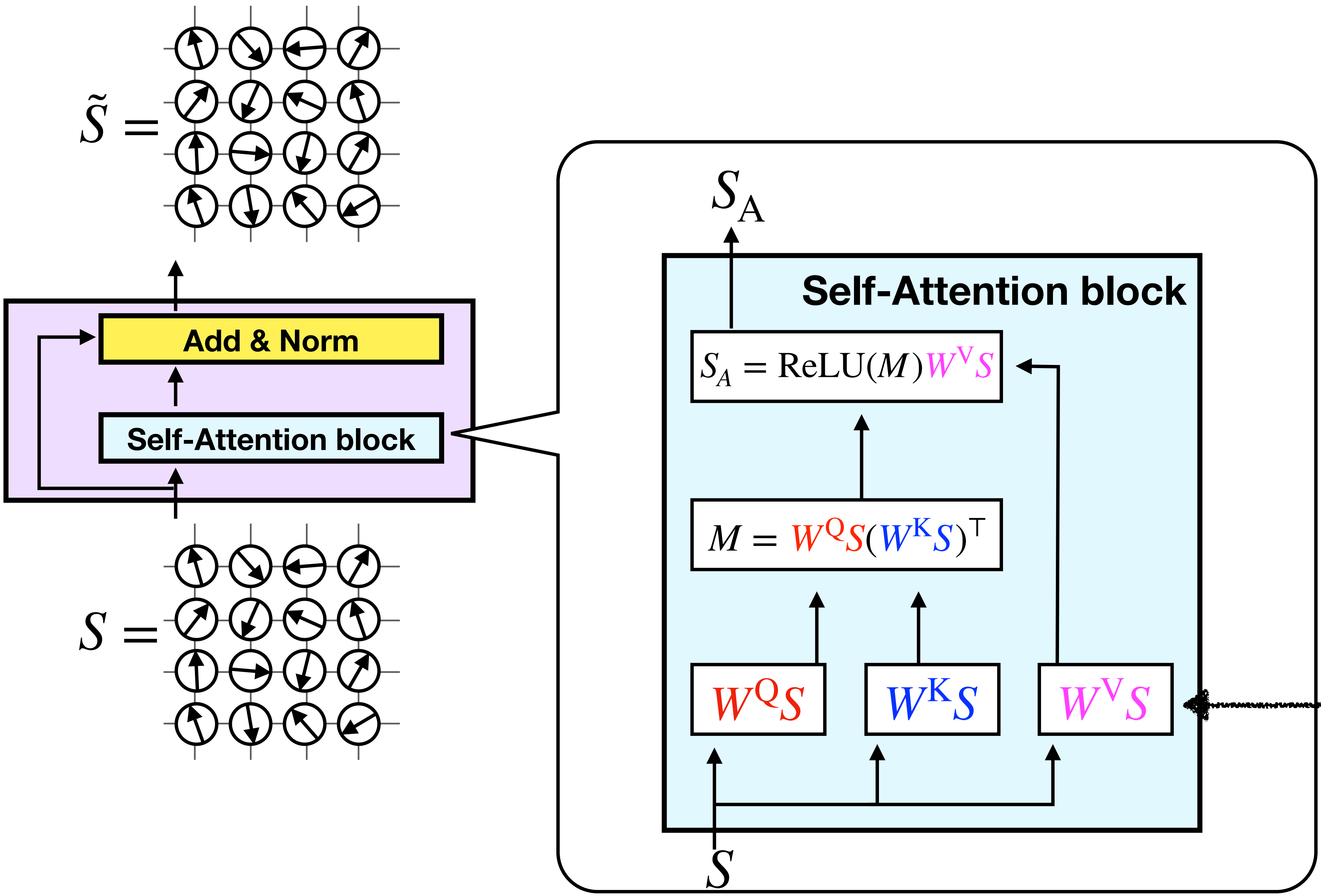
Equivariant attention

Self-learning Monte-Carlo

Akio Tomiya

arXiv: 2306.11527.

Attention block makes effective spin field with non-local BST



Smearing (BST)
Rot. equivariant
Trsl. equivariant
trainable!

Self-learning Monte-Carlo

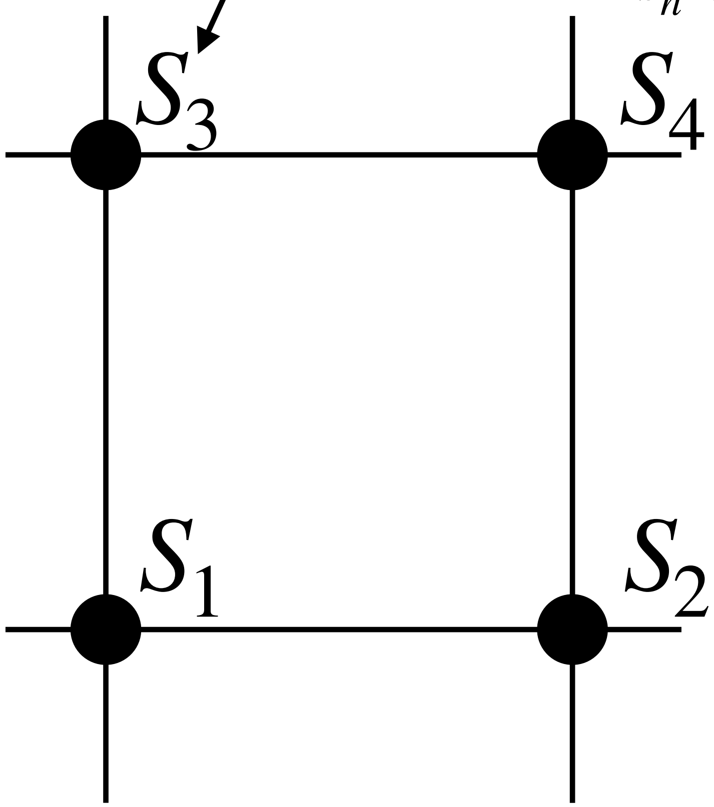
Equivariant under spin-rotation & translation

Akio Tomiya

arXiv: 2306.11527.

$S_3^\top = \begin{pmatrix} s_3^1 & s_3^2 & s_3^3 \end{pmatrix}^\top$

s_n^μ : O(3) vector



$$\mathbf{S} = \begin{pmatrix} S_1^\top & S_2^\top & S_3^\top & S_4^\top \end{pmatrix}^\top$$

- **Local weighted sum over neighbors**
  = "Smeared spin" with parameters
  ~ "Block spin sum" with parameters

$$\tilde{S}_i^\alpha = \sum_{l=0} w_l^\alpha S_{i+l} \qquad \alpha = \mathrm{Q}, \mathrm{K}, \mathrm{V}$$

$w_l^\alpha \in \mathbb{R}$: trainable

Translationally equivariant
Rotationally equivariant

$$S_i^\top = \begin{pmatrix} s_i^1 & s_i^2 & s_i^3 \end{pmatrix}^\top$$

$$|S_i| = \sqrt{(s_i^1)^2 + (s_i^2)^2 + (s_i^3)^2}$$
$$= 1$$

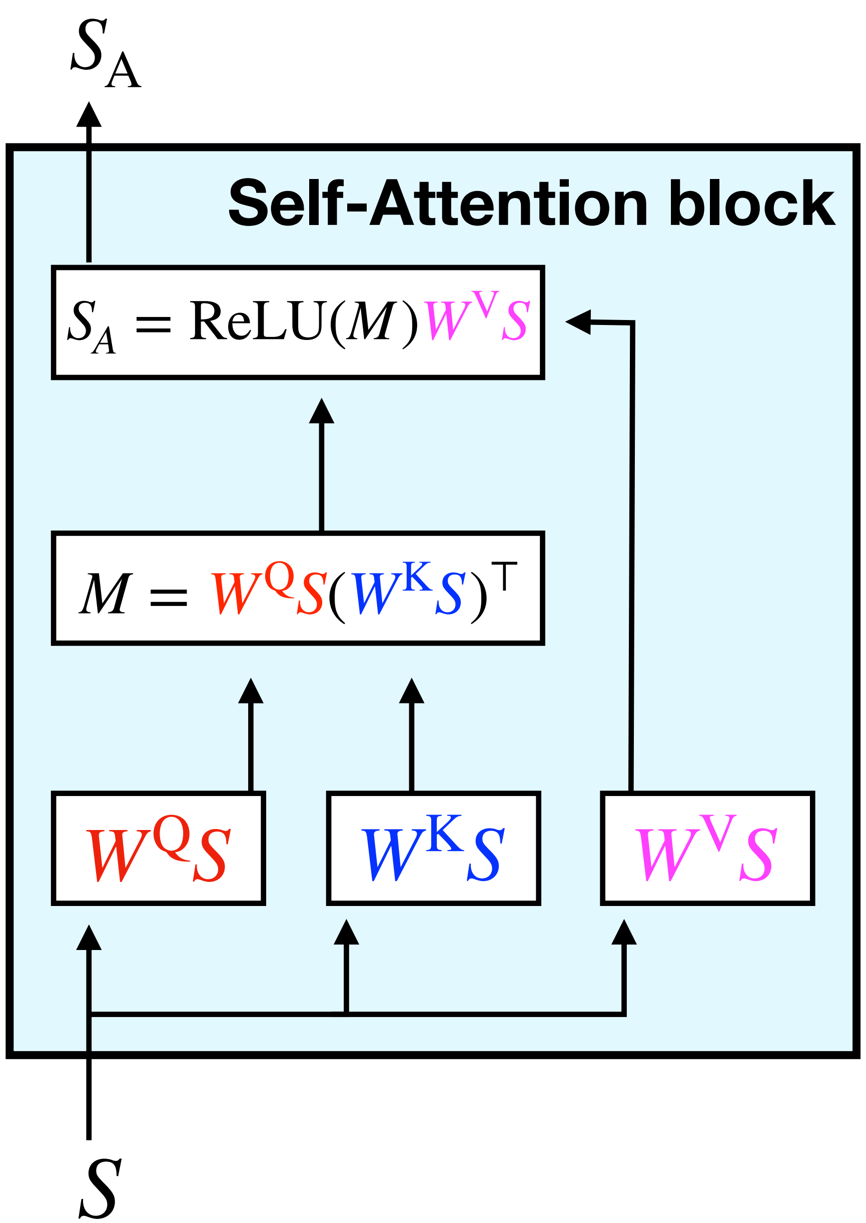
3 component scalar, normalized

Self-learning Monte-Carlo

Akio Tomiya

Equivariant under spin-rotation & translation

arXiv: 2306.11527.



$$\mathbf{S} = \begin{pmatrix} S_1^{\top} & S_2^{\top} & S_3^{\top} & S_4^{\top} \end{pmatrix}^{\top}$$

$$S_i^{\top} = \begin{pmatrix} s_i^1 & s_i^2 & s_i^3 \end{pmatrix}^{\top}$$

s_n^{μ} : O(3) vector

$$\tilde{S}_i^{\alpha} = W^{\alpha} S = \sum w_l^{\alpha} S_{i+l}$$

"averaged spin" by neighbors

Gram matrix with averaged spin

$$M = \tilde{G}^{\alpha} \equiv (\tilde{\mathbf{S}}^{\alpha})^{\top} \tilde{\mathbf{S}}^{\alpha} \quad \alpha = \mathrm{Q}, \mathrm{K}, \mathrm{V}$$

$$G \equiv \mathbf{S}^{\top}\mathbf{S} = \begin{pmatrix} S_1^{\top} S_1 & S_1^{\top} S_2 & S_1^{\top} S_3 & S_1^{\top} S_4 \\ S_2^{\top} S_1 & S_2^{\top} S_2 & S_2^{\top} S_3 & S_2^{\top} S_4 \\ S_3^{\top} S_1 & S_3^{\top} S_2 & S_3^{\top} S_3 & S_3^{\top} S_4 \\ S_4^{\top} S_1 & S_4^{\top} S_2 & S_4^{\top} S_3 & S_4^{\top} S_4 \end{pmatrix}$$

Translationally covariant, Rotationally invariant

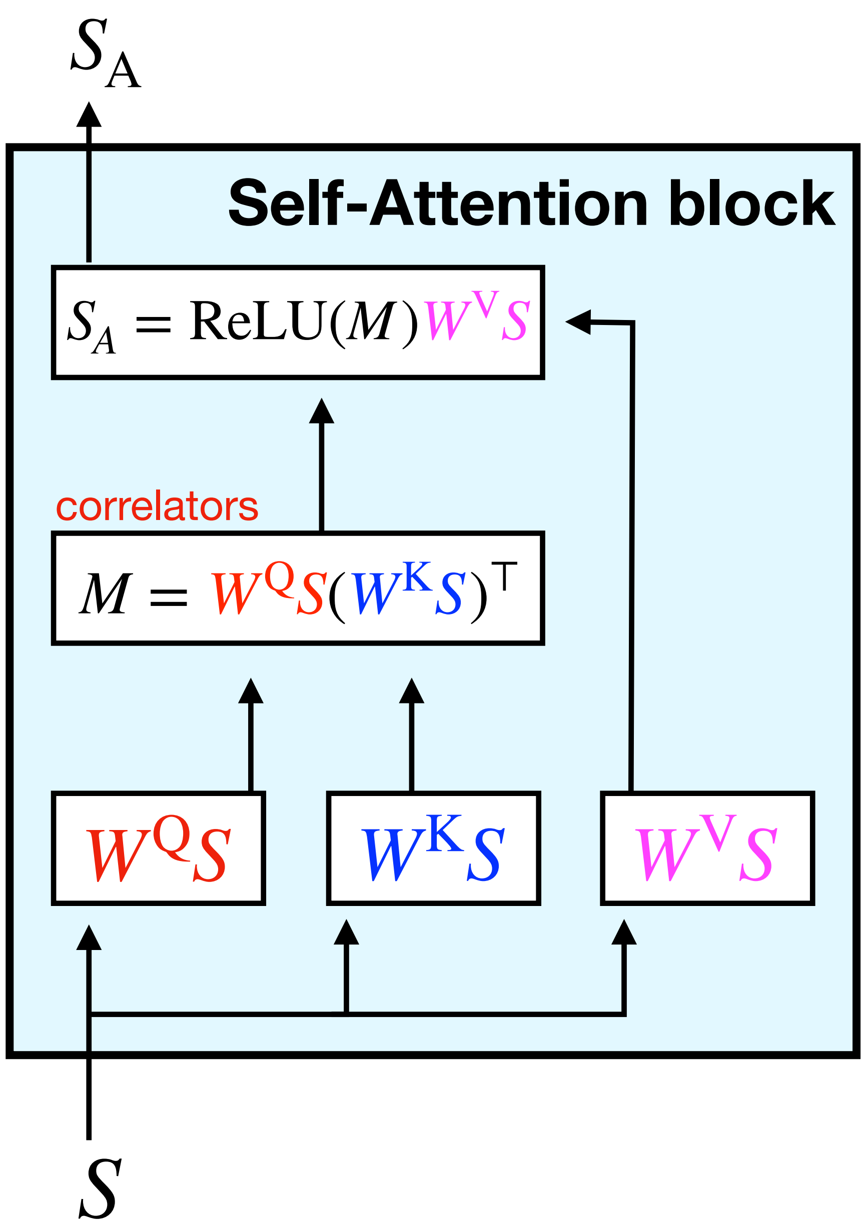
A set of correlators

Self-learning Monte-Carlo

Equivariant under spin-rotation & translation

Akio Tomiya

arXiv: 2306.11527.



$$\mathbf{S} = \begin{pmatrix} S_1^{\top} & S_2^{\top} & S_3^{\top} & S_4^{\top} \end{pmatrix}^{\top}$$

$$S_i^{\top} = \begin{pmatrix} s_i^1 & s_i^2 & s_i^3 \end{pmatrix}^{\top}$$

s_n^{μ} : O(3) vector

$$\tilde{S}_i^{\alpha} = W^{\alpha} S = \sum w_l^{\alpha} S_{i+l}$$

"averaged spin" by neighbors

Gram matrix with averaged spin

$$M = \tilde{G}^{\alpha} \equiv (\tilde{\mathbf{S}}^{\alpha})^{\top} \tilde{\mathbf{S}}^{\alpha} \quad \alpha = \mathrm{Q}, \mathrm{K}, \mathrm{V}$$

Translationally covariant
Rotationally invariant

$$S_A = \mathrm{ReLU}(M) W^{\mathrm{V}} S$$
$$= \mathrm{ReLU}(M) \tilde{S}^{\mathrm{V}}$$

A set of correlators

Akio Tomiya

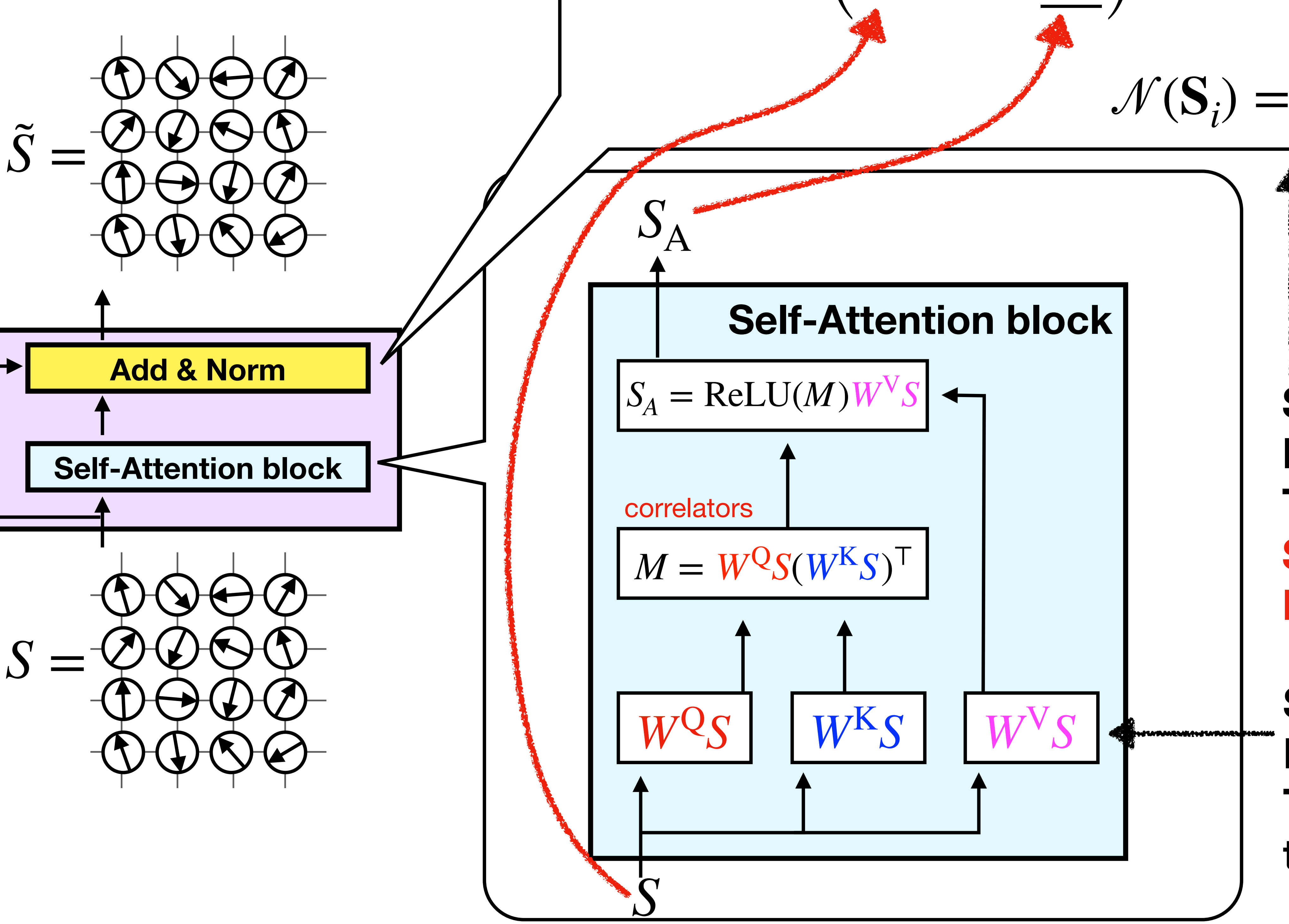
arXiv: 2306.11527.

Self-learning Monte-Carlo

Attention block makes effective spin field with non-local BST

$$\mathbf{S}^{(l)} \equiv \mathcal{N}\left(\mathbf{S}^{(l-1)} + \underline{\mathbf{S}_{\mathrm{A}}}\right)$$ position-wise

$$\mathcal{N}(\mathbf{S}_i) = \mathbf{S}_i / \|\mathbf{S}_i\|$$



Smeared fields
Rot. equivariant
Trsl. equivariant
Skip connection
Normalized!

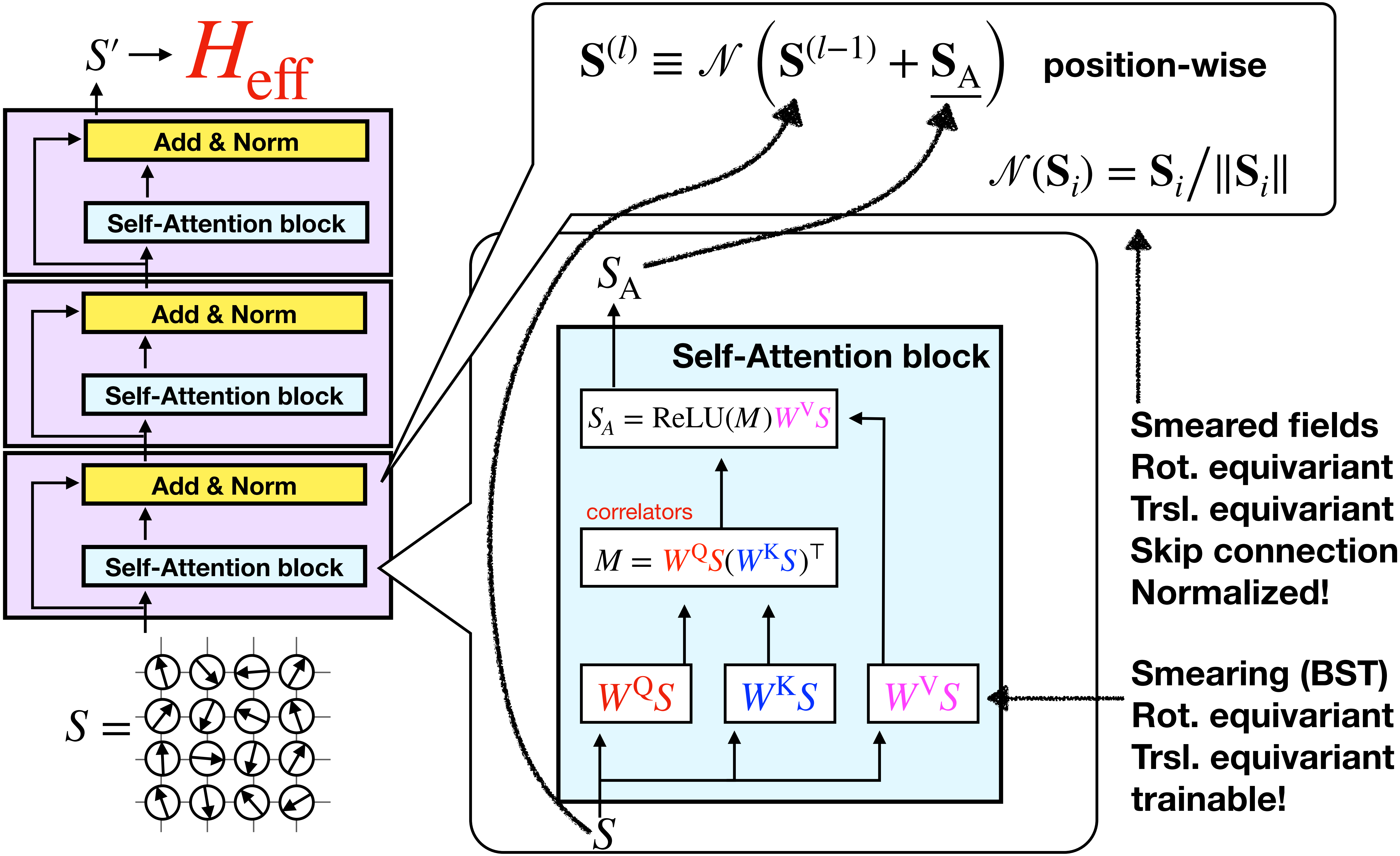
Smearing (BST)
Rot. equivariant
Trsl. equivariant
trainable!

Akio Tomiya

arXiv: 2306.11527.

Self-learning Monte-Carlo

Variational Hamiltonian with Equivariant Attention layers

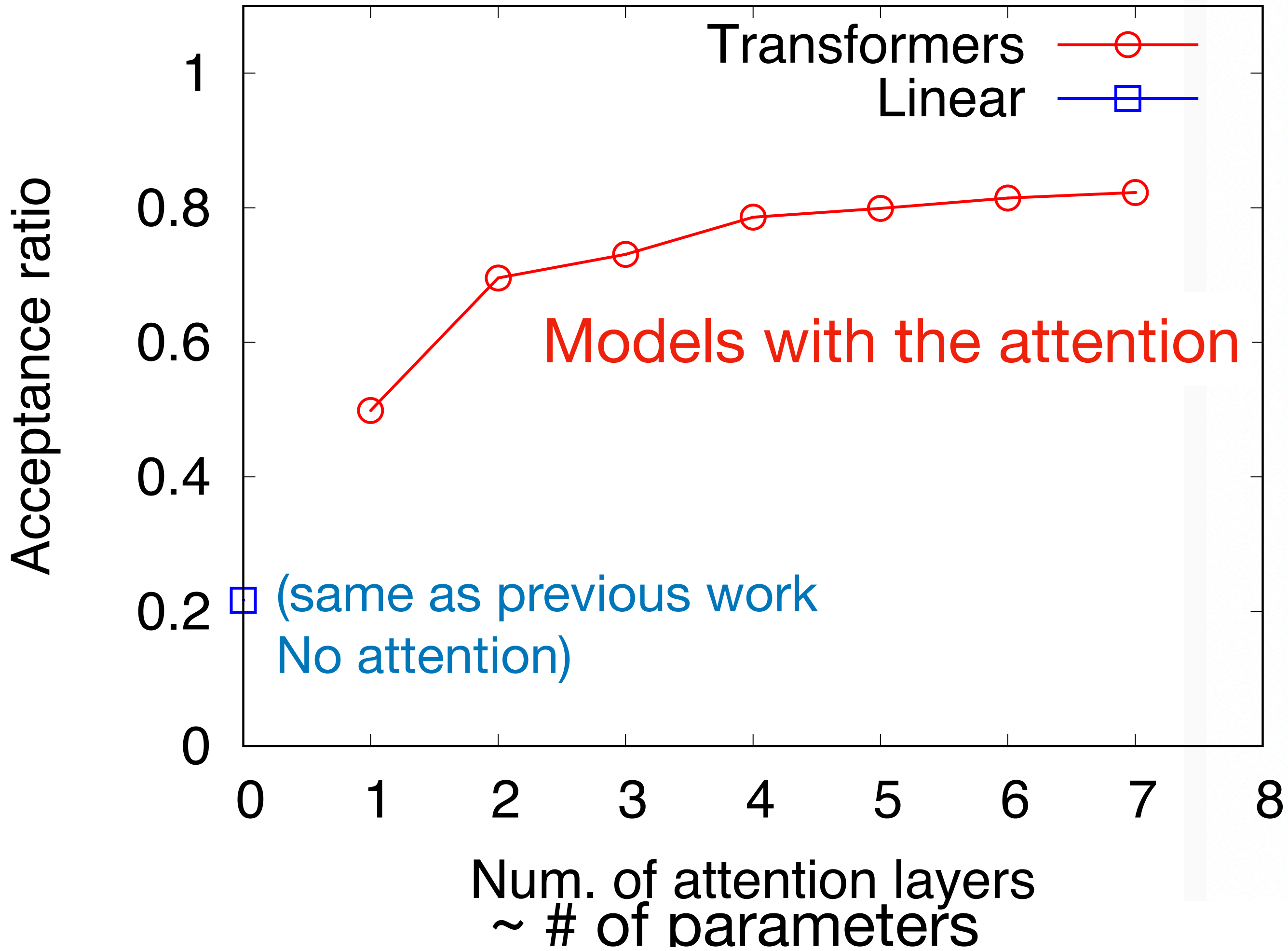


Transformer and Attention

Akio Tomiya
arXiv: 2306.11527 + update

Application to O(3) spin model with fermions

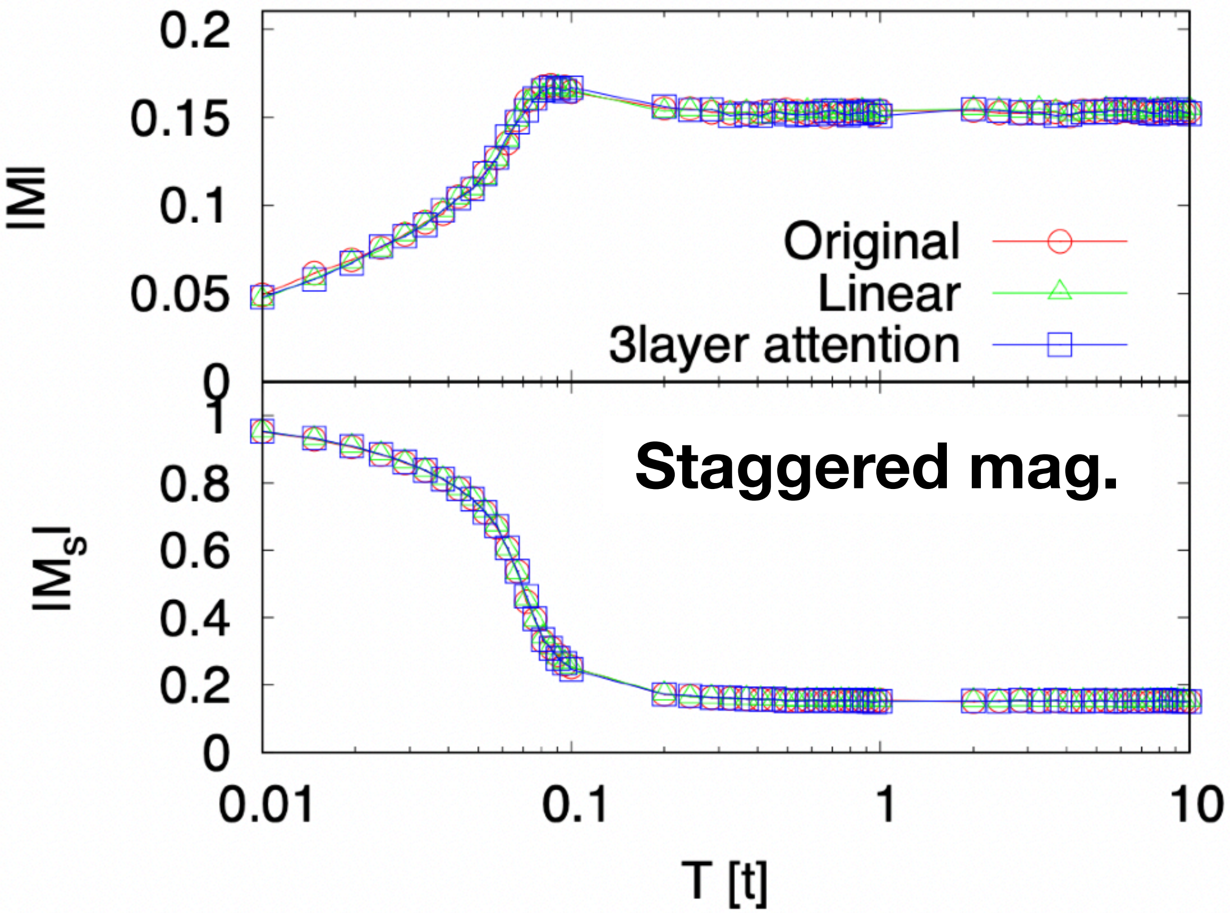
Acceptance rate ~ efficiency



Note: As far as we tested,
CNN-type does not work in this case.

No improvements with increase of layers.
(Global correlations of fermions from
Fermi-Dirac statistics make acceptance bad?)

Observables



Physical values are consistent (as we expected)

julia

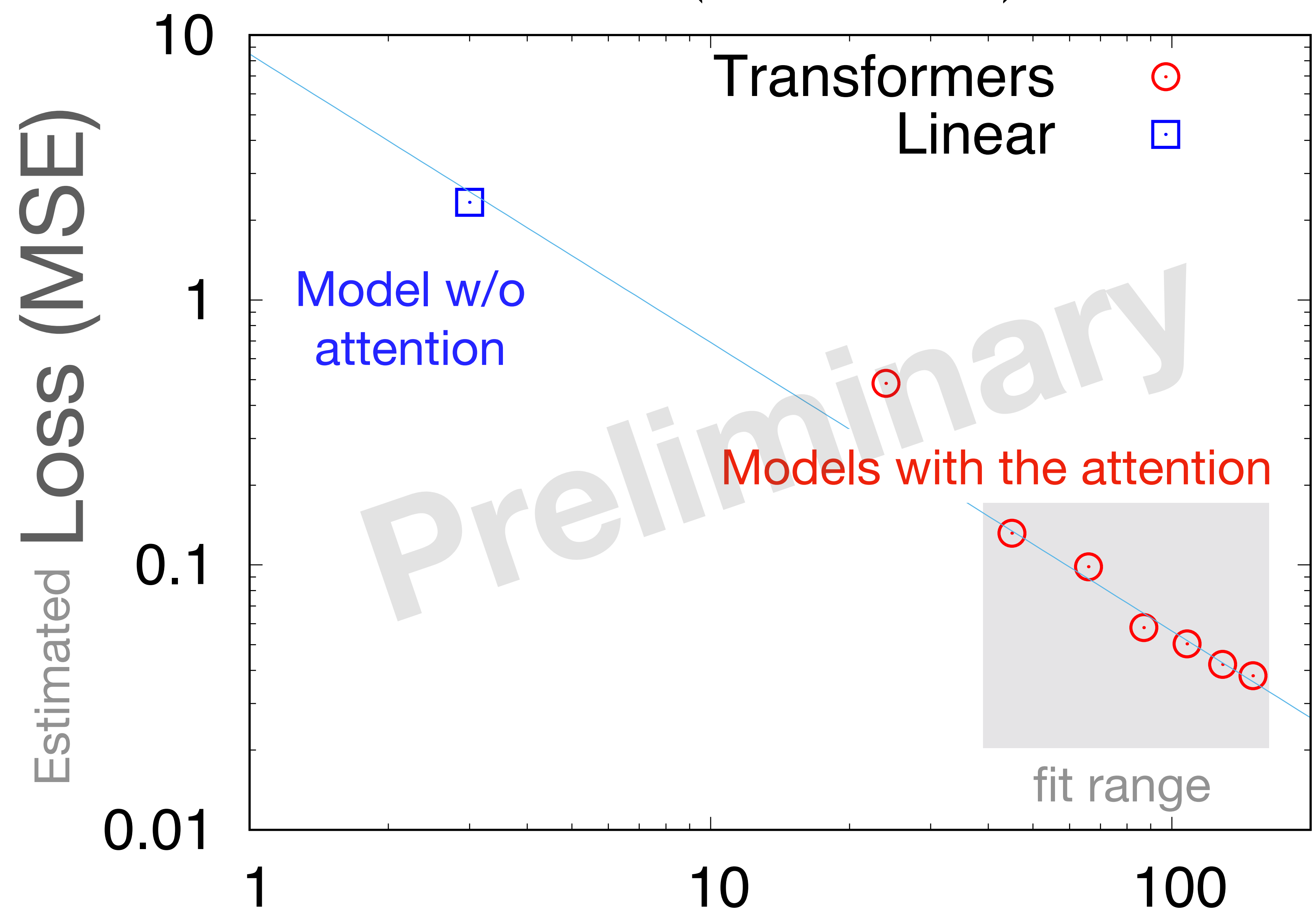
Nx=Ny=6
(Lattice sites)

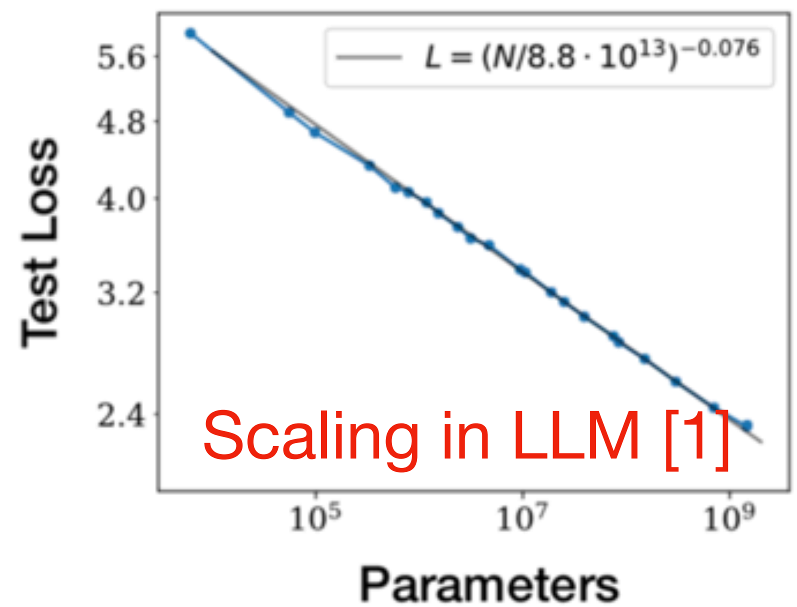
Transformer and Attention

Loss function shows Power-type scaling law as LLM

arXiv: 2306.11527 + update

$$\text{Acceptance rate} = \exp\left(-\sqrt{\text{MSE}}\right)$$





Line is just for guiding eyes (no meaning)

fit ~(7.1/x)^(1.1)

[1] arXiv:2001.08361

Applications on LQCD

Machine learning for lattice QCD

1. Transformer for O(3) spin model
2. CASK: Gauge symmetric transformer

Previous work of CASK

Akio Tomiya
AT Y. Nagai arXiv: 2103.11965

Gauge cov net= trainable smearing (= residual flow)

Stout-type covariant net

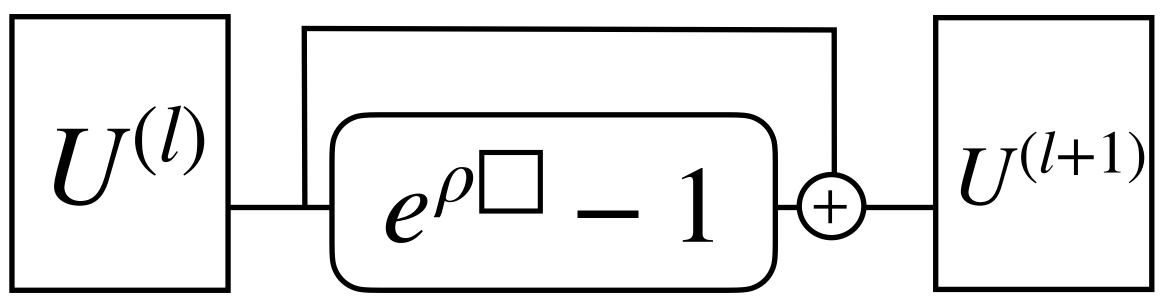
$$U_\mu(n) \to U_\mu^{\mathrm{smr}}(n) = \mathrm{e}^{\sum_i \rho_i L_i[U]} U_\mu(n)$$

staple

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

Trainable param

Training done by the back-prop
(extension to the stout paper [1])



It is gauge covariant variational function for gauge field

Pros 😀: Gauge/translational covariant

Cons 😢: It process data as same as convolution, it is local (not efficient)

There are several realization of gauge covariant maps arXiv:2012.12901 arXiv: 2305.02402

[1] C. Morningster+ 2003

Configuration generation in LQCD

Akio Tomiya

CASK?

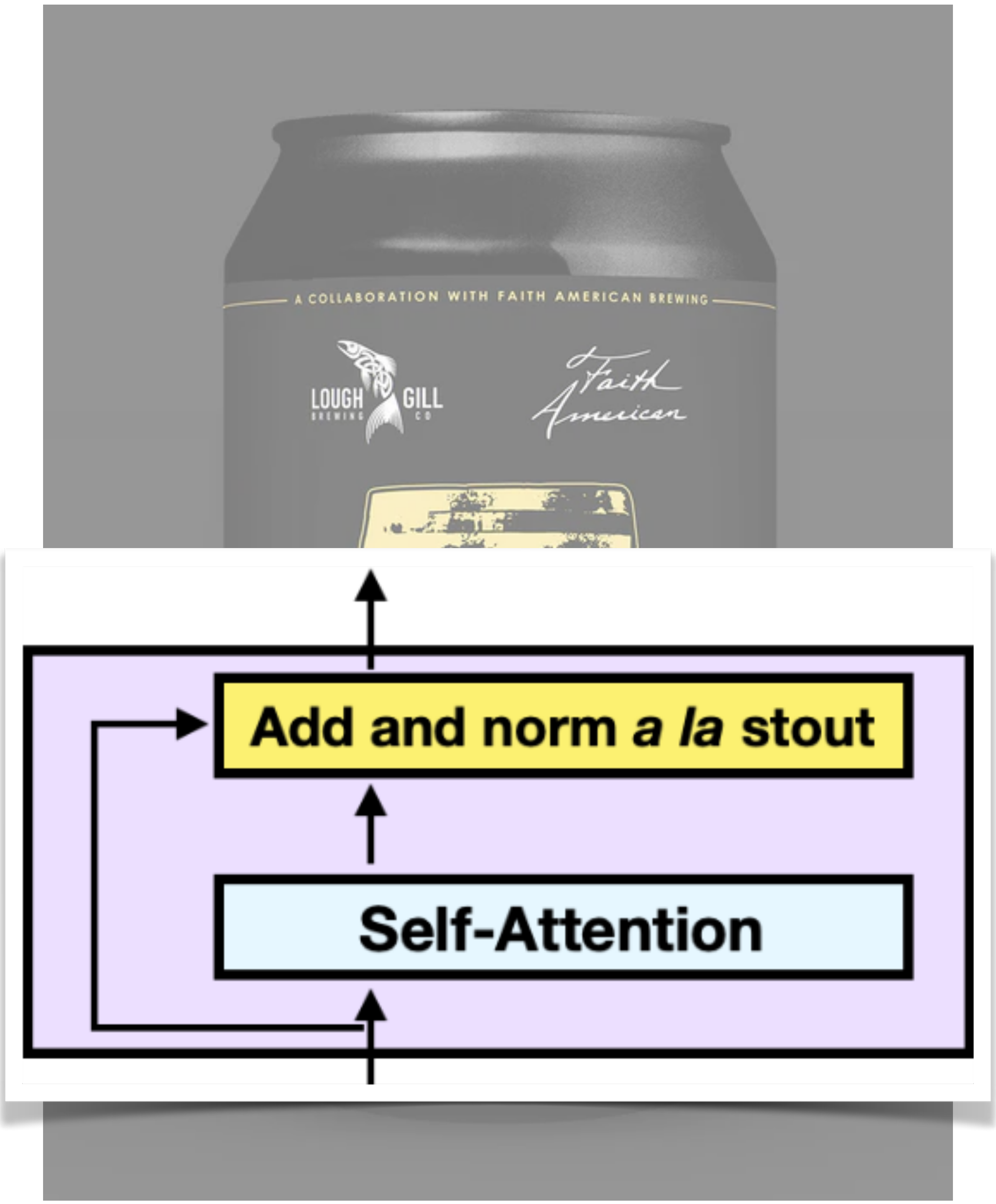
Cask stout
(Whisky Barrel-Aged Stout beer)
= stout beer in a cask

https://loughgillbrewery.com/products/cask-irish-coffee-stout

Configuration generation in LQCD

Akio Tomiya

CASK = Stout kernel, gauge covariant transformer for LQCD



Cask stout
(Whisky Barrel-Aged Stout beer)
= stout beer in a cask

Covariant attention block
CASK = Covariant Attention
with Stout Kernel

It is named in an obvious reason😜

Configuration generation in LQCD

Collection of ML/LQCD

Akio Tomiya

Lattice	**ML(Framework)**	**ML+Lattice**
• Demon method (inverse MC) arXiv1508.04986 AT+ • Hopping parameter	Linear regression	Phys. Rev. D 107, 054501 AT+ Gauge inv. SLMC Trivializing with SD eq a la Luscher 2212.11387 AT+
Stout & Flow	CNN/Equivariant NN	Gauge covariant net 2021 AT+
(nothing. mean field?)	Transformer - GPT	• Global symmetric Transformer 2306.11527 AT+ • CASK (this talk)

Configuration generation in LQCD

Akio Tomiya

Idea: Attention must be invariant

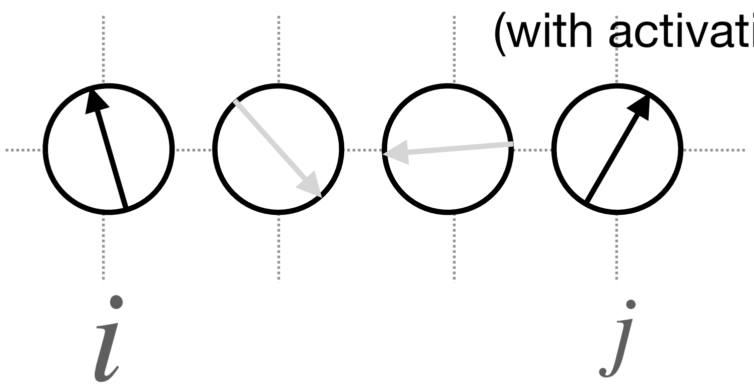
Attention matrix in transformer ~ correlation function (with block-spin transformed spin)

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

Transformer for Kondo spins

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

(with activation)



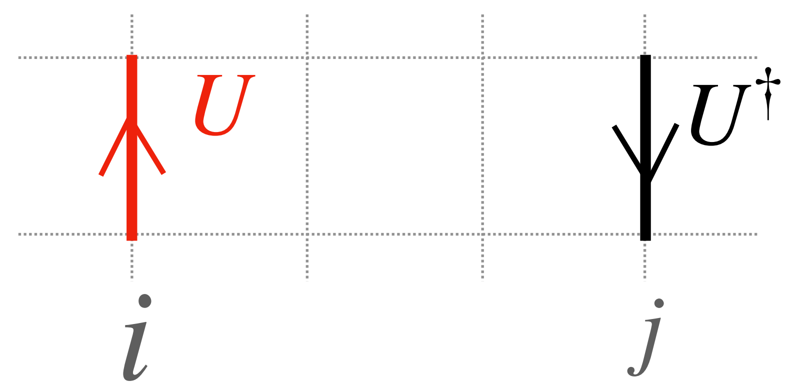
invariant
under global O(3)

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

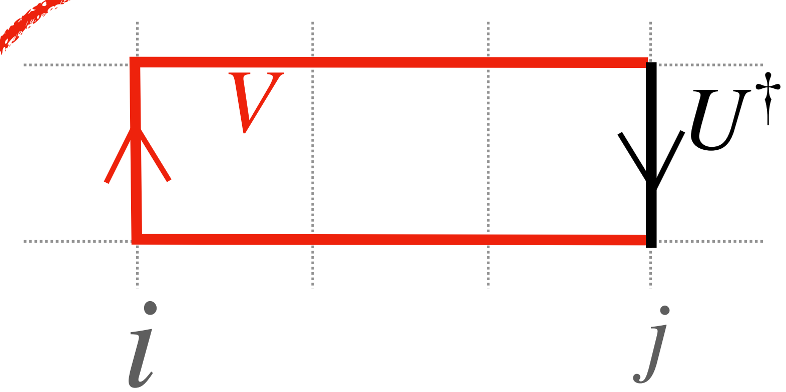
In total, output is covariant

2310.13222 AT+,2306.11527 AT+

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



not invariant
(cannot be used)



invariant under
local SU(N)

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

(with activation)

In total, output is covariant

WIP AT+

Configuration generation in LQCD

Akio Tomiya

Structure of gauge symmetric attention using stout

[1] 2021 AT+
WIP AT+

<u>Procedure in three steps:</u>

0. $U^{\rm in}$: Input configuration/Links

1. 3 types of (trainable) **stout** [1] -> $U^{\rm Q}$, $U^{\rm K}$, $U^{\rm V}$ (they have different weights)

Loop operator
projected on Lie algebra

$$U^{\alpha} = \exp[\rho^{\alpha} L[U^{\rm in}]] U^{\rm in} \qquad \alpha = {\rm Q, K, V}$$

weights

2. **Construct <u>attention matrix</u> (Extended Wilson loop)** using $U^{\rm Q}$, $U^{\rm K}$ -> $a_{(*,*)}$

$\sim a_{(*,*)}$ (with activation)

cf. sparse attention, star attention

3. Construct "**stout** smeared" [1] link with weight $a_{(*,*)}$ and $U^{\rm V}$, U (as matrix mult)

$$U^{\rm out} = \exp[a_{(*,*)} L[U^{\rm V}]] U^{\rm in} \qquad \text{Covariant}$$

(This can be extend to have multi-head trivially)

Loop operator
projected on Lie algebra

Configuration generation in LQCD

Physically symmetric Attention layer for LQCD

Akio Tomiya

[1] 2021 AT+
WIP AT+

Attention layer can capture global correlation
Equivariance reduces data demands for training

	Equivariance	**Gauge?**	**Capturable correlation**	**Data demmands**	**Applications**
Convolution (∈ equivariant layers)	Yes 👍	Yes 👍	Local 😲	Low 👍	VAE, GAN Normalizing flow SLHMC 2103.11965 AT+
Standard Attention layer arXiv:1706.03762	No 😲	No 😲	Global 👍	Huge 😲	ChatGPT GEMINI Vision Transformer
***Equivariant* attention for spin**	Yes 👍	No 😲	Global 👍	?	Kondo system (2310.13222 AT+ 2306.11527 AT+)
***Equivariant* attention for gauge**	Yes 👍	Yes 👍	Global 👍	?	WIP AT+

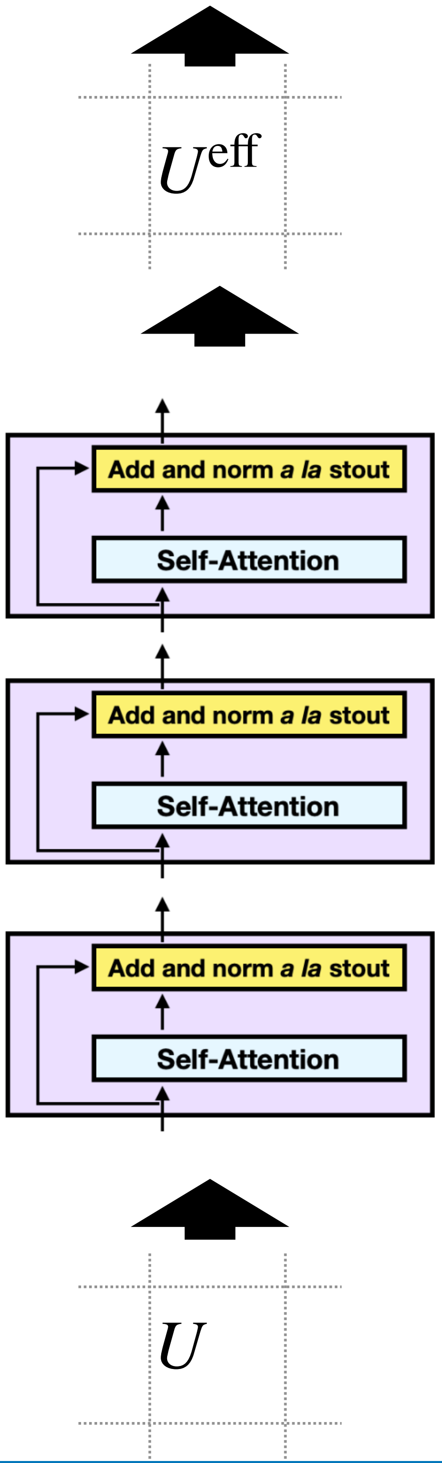
Configuration generation in LQCD

Akio Tomiya

Simulation parameter

WIP AT+

Construct effective action using operators with U^{eff}

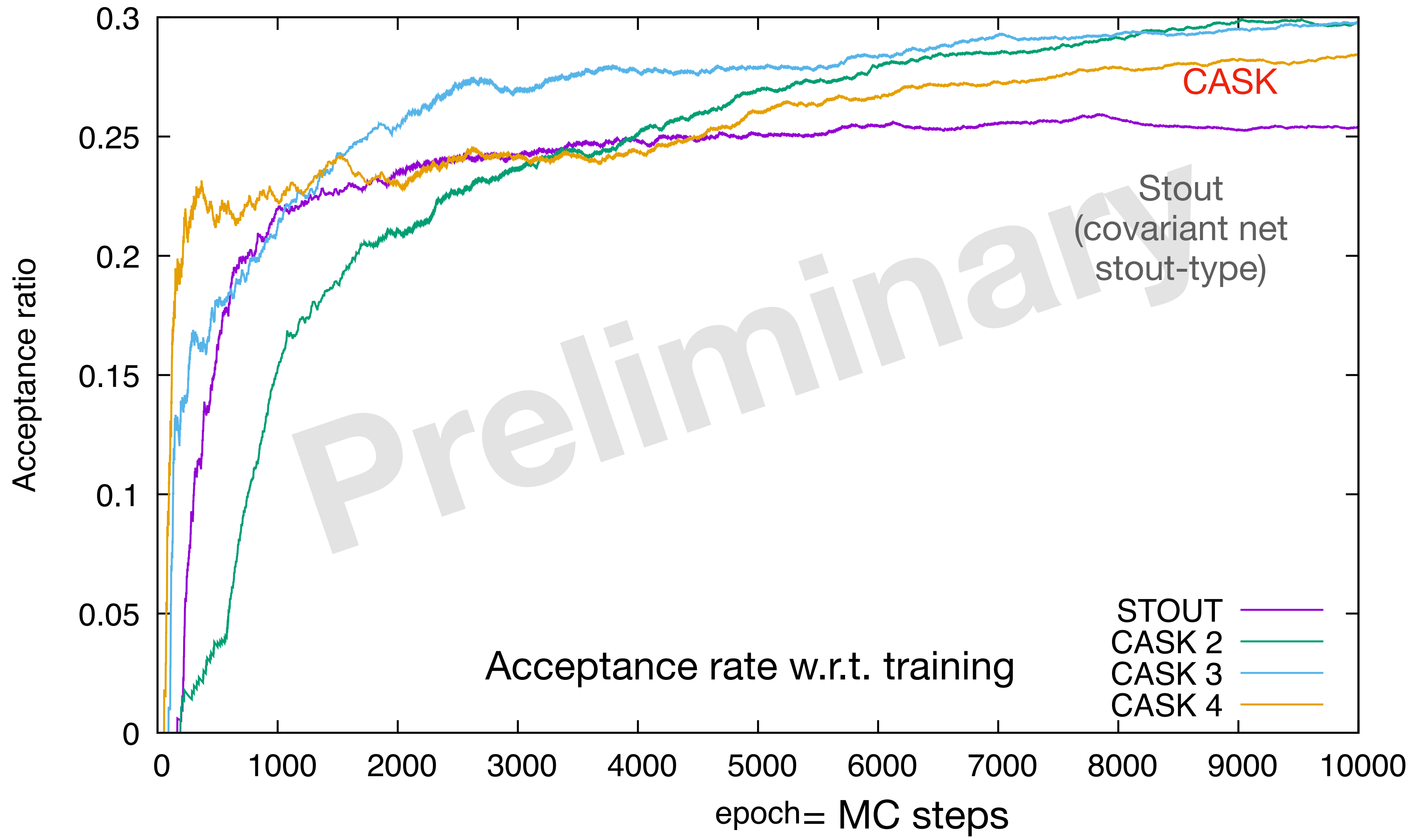


JuliaQCD

- Self-learning HMC (1909.02255, 2021 AT+)
  - Exact. Metropolis test and MD with effective action
- Target S : $m = 0.3$, dynamical staggered fermion, Nf=2, $L^4 = 4^4$, SU(2), $\beta = 2.7$. In Metropolis test
  - $M_{\text{target}} = D_{\text{stag}}[U] + m$
- Effective action S^{eff} in Molecular dynamics
  - Same gauge action
  - $m_{\text{eff}} = 0.4$ dynamical staggered fermion, Nf=2
    - Artificial example for mimicking different Dirac operator
  - CASK(smearing) with plaquette covariant kernel
    - Attention = 7-links rect staple (=3 plaquette)
  - MD uses $M_{\text{eff}} = D_{\text{stag}}[U^{\text{eff}}] + m^{\text{eff}}$
- It can be regarded as “Adaptively reweighted HMC”

Configuration generation in LQCD

Attention blocks improve acceptance

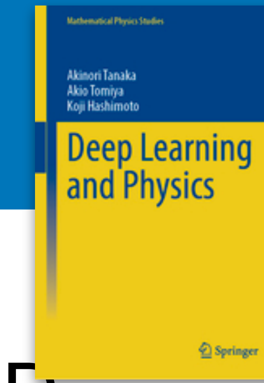


- In terms of acceptance, CASK has gain
    - Without trining, acceptance is zero. Training improves acceptance
    - After 5000 epoch, CASK is still improving
- Application? -> Future work

Summary

Machine learning + lattice field theory

Akio Tomiya



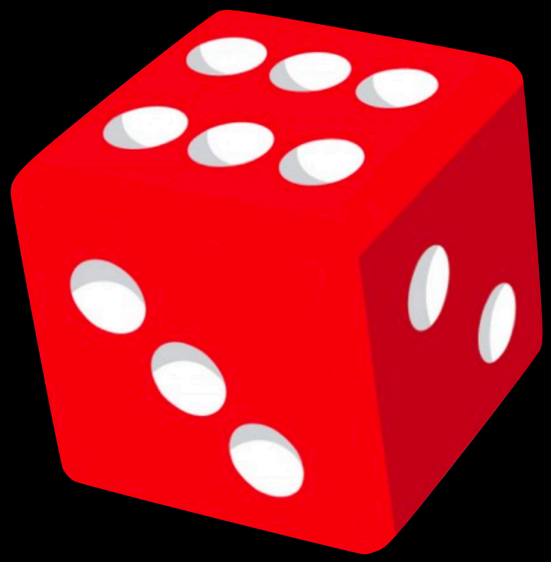
- Production and measurement need numerical cost
- Machine learning is useful for natural science/physics/Lattice QCD
  - Supervised learning requires data ahead of training
    - Self-learning does not require data (Self-learning HMC, flow based).
  - Gauge symmetry is now handled
  - The developed nets (transformers) works keeping symmetries
  - Apply to several generative NN approaches?
- Codes for LFT+ML are needed
  - Minimize code developing time + execution time
  - Maybe not only for machine learning, but also general R&D?
  - Julia might be good choice?
- Efficiency? We need more effort

julia

JuliaQCD

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
KAKENHI: 20K14479, 22H05112, 22H05111, 22K03539

Thanks!



Self-learning Monte-Carlo

Akio Tomiya

SLMC = MCMC with an effective model

arXiv:1610.03137+

For statistical spin system, we want to calculate expectation value with

$$W(\{\mathbf{S}\}) \propto \exp[-\beta H(\{\mathbf{S}\})]$$

On the other hand, an effective model $H_{\text{eff}}(\{\mathbf{S}\})$ can compose in MCMC,

$\{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\}$ this distributes $W_{\text{eff}}(\{\mathbf{S}\}) \propto \exp[-\beta H_{\text{eff}}(\{\mathbf{S}\})]$

if the update「→」 satisfies the detailed balance. We can employ Metropolis test like

$$A_{\text{eff}}(\{\mathbf{S}'\}, \{\mathbf{S}\}) = \min\left(1, W_{\text{eff}}(\{\mathbf{S}'\})/W_{\text{eff}}(\{\mathbf{S}\})\right).$$

SLMC: Self-learning Monte-Carlo

We can construct *double* MCMC with $H(\{\mathbf{S}\})$ and $H_{\text{eff}}(\{\mathbf{S}\})$

$$\{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\} \longrightarrow \{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\} \xrightarrow{\text{eff}} \{\mathbf{S}\} \longrightarrow$$

with Metropolis-*Hastings* test: $A(\{\mathbf{S}'\}, \{\mathbf{S}\}) = \min\left(1, \frac{W(\{\mathbf{S}'\})}{W(\{\mathbf{S}\})} \frac{W_{\text{eff}}(\{\mathbf{S}\})}{W_{\text{eff}}(\{\mathbf{S}'\})}\right).$

- **Effective model can have fit parameters**
- **Exact! It satisfies detailed balance with $W(\{\mathbf{S}\})$ (exact)**
- **It has been used for full QCD too (arXiv: 2010.11900, 2103.11965)**

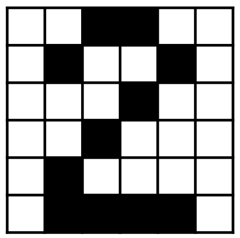
What is the neural networks?

Akio Tomiya

Neural network is a *universal* approximation function

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

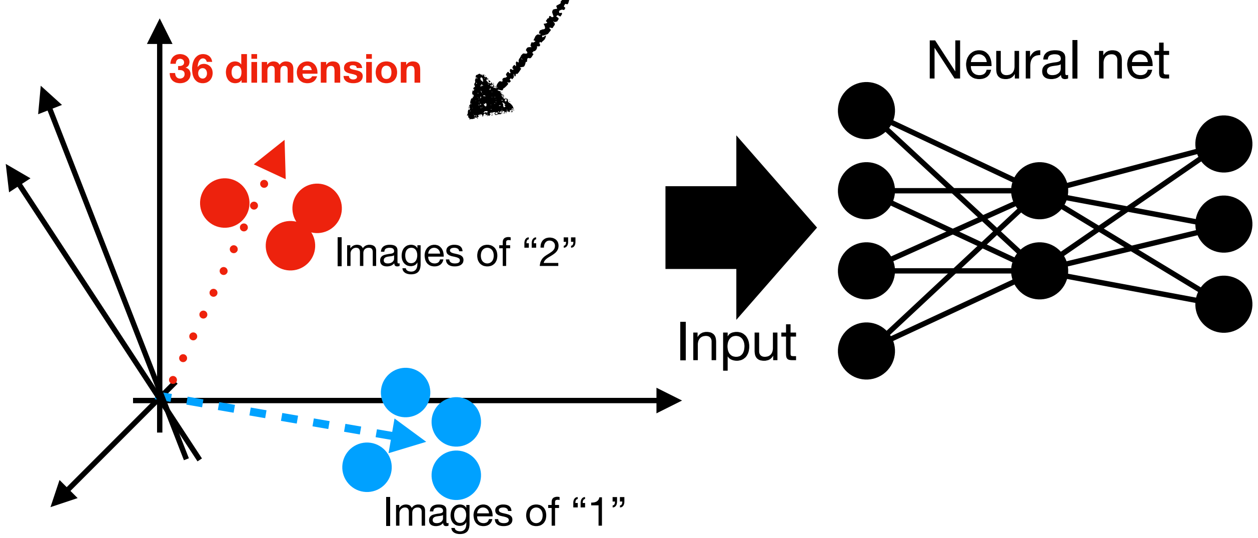
6x6



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

Image is a vector (6x6=36 dim)

Regard



Configuration generation in LQCD

Akio Tomiya

Loss = difference of action

WIP AT+

= MC steps

- Loss decreases along with the training steps
- it works as same as the stout (covariant net)
- Gain?

Introduction

Configuration generation with machine learning is developing

Year	Group	ML	Dim.	Theory	Gauge sym	Exact?	Fermion?	Lattice2021/ref
2017	**AT+**	RBM + HMC	2d	Scalar	-	No	No	arXiv: 1712.03893
2018	K. Zhou+	GAN	2d	Scalar	-	No	No	arXiv: 1810.12879
2018	J. Pawlowski +	GAN +HMC	2d	Scalar	-	Yes?	No	arXiv: 1811.03533
2019	MIT+	Flow	2d	Scalar	-	Yes	No	arXiv: 1904.12072
2020	MIT+	Flow	2d	U(1)	Equivariant	Yes	No	arXiv: 2003.06413
2020	MIT+	Flow	2d	SU(N)	Equivariant	Yes	No	arXiv: 2008.05456
2020	**AT+**	SLMC	**4d**	SU(N)	Invariant	Yes	Partially	arXiv: 2010.11900
2021	M. Medvidović+	A-NICE	2d	Scalar	-	No	No	arXiv: 2012.01442
2021	S. Foreman	L2HMC	2d	U(1)	Yes	Yes	No	
2021	**AT+**	SLHMC	**4d**	QCD	Covariant	Yes	YES!	
2021	L. Del Debbio+	Flow	2d	Scalar, O(N)	-	Yes	No	
2021	MIT+	Flow	2d	Yukawa	-	Yes	Yes	
2021	**S. Foreman, AT+**	Flowed HMC	2d	U(1)	Equivariant	Yes	No but compatible	arXiv: 2112.01586
2021	XY Jing	Neural net	2d	U(1)	Equivariant	Yes	No	
2022	J. Finkenrath	Flow	2d	U(1)	Equivariant	Yes	Yes (diagonalization)	arxiv: 2201.02216
2022	MIT+	Flow	2d	U(1)	Equivariant	Yes	Yes (diagonalization)	arXiv:2202.11712

+...

This is not complete list. Related to lattice field theory and biased

Akio Tomiya
Machine learning for theoretical physics

Organizing "Deep Learning and physics"

https://cometscome.github.io/DLAP2020/

What am I?

I am a particle physicist, working on lattice QCD.

I want to apply machine learning on lattice QCD.

My papers https://scholar.google.co.jp/citations?user=LKVqy_wAAAAJ

Detection of phase transition via convolutional neural networks
A Tanaka, A Tomiya
Journal of the Physical Society of Japan 86 (6), 063001

Detecting phase transition

Digital quantum simulation of the schwinger model with topological term via adiabatic state preparation
B Chakraborty, M Honda, T Izubuchi, Y Kikuchi, A Tomiya
arXiv preprint arXiv:2001.00485

Quantum computing for quantum field theory

Biography

2006-2010 : University of Hyogo (Superconductor)
2015 : PhD in Osaka university (Particle phys)
2015 - 2018 : Postdoc in Wuhan (China)
2018 - 2021 : SPDR in Riken/BNL (US)
2021 - : Assistant prof. in IPUT Osaka (ML/AI)

Kakenhi and others

Leader of proj A01 Transformative Research Areas, Fugaku

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

+quantum computer

Others:

2024 The 29th Outstanding Paper Award of the Physical Society of Japan

2023 Supervision of Shin-Kamen Rider

2021 14th Particle Physics Medal: Young Scientist Award

My team: LQCD + ML

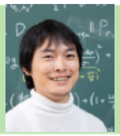
“Machine Learning Physics Initiative”

MLPhYs

2022-2027, 10M USD, 70 researchers

Director : K. Hashimoto

B01 A.Tanaka: Math and Application of DL



B02 Y.Kabashima: Statistical data ML

B03 K.Fukushima: Topology and Geometry of ML

+...

A01 A.Tomiya: Computational physics

+...

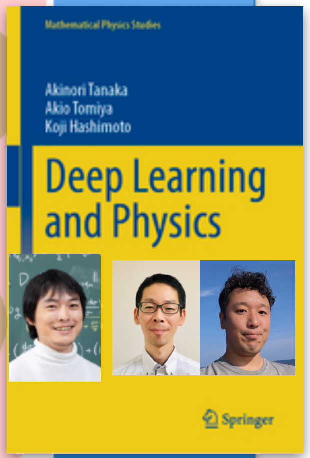
A02 M.Nojiri: High Energy Physics

A03 T.Ohtsuki: Condensed Matter Physics

A04 K.Hashimoto: Quantum and Gravity Physics

+...

ML Phys



2021

FY2022-2026 MEXT -KAKENHI- Grant-in-Aid for Transformative Research Areas (A)

科研費 KAKENHI

My team (A01): LQCD + ML

PI: Akio Tomiya (Me)
TWCU
LQCD, ML

Kouji Kashiwa
Fukuoka Institute of Technology
LQCD, ML

Hiroshi Ohno
U. of Tsukuba
LQCD

Tetsuya Sakurai
U. of Tsukuba
Computation

Yasunori Futamura
U. of Tsukuba
Computation

post-docs
& external members

B. J. Choi
U. of Tsukuba

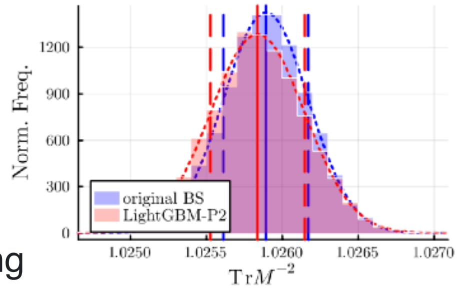
J. Takahashi
Meteorological College

Y. Nagai
U. of Tokyo

- **Apply machine learning techniques on LQCD (To increase what we can do)**
- **Find physics-oriented ML architecture**
- **Making codes for LQCD + ML**

ML Phys A01

measurement with BDT



On going

LatticeQCD.jl

Open source

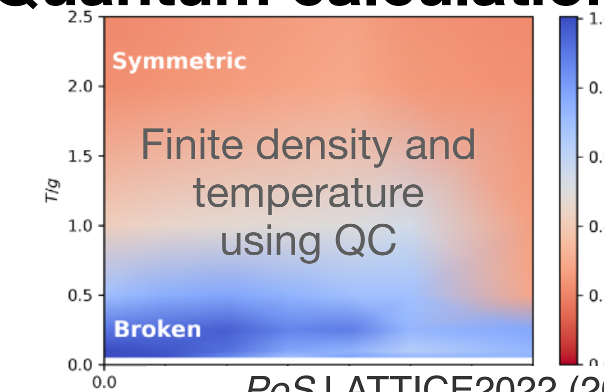
LQCD (+ML) with julia

This covers most of modern tech

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

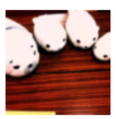
(and associated sub-libraries)

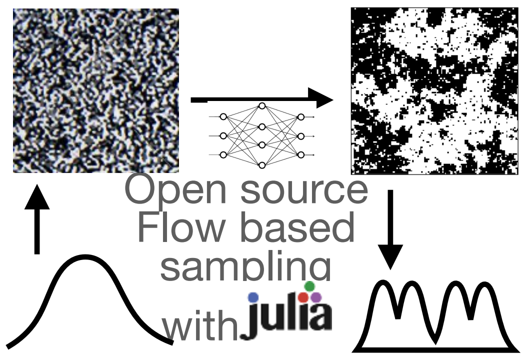
Quantum calculation



ML + QC: Quantum thermodynamics using Density matrix and MADE

PoS LATTICE2022 (2023) 039

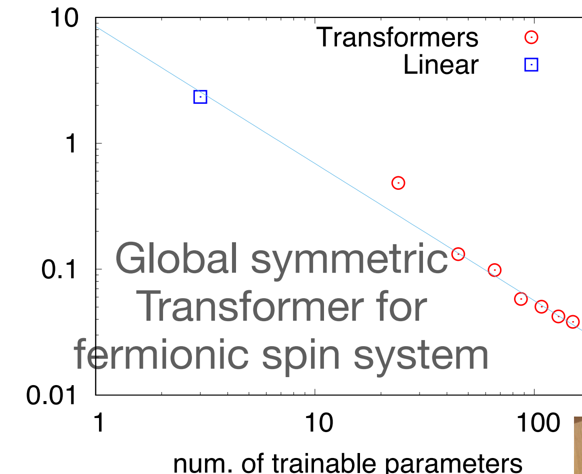




arXiv: 2208.08903

$$\frac{dU_\mu^{(t)}(n)}{dt} = \mathcal{G}^{\bar{\theta}}(U_\mu^{(t)}(n))$$

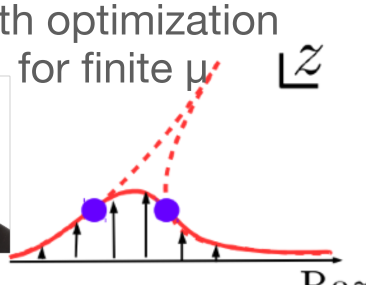
Gauge covariant neural net
arXiv: 2103.11965



2310.13222
2306.11527

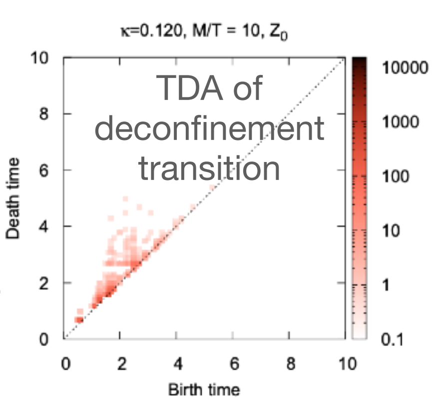
Gauge configuration

Path optimization for finite μ



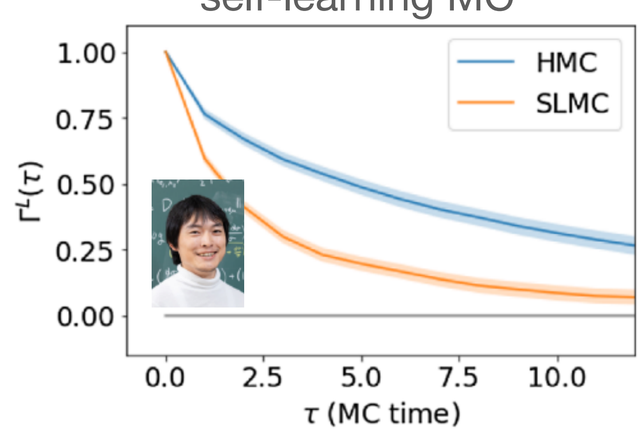
Phys. Rev. D 108, 094504
(Figure from 1812.11506)

Sign problem



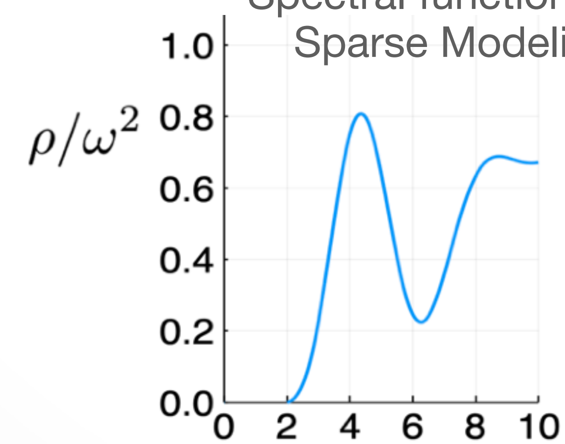
1810.07635

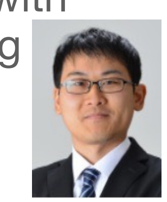
Gauge invariant self-learning MC



Phys. Rev. D 107, 054501

Spectral function with Sparse Modeling



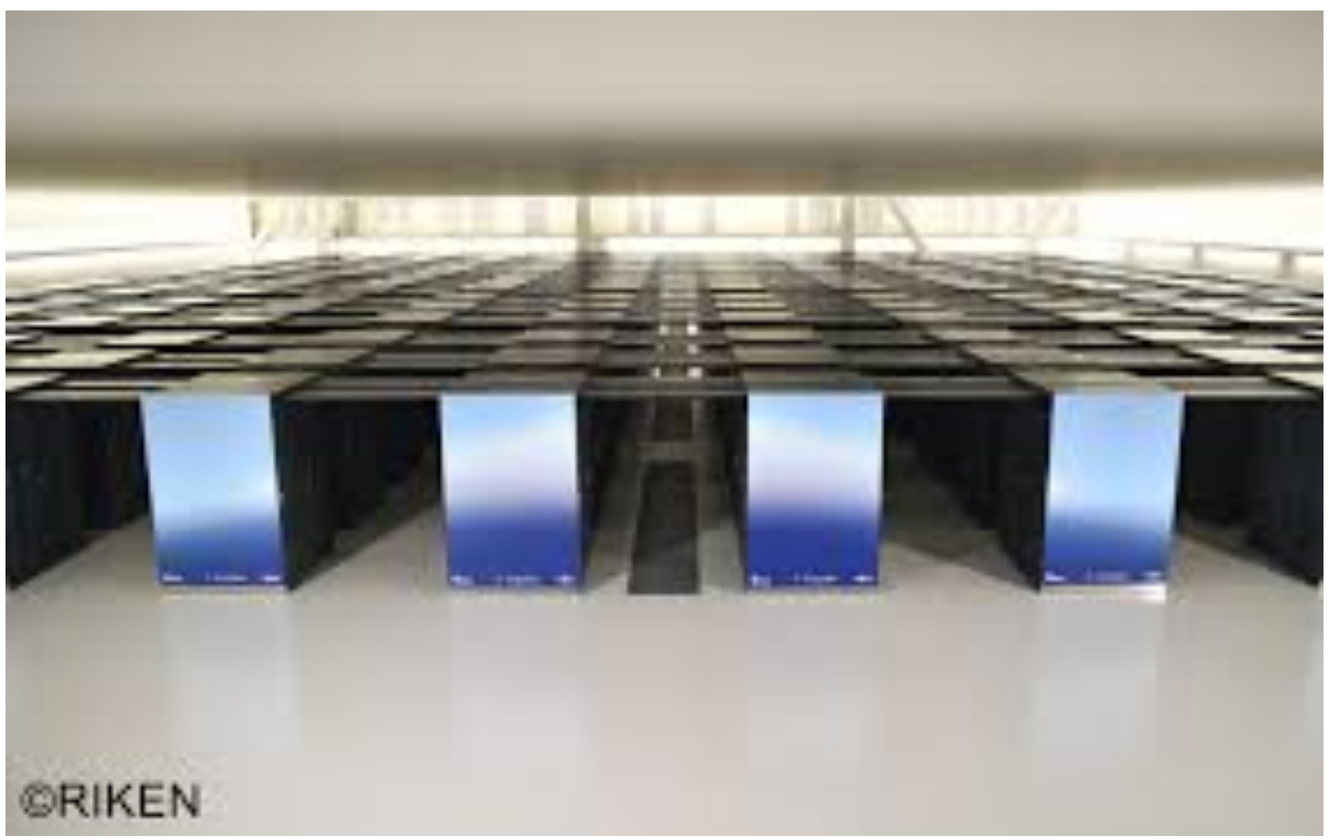


(arXiv: 2311.15233)
+ on going

Other projects are going (with me)

“Program for Promoting Researches on the Supercomputer Fugaku”

- Simulation for basic science: approaching the new quantum era
  - PI: Shoji Hashimoto
- Search for physics beyond the standard model using large-scale lattice QCD simulation and development of AI technology toward next-generation lattice QCD
  - PI: Takeshi Yamazaki



Intro: Lattice QCD& Monte-Carlo

LQCD = Non-perturbative calculation of QCD

QCD in 3 + 1 dimension

$$S = \int d^4x \left[-\frac{1}{2} \mathrm{tr}\, F_{\mu\nu} F^{\mu\nu} + \bar{\psi} \left(\mathrm{i} \partial\!\!\!/ + g A\!\!\!/ - m \right) \psi \right]$$

$$Z = \int \mathscr{D}A \mathscr{D}\bar{\psi} \mathscr{D}\psi e^{\mathrm{i}S} \qquad F_{\mu\nu} = \partial_\mu A_\nu - \partial_\nu A_\mu - \mathrm{i}g[A_\mu, A_\nu]$$

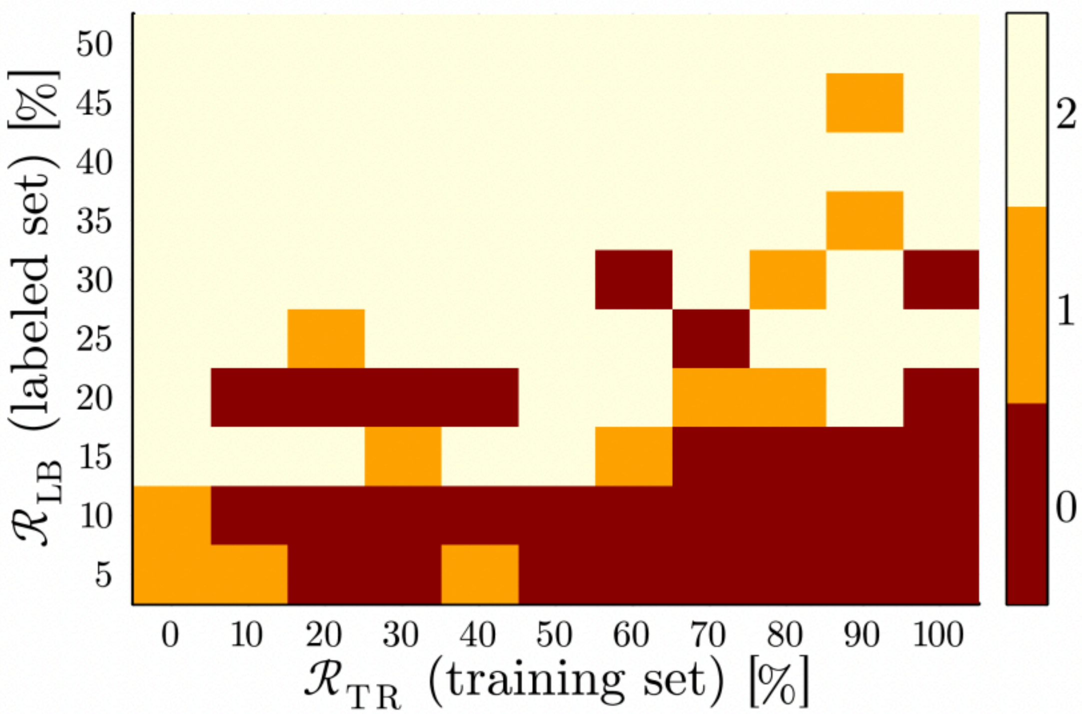
QCD in Euclidean 4 dimension (imaginary time)

$$S = \int d^4x \left[+\frac{1}{2} \mathrm{tr}\, F_{\mu\nu} F_{\mu\nu} + \bar{\psi} \left(\partial\!\!\!/ - \mathrm{i}g A\!\!\!/ + m \right) \psi \right]$$

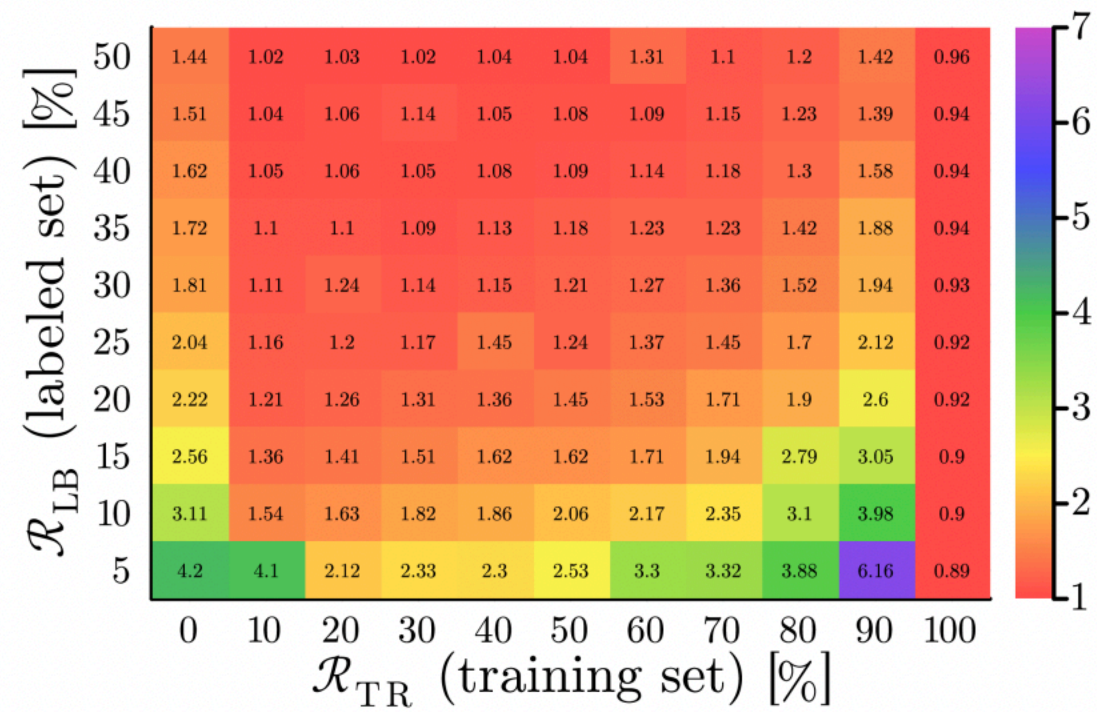
$$Z = \int \mathscr{D}A \mathscr{D}\bar{\psi} \mathscr{D}\psi e^{-S}$$

- Same Hamiltonian with real-time formalism
- Static property is the same (mass etc)
- How to calculate?

Example: Plaquette $\to \mathrm{Tr}M^{-3}$ estimation ($\mathcal{P}2$, ID-0)

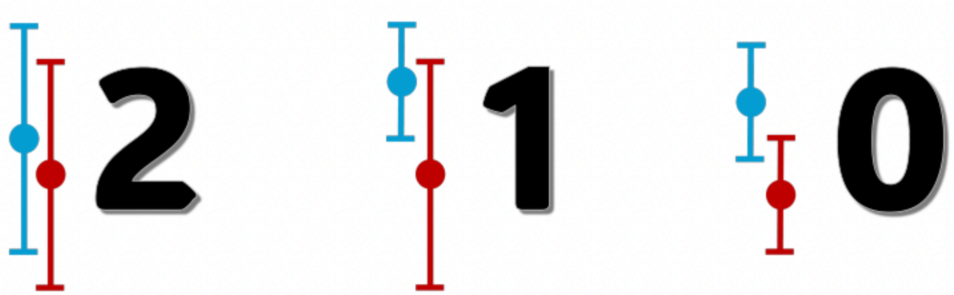


(a) $\bar{Y}$ (central value) check



(b) Magnitude of $\sigma_{\mathcal{P}2}/\sigma_{\mathrm{Orig.}}$

- $\bar{Y}$ score: white, orange, red



1 **Eval. 1:** central value check
➡ consistently white region
n $\mathcal{R}_{\mathrm{LB}} \geq 30\,\%$, $\mathcal{R}_{\mathrm{TR}} \leq 50\,\%$

2 **Eval. 2:** $\sigma_{\mathcal{P}2}/\sigma_{\mathrm{Orig.}}$ check
➡ Roughly $\boldsymbol{\sigma_{\mathcal{P}2} \lesssim 1.1\sigma_{\mathrm{Orig.}}}$
in $\mathcal{R}_{\mathrm{LB}} \geq 30\,\%$, $\mathcal{R}_{\mathrm{TR}} \leq 50\,\%$

Motivation

Monte-Carlo integration is available, but still expensive!

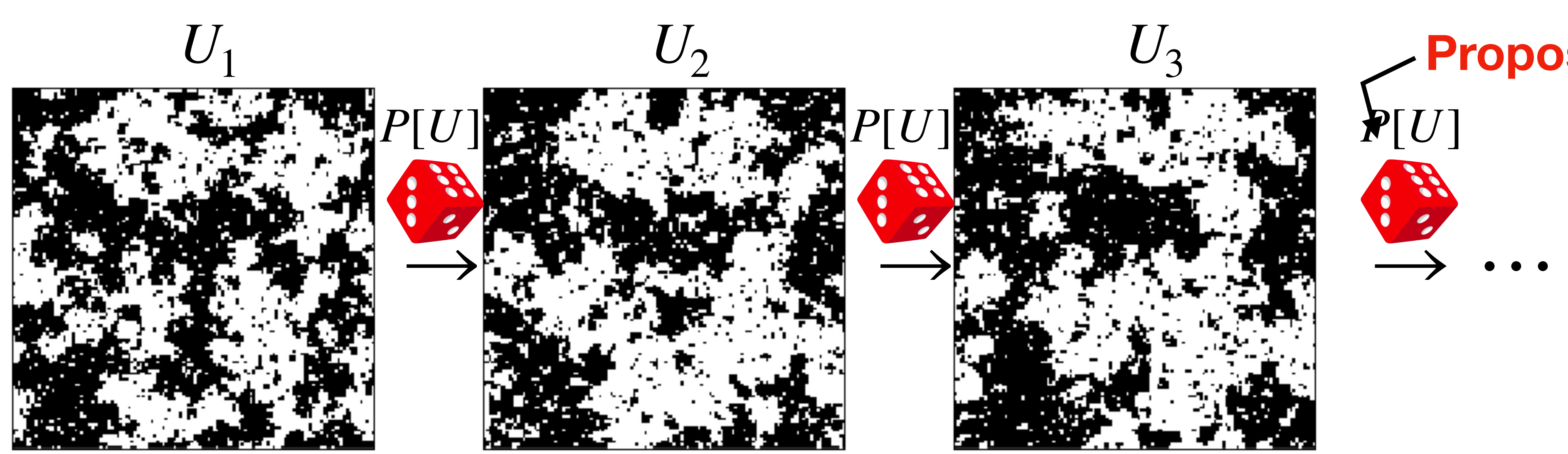
M. Creutz 1980

Target integration
= expectation value $\langle \mathcal{O} \rangle = \frac{1}{Z}\int \mathcal{D}U e^{-S_{\rm eff}[U]}\mathcal{O}(U)$

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

Monte-Carlo: Generate field configurations with "$P[U] \propto e^{-S_{\rm eff}[U]}$". It gives expectation value

Markov-Chain



$$\langle \mathcal{O} \rangle \approx \frac{1}{N_{\rm sample}} \sum_{k=1}^{N_{\rm sample}} \mathcal{O}[U_k]$$

Production with is numerically expensive
and how can we accelerate it? We use machine learning!

Introduction

Use of symmetry is crucial

Symmetries are essential for theoretical physics.
This is actually true as well in machine learning.
Equivariance/Covariance of symmetries helps generalization, and avoiding wrong extrapolation
(Symmetry restricts the function form)

Example in ML:

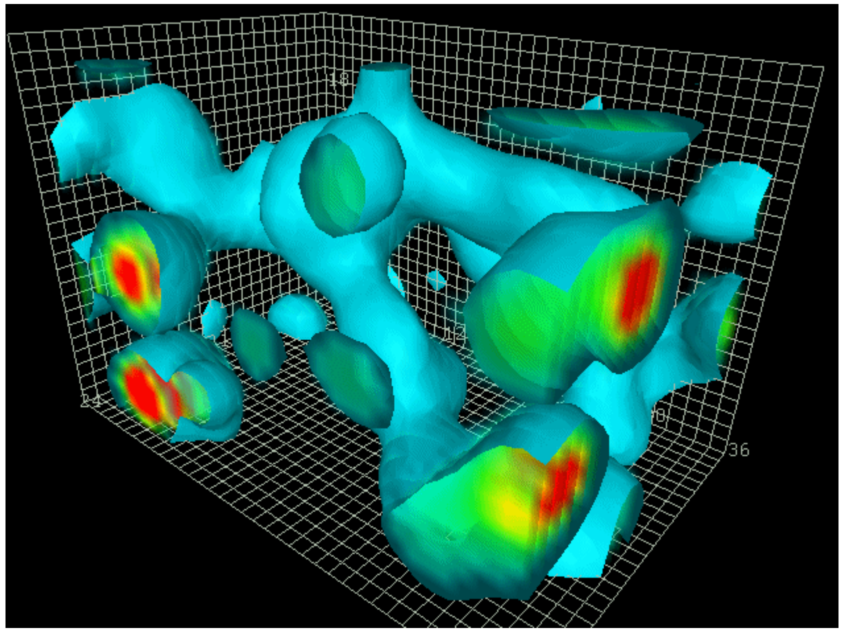
If data is translationally symmetric like photo images,
the frame work should respect this and one should implement
with this translational symmetry in a neural network
= Convolutional neural net!

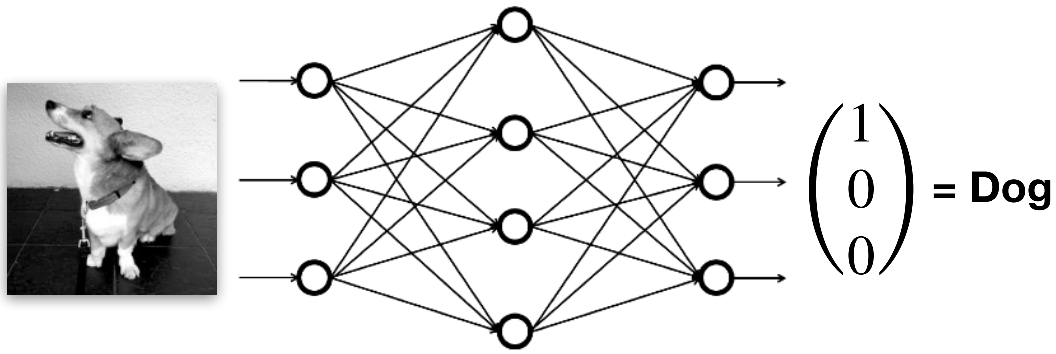
In physics + Machine learning,
= Physics embedded neural networks

We use symmetry in the system
as much as we can

Introduction

What is our final goal for QCD + Machine learning?





What we want to solve using machine learning?

- Reduction of numerical cost to beyond our current numerical limitations
  - Production and measurements
  - Use of machine learning may be useful

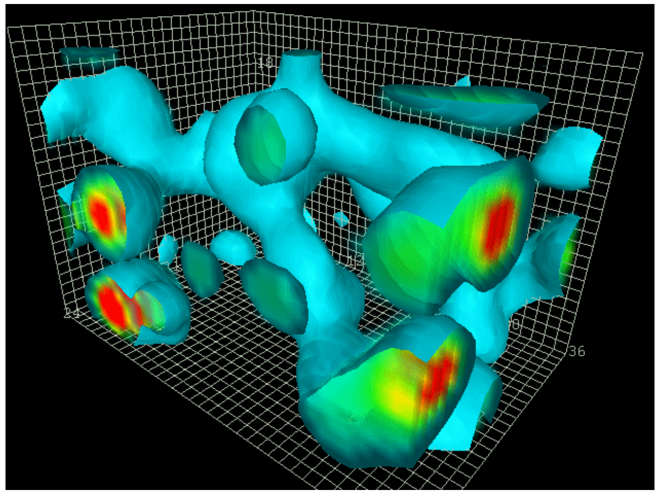
Restrictions (problems) to use ML:

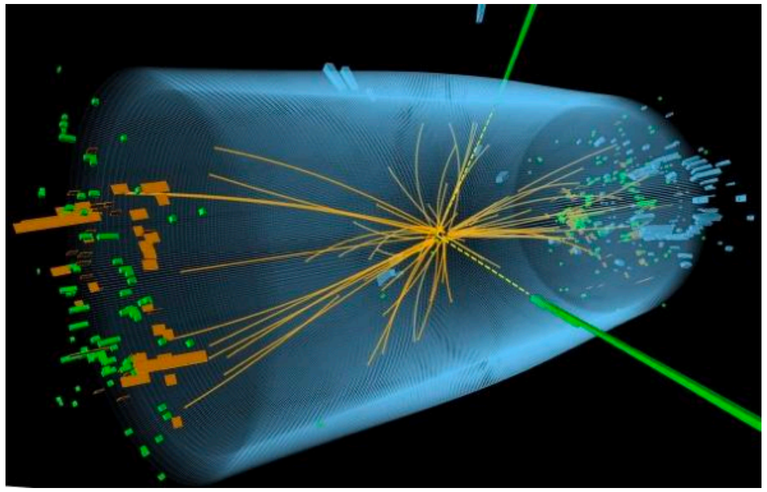
- Exactness & quantitative. Machine learning is an approximator
- **Gauge symmetry**, global symmetry is essential. While ML is not for physics
- Code. How can we make neural nets w/ HPC? (not showing in this talk)

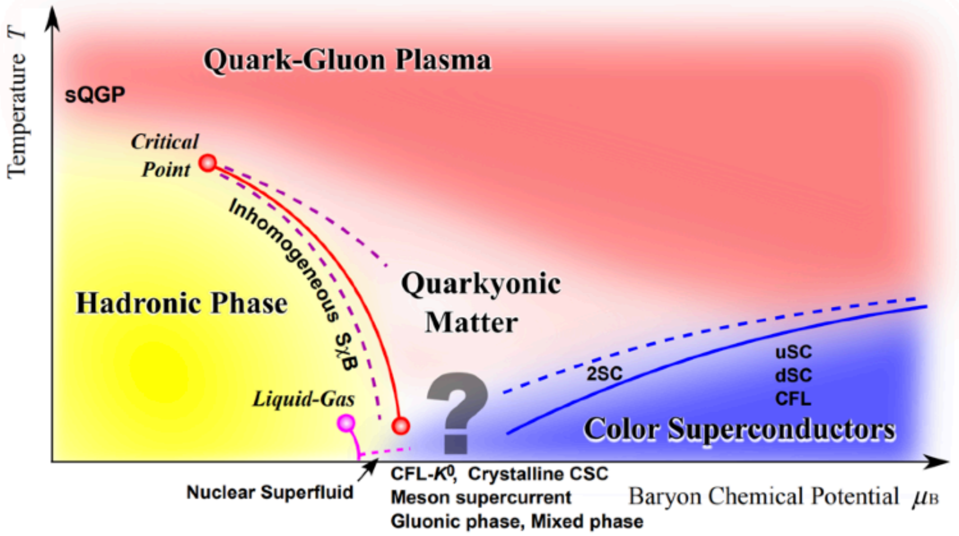
Introduction

What is our final goal for our research field?

Fukushima , Hatsuda
Rept.Prog.Phys.74:014001,2011







In short, we simulate of elementary particles in nuclei

Using super computers + Lattice QCD, we can understand…
- melting of protons/neutrons etc. at high temperatures
- attractive/repulsive forces between atomic nuclei
- candidate properties of dark matter

etc.

Intro: Lattice QCD& Monte-Carlo

Akio Tomiya

Numerical integral (via trapezoidal type) is impossible

$$S = \int d^4x \left[+ \frac{1}{2} \mathrm{tr}\, F_{\mu\nu} F_{\mu\nu} + \bar{\psi} \left(\partial\!\!\!/ - ig A\!\!\!/ + m \right) \psi \right]$$

Lattice regularization

$$S[U, \psi, \bar{\psi}] = a^4 \sum_n \left[- \frac{1}{g^2} \mathrm{Re}\ \mathrm{tr}\ U_{\mu\nu} + \bar{\psi} \left(D\!\!\!\!/ + m \right) \psi \right]$$

a is lattice spacing (cutoff)

They are ``same" up to irreverent operators

$$\mathrm{Re}\ U_{\mu\nu} \sim \frac{-1}{2} g^2 a^4 F_{\mu\nu}^2 + O(a^6)$$

$$\langle \mathcal{O} \rangle = \frac{1}{Z} \int \mathcal{D}U \mathcal{D}\bar{\psi} \mathcal{D}\psi e^{-S} \mathcal{O}(U) = \frac{1}{Z} \int \mathcal{D}U e^{-S_{\mathrm{gauge}}[U]} \det(D+m) \mathcal{O}(U)$$

$$= \frac{1}{Z} \int \underline{\mathcal{D}U} e^{-S_{\mathrm{eff}}[U]} \mathcal{O}(U)$$

$$= \prod_{n \in \{\mathbb{Z}/L\}^4} \prod_{\mu=1}^{4} dU_\mu(n)$$

>1000 dim, no hope with
trapezoidal type numerical Integration -> use (Markov-chain) Monte Carlo

Flow based sampling algorithm

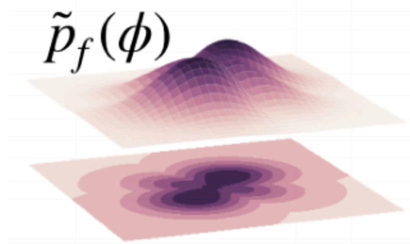
Akio Tomiya

Trivialization is attractive

Joint dist.

QFT probability:
Propagating modes
~ correlations

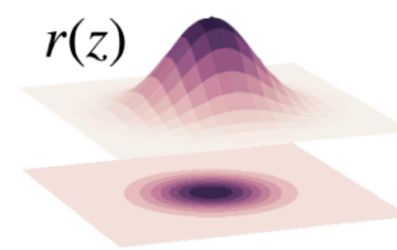
$$P[\phi] = \frac{1}{Z} \mathrm{e}^{-S[\phi]} = P(\phi_1, \phi_2, \cdots, \phi_{L^4})$$



Can we find a change of variable?

Trivial distribution
Trivial theory
No propagation, factorized
(Not the Gaussian FP)

$$P^{\mathrm{tri}}[z] = r(z_1) r(z_2) \cdots r(z_{L^4})$$



$r(z_i)$ probability for 1 variable
Easy to sample

- Correlations in $P[\phi]$ makes theory non-trivial and it makes MCMC harder.
- $P^{\mathrm{tri}}[z] = r(z_1) r(z_2) \cdots r(z_{L^4})$ has no correlation, sampling is trivial.
- Actually, there is a map between them. Trivializing map!
  - We can trivialize the target theory

Famous example: Nicolai map in SUSY. **Change of variable makes theory bilinear (~trivial)**. How about for non-SUSY?

arxiv 1904.12072, 2003.06413, 2008.05456 and more.

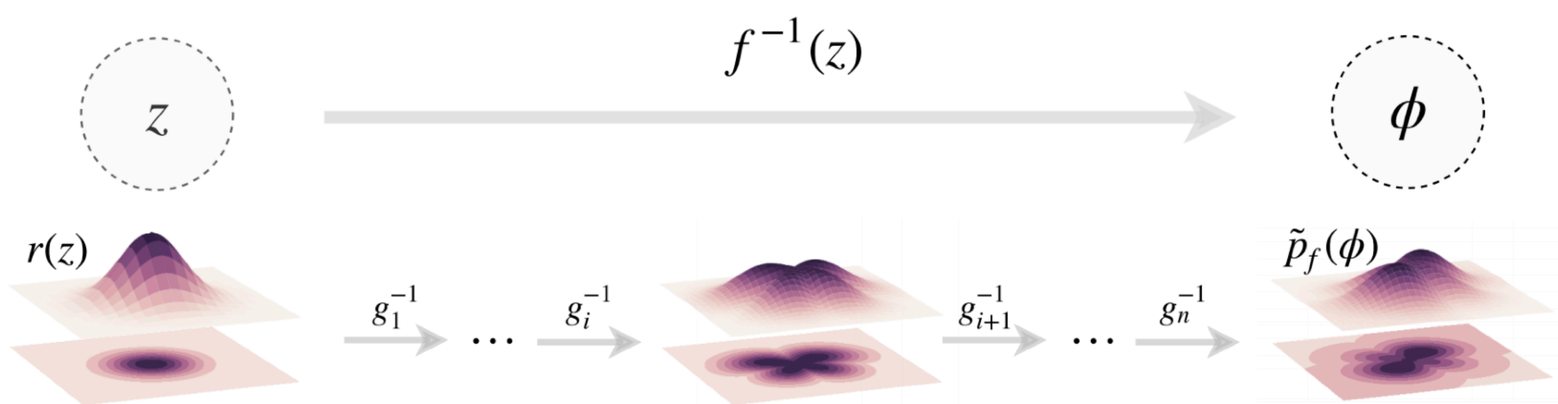
Nicolai, H., Phys. Lett. 89B, 341 (1980); Nucl. Phys. B176, 419 (1980).

Related works

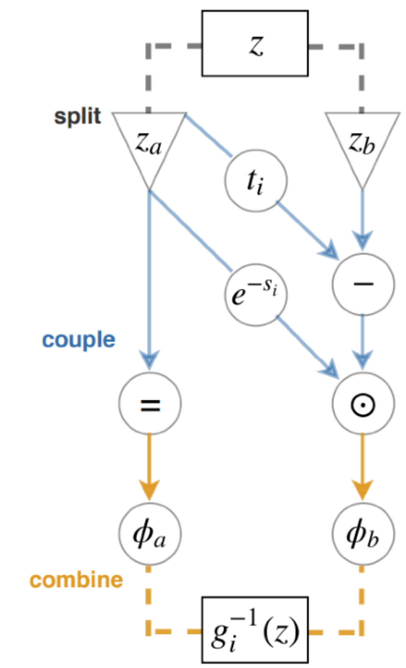
Flow based algorithm = neural net represented flow algorithm

Real scalar in 2 dimension

MIT + DeepMind 2019~



(a) Normalizing flow between prior and output distributions



(b) Inverse coupling layer

FIG. 1: In (a), a normalizing flow is shown transforming samples z from a prior distribution $r(z)$ to samples ϕ distributed according to $\tilde{p}_f(\phi)$. The mapping $f^{-1}(z)$ is constructed by composing inverse coupling layers g_i^{-1} as defined in Eq. (10) in terms of neural networks s_i and t_i and shown diagrammatically in (b). By optimizing the neural networks within each coupling layer, $\tilde{p}_f(\phi)$ can be made to approximate a distribution of interest, $p(\phi)$.

Their sampling strategy

sample gaussian → inverse trivializing map → QFT configurations
Calculate Jacobian
After sampling, Metropolice-Hasting test (Detailed balance)→ exact!

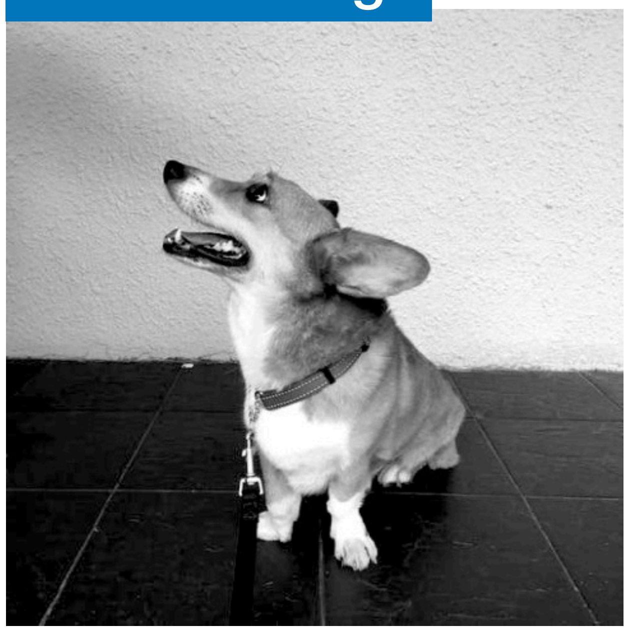
arxiv 1904.12072, 2003.06413, 2008.05456

Configuration generation in LQCD

Akio Tomiya

Convolution layer = trainable filter

Filter on image



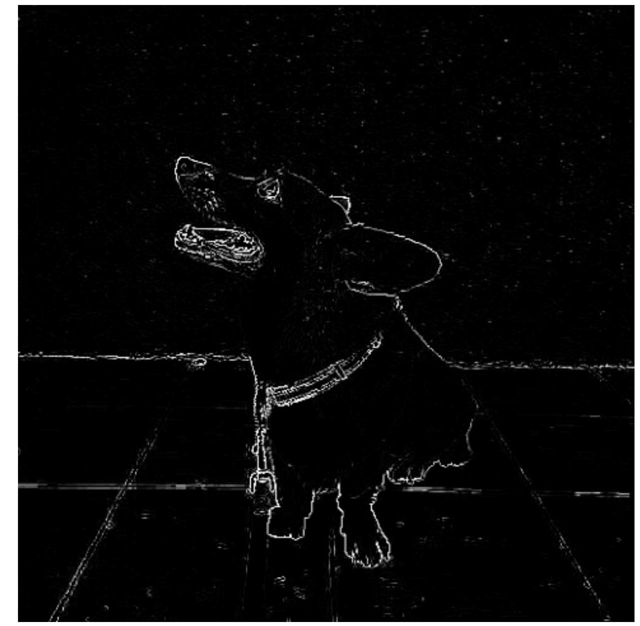
$*$

Laplacian filter

0	1	0
1	-2	1
0	1	0

(Discretization of ∂^2)

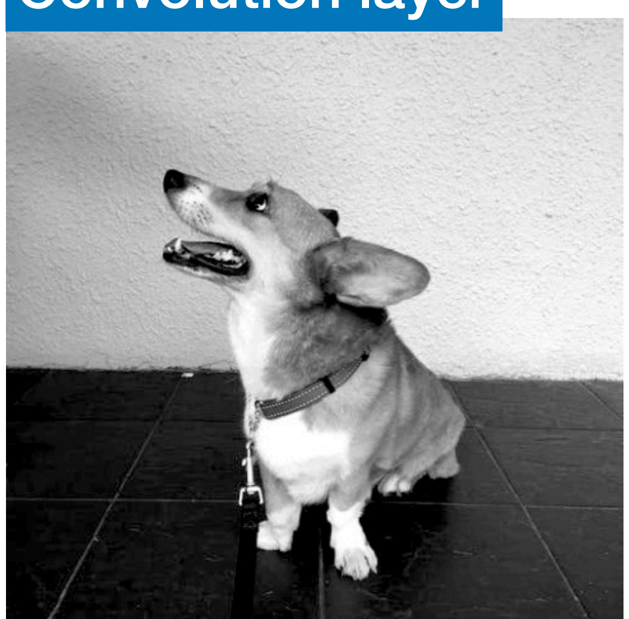
$=$



Edge detection

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

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

Convolution respects transnational symmetry as well

Configuration generation in LQCD

Smearing = Smoothing of gauge fields

Eg.

Coarse image

Gaussian filter

$$\frac{1}{16}\begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 4 & 1 \\ \hline 1 & 2 & 1 \\ \hline\end{array} \longrightarrow$$

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

Configuration generation in LQCD

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)

It has similar structure with neural networks,

$$\begin{cases} z_i^{(l)} = \sum_j w_{ij}^{(l)} u_j^{(l-1)} + b_i^{(l)} \\ u_i^{(l)} = \sigma^{(l)}(z_i^{(l)}) \end{cases}$$

Matrix product
vector addition

element-wise (local)
Non-linear transf.
Typically σ ~ tanh shape

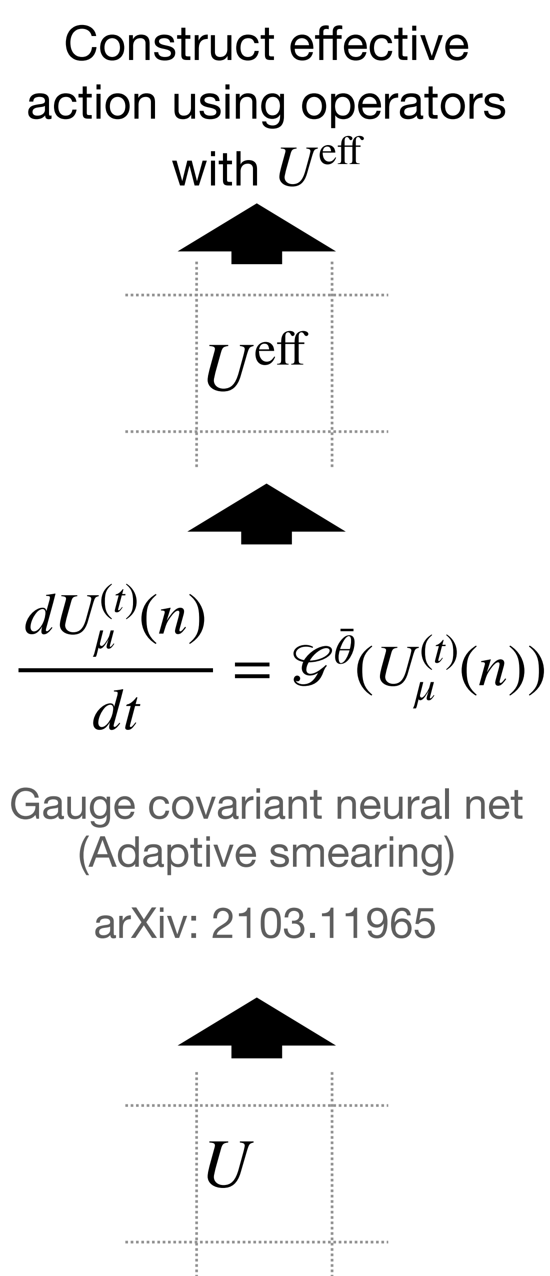
(Index i in the neural net corresponds to n & μ in smearing. Information processing with NN is evolution of scalar field)

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

As same as the convolution, we can train weights.

Configuration generation in LQCD
Simulation parameter

LatticeQCD.jl

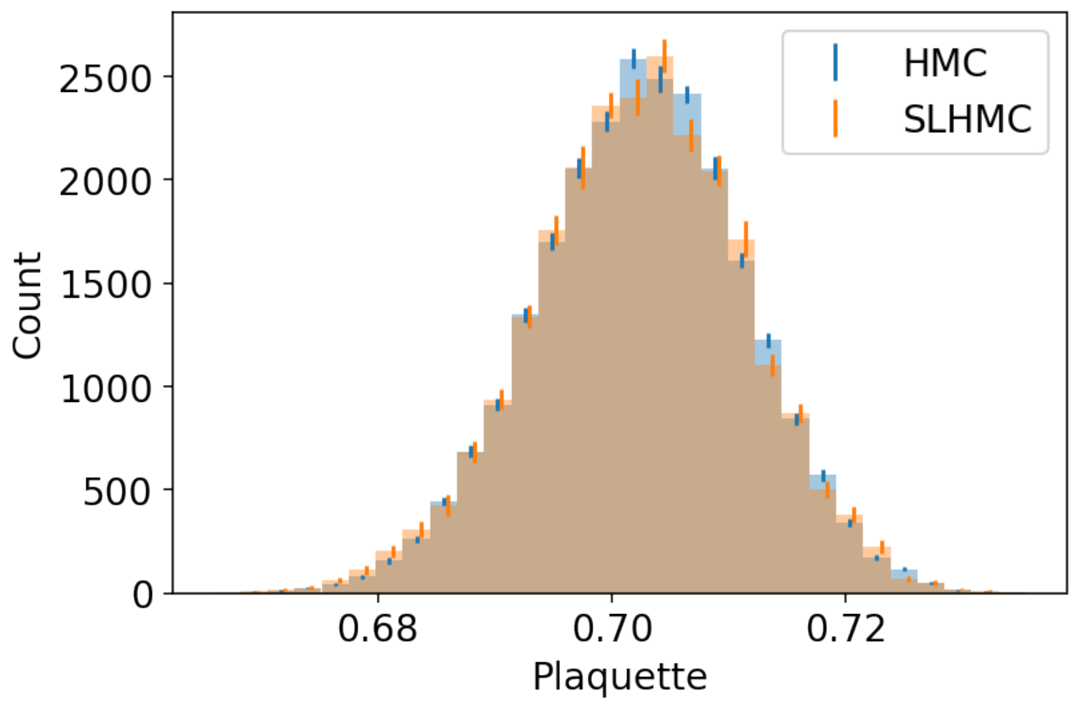


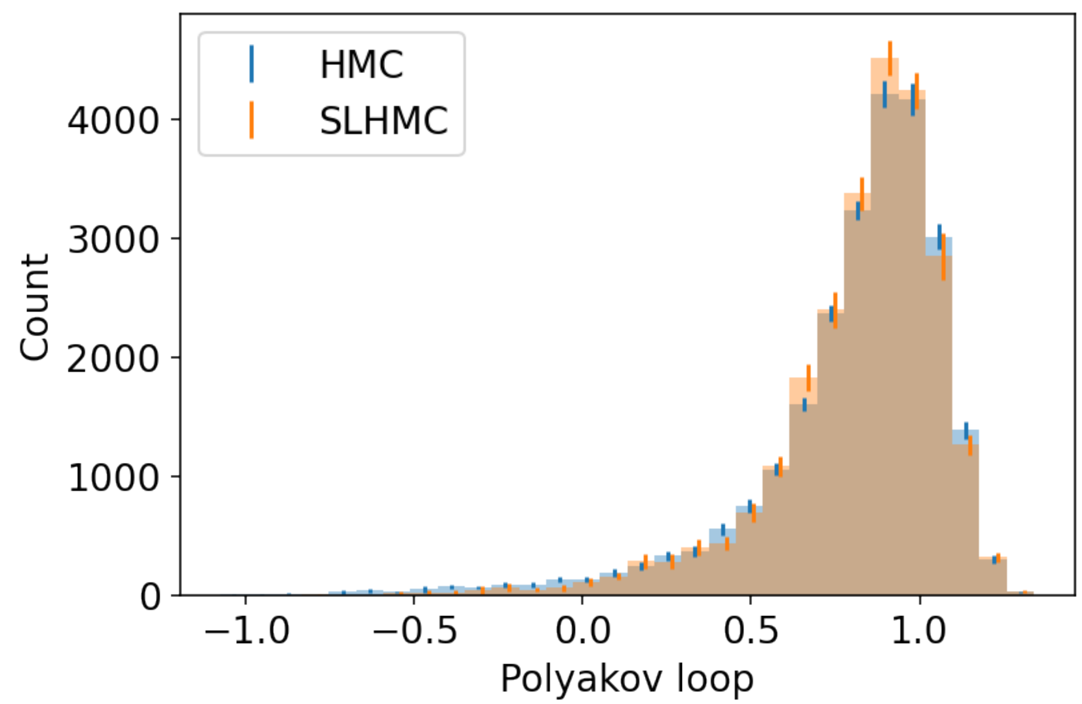
- Self-learning HMC (1909.02255, 2021 AT+), an exact algorithm
  - Exact Metropolis test and MD with effective action
- Target S : $m = 0.3$, dynamical staggered fermion, Nf=2, $L^4 = 4^4$, SU(2), $\beta = 2.7$. In Metropolis test
- Effective action S^{eff} in Molecular dynamics
  - Same gauge action
  - $m_{\mathrm{eff}} = 0.4$ dynamical staggered fermion, Nf=2
    - Gauge covariant neural network (adaptive stout)
    - Bare U is fed, adaptively smeared U^{eff} is pop out
  - **U links are replaced by U^{eff} in D_{stag}**
- “Adaptively reweighted HMC”

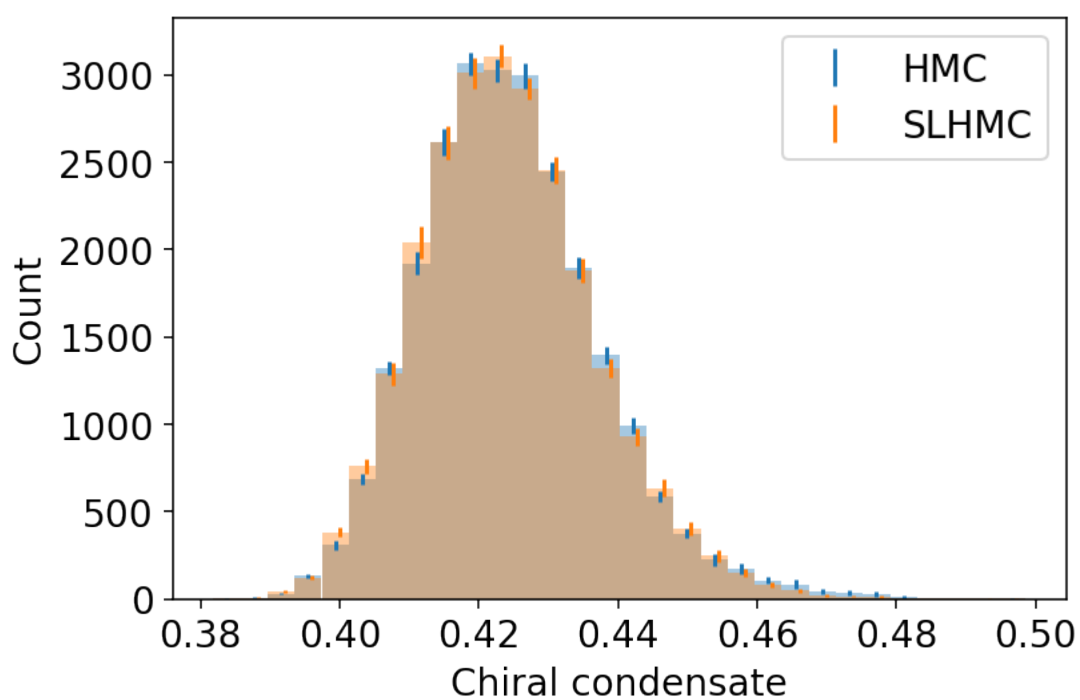
Configuration generation in LQCD

Application for the Full QCD in 4d

AT Y. Nagai arXiv: 2103.11965







Expectation value Acceptance = 40%

Algorithm	Observable	Value
HMC	Plaquette	0.7025(1)
SLHMC	Plaquette	0.7023(2)
HMC	|Polyakov loop|	0.82(1)
SLHMC	|Polyakov loop|	0.83(1)
HMC	Chiral condensate	0.4245(5)
SLHMC	Chiral condensate	0.4241(5)

What is showed?
Covariant net can mimic/absorb mass difference
SLHMC (~Adaptive reweighting) works