Applications of machine learning to computational physics

RBRC
RIKEN BNL Research Center

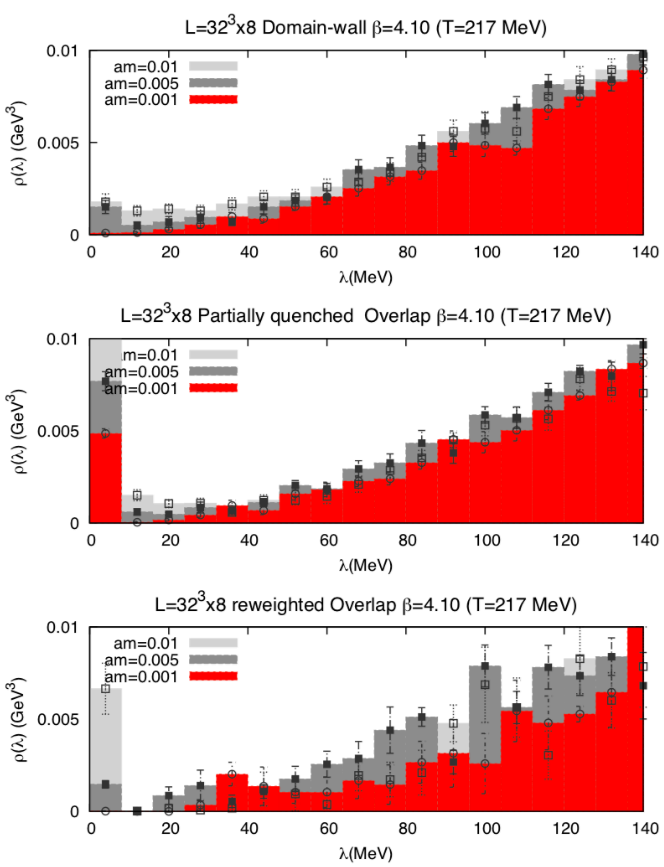
Akio Tomiya (RIKEN-BNL)

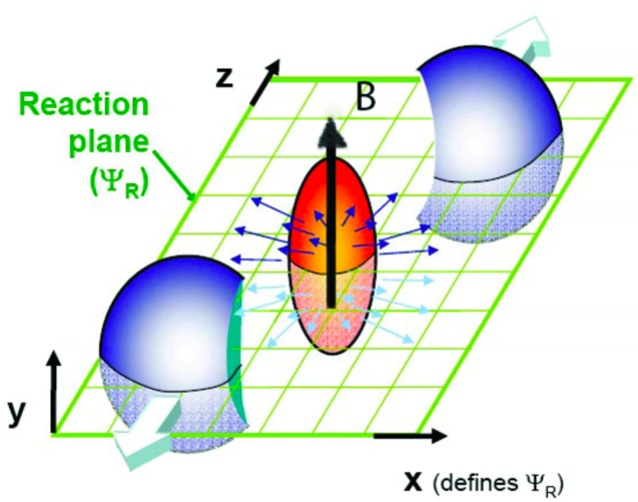
Based on arXiv: 1609.09087, 1812.01522, 1712.03893
and works in progress and preliminary

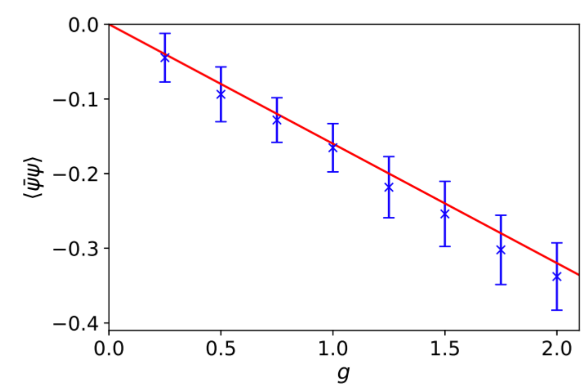
Self-Introduction

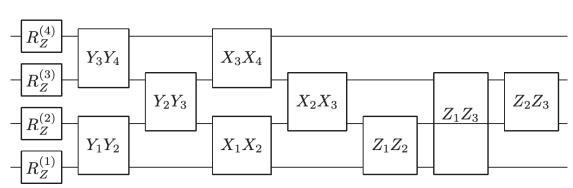
Who and what am I?

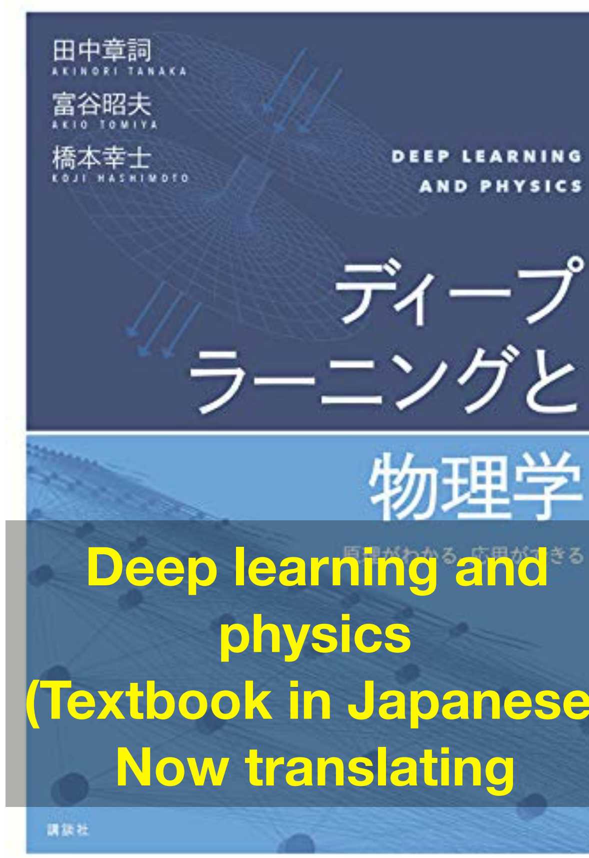
1. I have been working on lattice gauge theory
   1. Walking technicolor. arXiv:1411.1155
   2. U(1) axial anomaly at finite temperature with OV/DW. arXiv:1612.01908, …
   3. Finite temperature QCD with magnetic field. arXiv: 1904.01276 …
   4. lattice QED in 2D via quantum computing. arXiv: 2001.00485
2. **Machine learning (today's topic, some of them are on-going)**
   1. **Detection of phase transition. arXiv:1609.09087, 1812.01522**
   2. **Gauge configuration generation. arXiv: 1712.03893 + α**
   3. **QCD Spectral function**











Deep learning and physics (Textbook in Japanese) Now translating

Outline

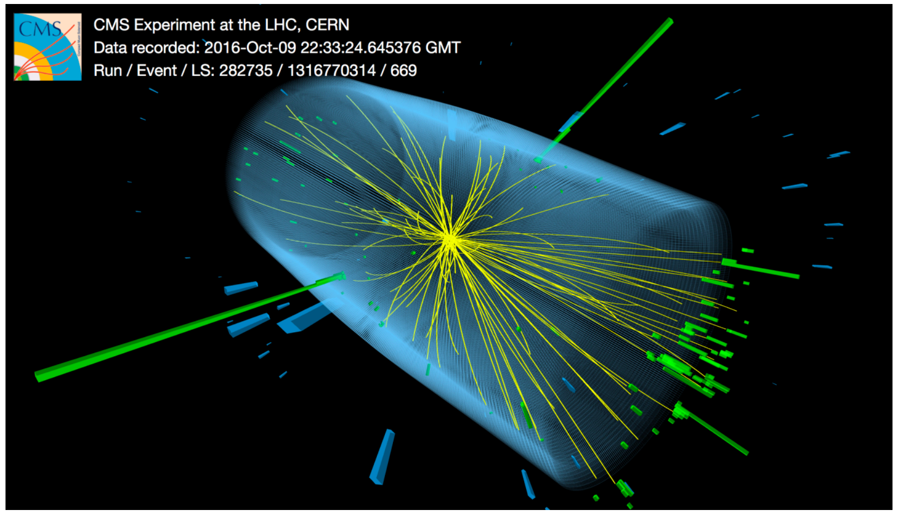
1. **Machine learning?**
2. **Detection of phase transition**
3. **Configuration generation** **(work in progress)**
4. **QCD spectral function** **(work in progress)**
5. **Summary**

Machine learning?

What is machine learning?

A way of theoretical high-energy physics

Akio Tomiya

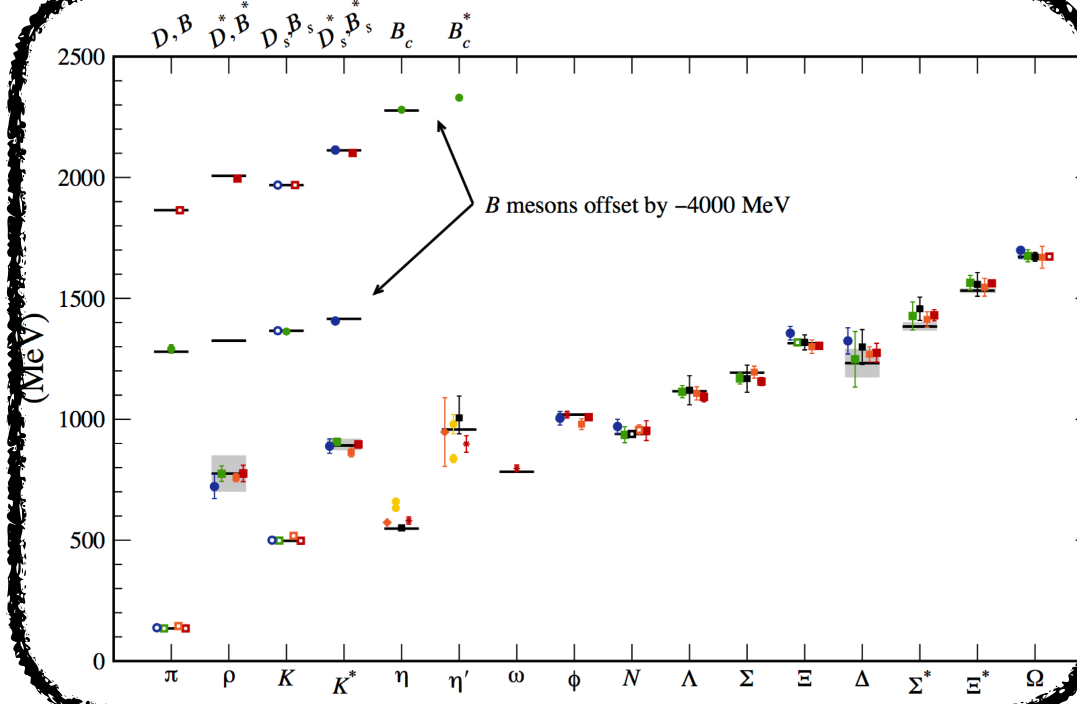


"Unknown theory"

generate

"mimic"

Data, input



Prediction (outside of data)

$$\mathcal{L} = -\frac{1}{4} F_{\mu\nu} F^{\mu\nu} + i\bar{\psi} \not{D} \psi + h.c. + \psi_i y_{ij} \psi_j \phi + h.c. + |D_\mu \phi|^2 - V(\phi)$$

The standard model

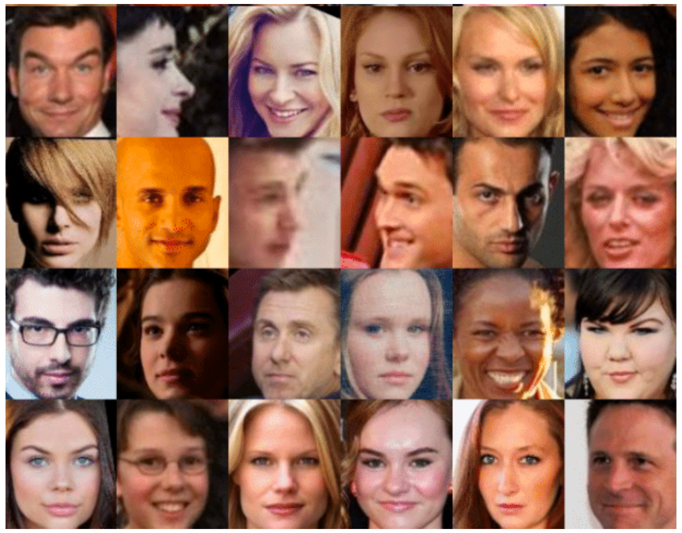
Determine parameters

https://www.usqcd.org/computing.html

What is machine learning?

Akio Tomiya

In physics language, modeling and prediction



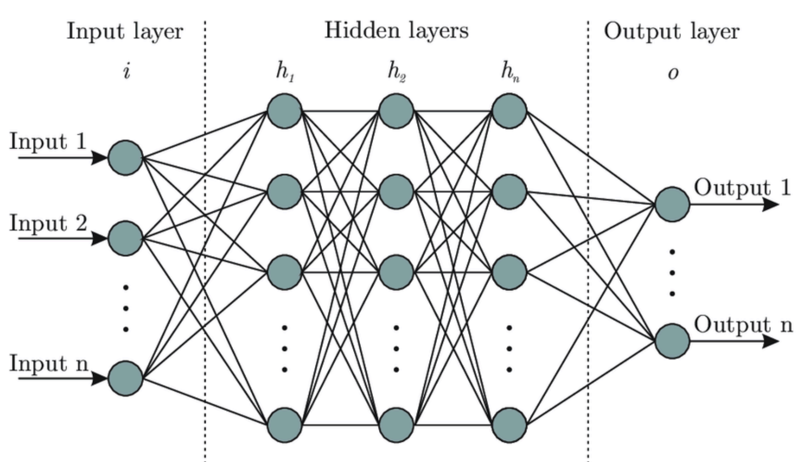
"Unknown theory"

generate

"mimic"

Data, input

"Prediction" (outside of data)



Determine parameters

Neural network (or some model)

results by style-GAN

Example: neural network?

Fit ansatz with multi-nested linear/non-linear func. with parameters

Input

Flatten $\longrightarrow$ $\vec{x} = \begin{pmatrix} 0.1 \\ 0.0 \\ 0.5 \\ \vdots \end{pmatrix}$ **vector**

Answers

Dog
Cat

Regard as a vector $\longrightarrow$

$\left.\begin{array}{ll} \textbf{Dog} & (1,0)^\top \\ \textbf{Cat} & (0,1)^\top \end{array}\right\} \vec{y}_{ans}$ **vector**

$$\vec{x} = \begin{pmatrix} 0.1 \\ 0.0 \\ 0.5 \\ \vdots \end{pmatrix} \xrightarrow{\text{f:“Neural net.”}} \vec{y}_{ans}$$

Neural network is a parametrized non-linear map between two vector space

What is the neural network?

Akio Tomiya

Fit ansatz with multi-nested linear/non-linear func. with parameters

Input

Flatten →

$$\vec{x} = \begin{pmatrix} 0.1 \\ 0.0 \\ 0.5 \\ \vdots \end{pmatrix}$$

vector

Answers

Dog
Cat

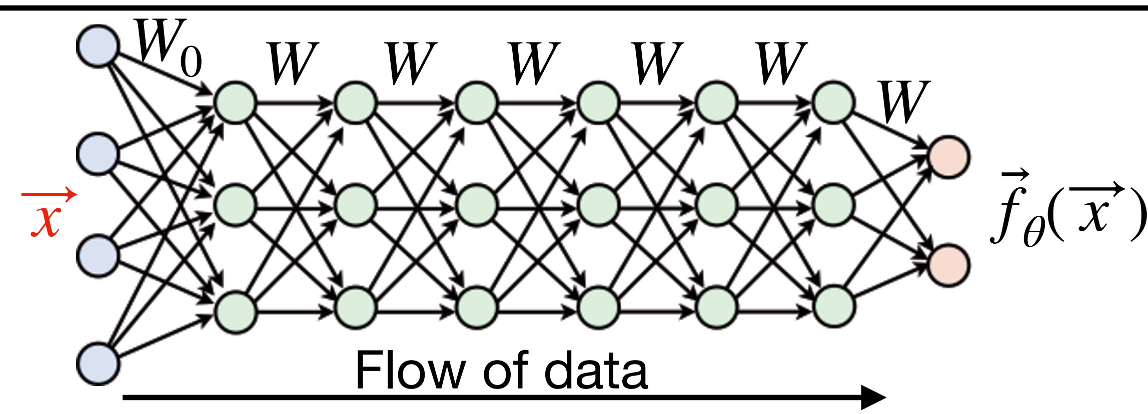
Regard as a vector →

Dog $(1,0)^\top$
Cat $(0,1)^\top$

$\left.\begin{matrix} (1,0)^\top \\ (0,1)^\top \end{matrix}\right\} \vec{y}_{ans}$

vector

Neural network



Parameters

$\left.\begin{matrix} W_i \text{ (matrix)} \\ \vec{b}_i \end{matrix}\right\} \theta$

$$\vec{f}_\theta(\vec{x}) = \sigma(W_3\sigma(W_2\sigma(W_1\sigma(W_0\vec{x} + \vec{b}_0) + \vec{b}_1) + \vec{b}_2) + \vec{b}_3)$$

$\sigma(\vec{v})$: element-wise nonlinear function (eg tanh), “activation”

What is the neural network?

Akio Tomiya

Fit ansatz with multi-nested linear/non-linear func. with parameters

Input

Flatten

$\vec{x} = \begin{pmatrix} 0.1 \\ 0.0 \\ 0.5 \\ \vdots \end{pmatrix}$

vector

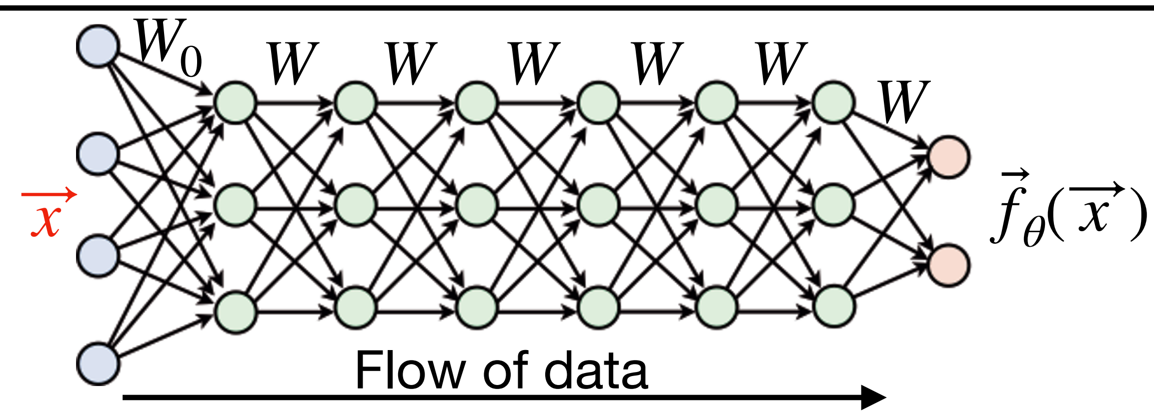
Answers

Dog	**Regard as a vector** →	**Dog**	$(1,0)^\top$	
Cat		**Cat**	$(0,1)^\top$	

$\left.\begin{matrix}(1,0)^\top \\ (0,1)^\top\end{matrix}\right\} \vec{y}_{ans}$

vector

Neural network



Parameters

$\left.\begin{matrix} W_i \text{ (matrix)} \\ \vec{b}_i \end{matrix}\right\} \theta$

$$\vec{f}_\theta(\vec{x}) = \sigma(W_3\sigma(W_2\sigma(W_1\sigma(W_0\vec{x} + \vec{b}_0) + \vec{b}_1) + \vec{b}_2) + \vec{b}_3)$$

$\sigma(\vec{v})$: element-wise nonlinear function (eg tanh), “activation”

“Training” = optimization

Minimize “distance” between $\vec{f}_\theta(\vec{x})$ and $\vec{y}_{ans}$ for data in dataset by tuning θ

1.Detection of phase transition

arXiv: 1609.09087(w/ A. Tanaka),
1812.01522(w/ K. Kashiwa, Y. Kikuchi)

Can “AI” detect phase transition?

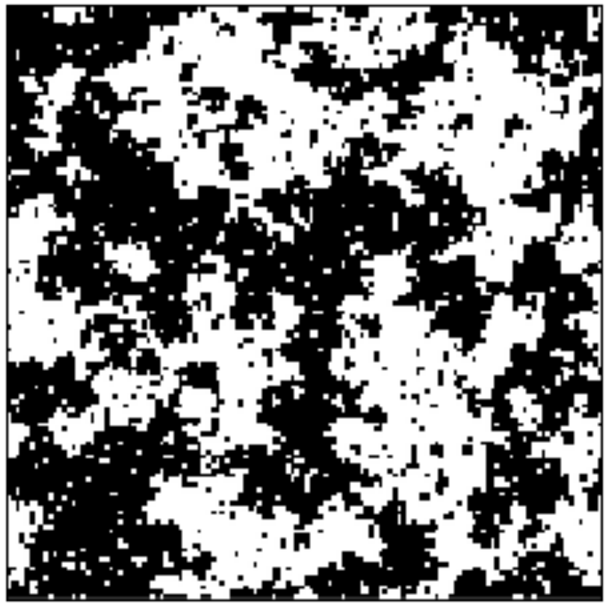
We train Neural net as a thermometer (Classification problem)

A.Tanaka AT 1609.09087
K. Kashiwa, Y. Kikuchi AT 1812.01522

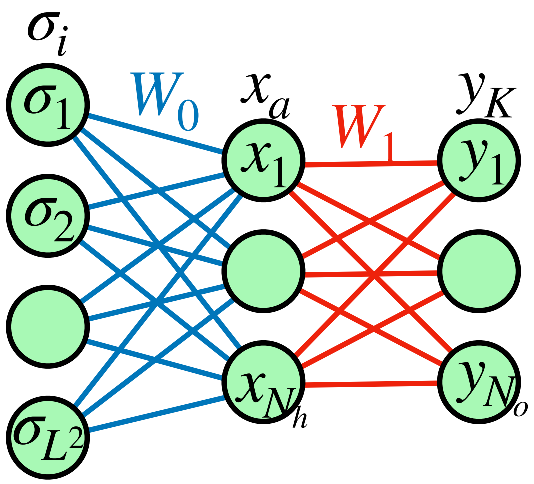
Q. Can AI detect phase transition?

With fewer information…

Juan Carrasquilla & Roger G. Melko (2017)



$$\beta = \frac{1}{T}$$



configurations, temperature

A. YES!

Can “AI” detect phase transition?

Neural net as a thermometer (Classification problem)

A.Tanaka AT 1609.09087
K. Kashiwa, Y. Kikuchi AT 1812.01522

Input = Ising configurations (by MCMC) with inverse temperature $\beta \in (\beta_{min} < \beta_{cr} < \beta_{max})$

Output = A class of temperature (Discretized inverse temperature)

NN is trained as a “thermometer”

Can “AI” detect phase transition?

Akio Tomiya

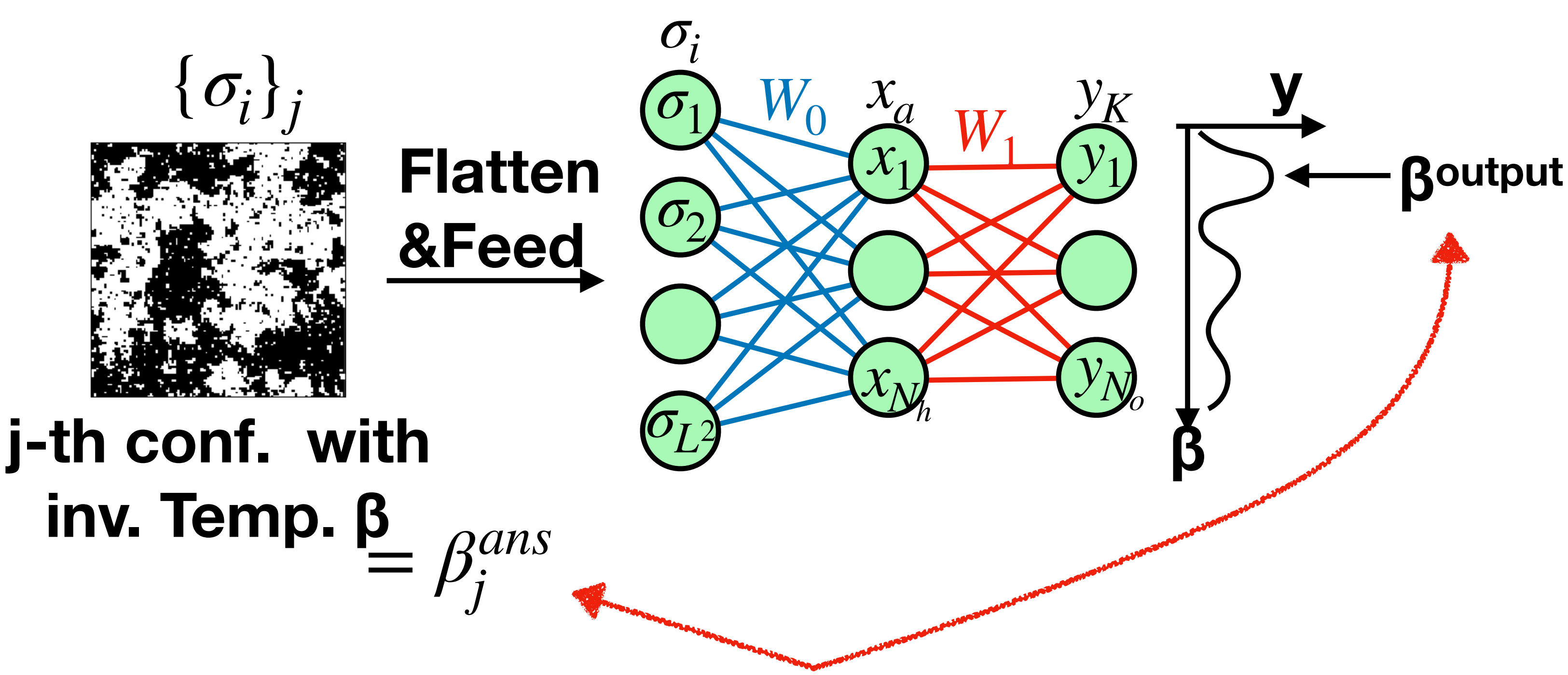
Neural net as a thermometer (Classification problem)

A.Tanaka AT 1609.09087
K. Kashiwa, Y. Kikuchi AT 1812.01522

Input = Ising configurations (by MCMC) with inverse temperature $\beta\in(\beta_{min}<\beta_{cr}<\beta_{max})$

Output = A class of temperature (Discretized inverse temperature)

NN is trained as a “thermometer”



$$
\left[\begin{array}{l}
\mathcal{I} = \left\{\{\sigma_i\} \middle| \text{ Ising configs on } L\times L \text{ lattice.}\right\} \\
\downarrow \left\{\begin{array}{l}\text{Fully-connected (Dense) layer} \quad {\color{blue}W_0} \\ \text{Softmax activation}\end{array}\right. \\
x_a \in [0,1]^{N_h} \ : \ \text{hidden units} \\
\downarrow \left\{\begin{array}{l}\text{Fully-connected (Dense) layer} \quad {\color{red}W_1} \\ \text{Softmax activation}\end{array}\right. \\
y_K \in [0,1]^{N_o} \ : \ \text{output}
\end{array}\right]
$$

$i = 1,\cdots,L\times L$

$a = 1,\cdots,N_h$

$K = 1,\cdots,N_o$

“Distance” (cross entropy)

$$E(y_K) \propto \sum_{j\in Conf} \delta_{\beta,\beta_j^{ans}} \times (-\log y_K(\{\sigma\}_j)) \qquad 0 < y_K < 1$$

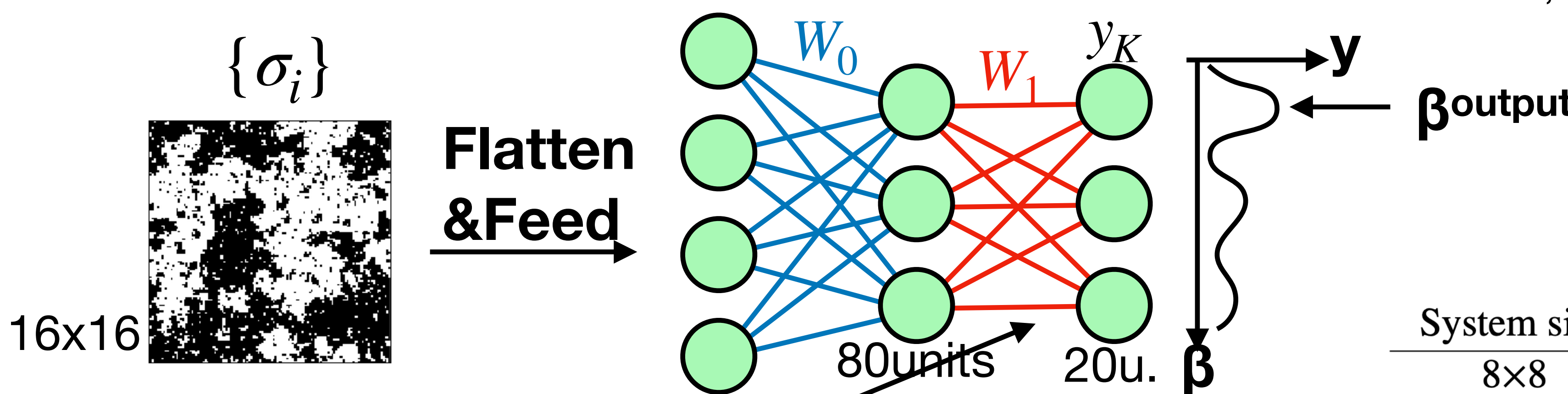
Minimize E, NN becomes a thermometer but we focus on W_1

Neural net captures phase transition

Akio Tomiya

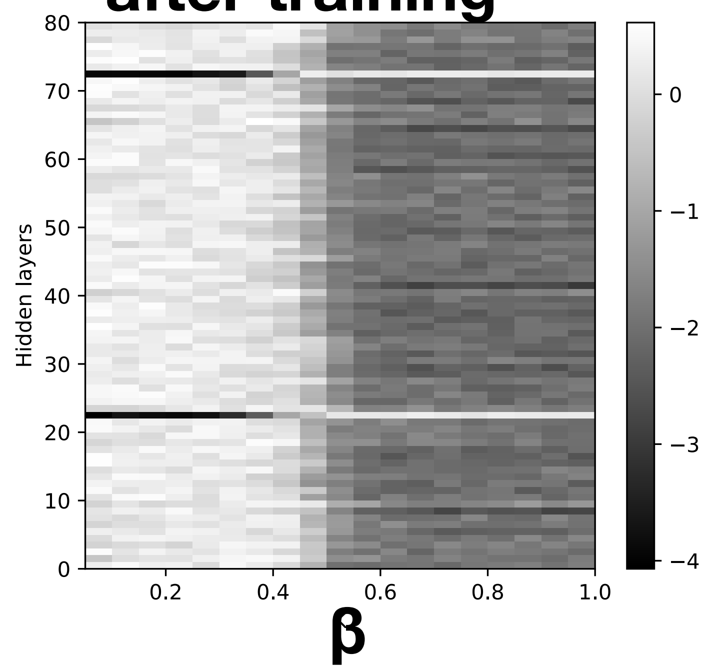
Heat map of weight W in second layer has structure

A.Tanaka AT 1609.09087
K. Kashiwa, Y. Kikuchi AT 1812.01522

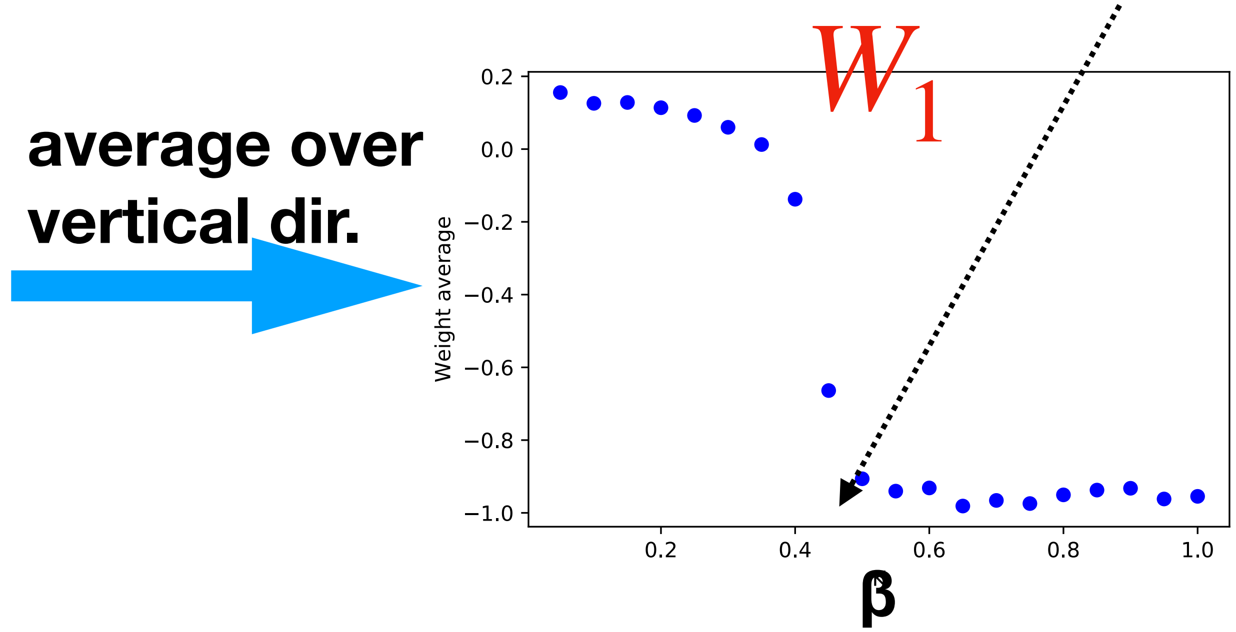


System size	β_c (CNN)	β_c (FC)
8×8	0.478915	0.462494
16×16	0.448562	0.433915
32×32	0.451887	0.415596
$L \to \infty$	$\beta_c^{\mathrm{Exact}} \sim 0.440686$	

Heat map of W_1
after training



average over
vertical dir.



After training, W₁ gets some pattern, especially there is a border around critical temp.

From detail analysis, output of W0 is correlated to magnetization

Summary for detection

After training, Neural network captures Tc

A.Tanaka AT 1609.09087
K. Kashiwa, Y. Kikuchi AT 1812.01522

- After training neural networks as a thermometer, it captures phase boundary
- Output of first layer is correlated to magnetization, so second layer gets a pattern.
- This framework actually works also for 3-states Potts model (skipped)
- If make it deeper with convolution layers <u>to improve the temperature prediction</u>, but <u>the pattern of weights becomes blurred</u>
- Applicability for gauge system? How can we input data?

Cf
P Shanahan, D Trewartha, W Detmold 1801.05784
S Wetzel, M Scherzer 1705.05582

2. Configuration generation

for gauge theory

arXiv: 1712.03893 (w/ A. Tanaka)
and **work in progress** (w/ A.Tanaka, Y. Nagai)

Markov chain Monte-Carlo

It enables us to calculate observables

- **Quantum fired theories, lattice QCD, are written by very high dimensional integral**

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

Markov chain Monte-Carlo

It enables us to calculate observables

Akio Tomiya

- **Quantum fired theories, lattice QCD, are written by very high dimensional integral**

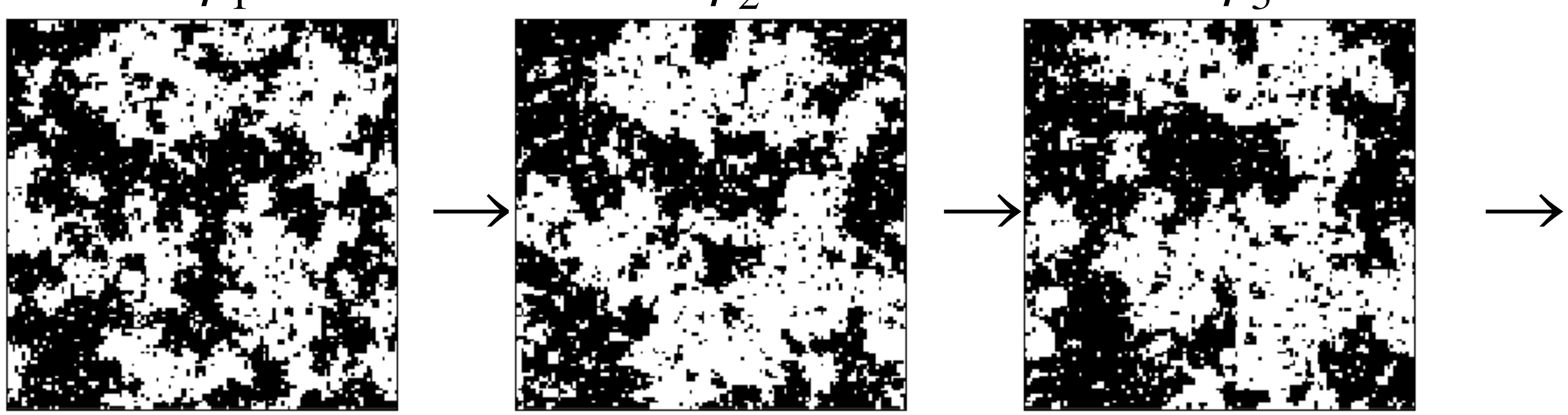
$$\langle O[\phi]\rangle = \frac{1}{Z}\int \underline{\mathscr{D}\phi} e^{-S[\phi]}\, O[\phi]$$

$$\mathscr{D}\phi = \prod_{i\in\{\mathbb{Z}/L\}^4} d\phi_i$$

: Very high dimensional integral
~ 10^4 dim

Markov chain with $P[\phi] = \frac{1}{Z} e^{-S[\phi]}$

ϕ_1 ϕ_2 ϕ_3



We can calculate expectation values by using Markov chain Monte-Carlo!

Markov chain Monte-Carlo

Akio Tomiya

It enables us to calculate observables

- **Quantum fired theories, lattice QCD, are written by very high dimensional integral**

$$\langle O[\phi] \rangle = \frac{1}{Z} \int \underline{\mathscr{D}\phi} e^{-S[\phi]}\, O[\phi]$$

$$\mathscr{D}\phi = \prod_{i \in \{\mathbb{Z}/L\}^4} d\phi_i$$

: Very high dimensional integral
~ 10^4 dim

$$= \frac{1}{N} \sum_{k}^{N} O[\phi_k] \pm O\left(\frac{1}{\sqrt{N}}\right)$$

Markov chain
Monte-Carlo with $P[\phi] = \frac{1}{Z} e^{-S[\phi]}$

- It is difficult to estimate expectation values using simple numerical integral like the trapezoid method.
- Markov chain Monte-Carlo can do it, independent to the dimensionality!
- IF a system has fermions, cost becomes expensive…
- We make this cheaper via "self-learning algorithm" in lattice gauge theory.

Exact algorithm is needed

Self-learning Monte Carlo (SLMC) is exact

SLMC for spin systems

J.Liu, Y.Qi, Z.Meng, L.Fu (arXiv:1610.03137)

Accept/Reject **Proposing part**

θ : tunable parameter = coupling

$$P(S_{k'}|S_k) = \min\left(1, \frac{e^{-\beta(H[S_{k'}]-H^\theta_{eff}[S_{k'}])}}{e^{-\beta(H[S_k]-H^\theta_{eff}[S_k])}}\right) \underline{Q^\theta_{eff}(S_{k'}|S_k)}$$

Corrected by modified Metropolis test

Update using effective model this must satisfy detailed balance

This is an exact algorithm: It gives correct configurations and if the effective model is far from the target system, acceptance is zero.

Intuitively,
Self-learning MC = Metropolis + reweighting on-fly + <u>update with tunable param θ.</u>

Other possibility: FLOW based model (M. S. Albergo et al.1904.12072)

Exact algorithm is needed

Self-learning Monte Carlo (SLMC)

SLMC for spin systems

J.Liu, Y.Qi, Z.Meng, L.Fu (arXiv:1610.03137)

Accept/Reject **Proposing part**

θ : tunable parameter = coupling

$$P(S_{k'}|S_k) = \min\left(1, \frac{e^{-\beta(H[S_{k'}]-H^{\theta}_{eff}[S_{k'}])}}{e^{-\beta(H[S_k]-H^{\theta}_{eff}[S_k])}}\right) \underline{Q^{\theta}_{eff}(S_{k'}|S_k)}$$

Corrected by modified Metropolis test

Update using effective model this must satisfy detailed balance

This is an exact algorithm:
if the effective model far from the system, acceptance is zero.

Testcase

$$H = -J\sum_{\langle ij\rangle} S_iS_j - K\sum_{ijkl\in\square} S_iS_jS_kS_l,$$

No "efficient" update because of 2nd term

$$H_{\rm eff} = E_0 - \tilde{J}_1 \sum_{\langle ij\rangle_1} S_iS_j$$

$$S_i = \pm 1$$

Ising model with parameter $\tilde{J}_1$
, which is determined by fitting!
(no fancy ML is needed!)
This has effective update

Exact algorithm is needed

Self-learning Monte Carlo (SLMC)

Akio Tomiya

SLMC for spin systems

J.Liu, Y.Qi, Z.Meng, L.Fu (arXiv:1610.03137)

Accept/Reject **Proposing part**

θ : tunable parameter = coupling

$$P(S_{k'}|S_k) = \min\left(1, \frac{e^{-\beta(H[S_{k'}]-H^{\theta}_{eff}[S_{k'}])}}{e^{-\beta(H[S_k]-H^{\theta}_{eff}[S_k])}}\right) Q^{\theta}_{eff}(S_{k'}|S_k)$$

Corrected by modified Metropolis test

Update using effective model this must satisfy detailed balance

This is an exact algorithm.

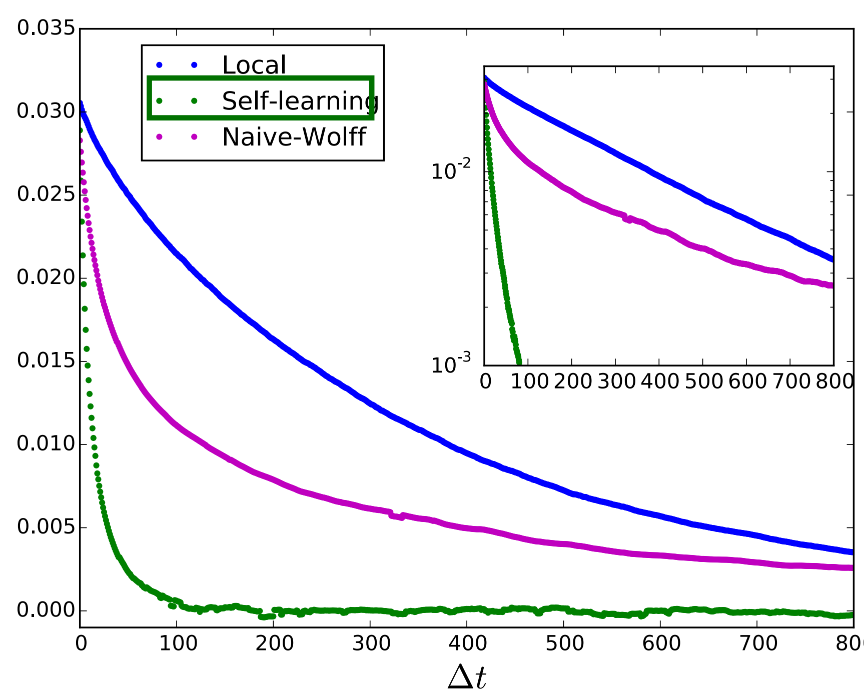
Testcase

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

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

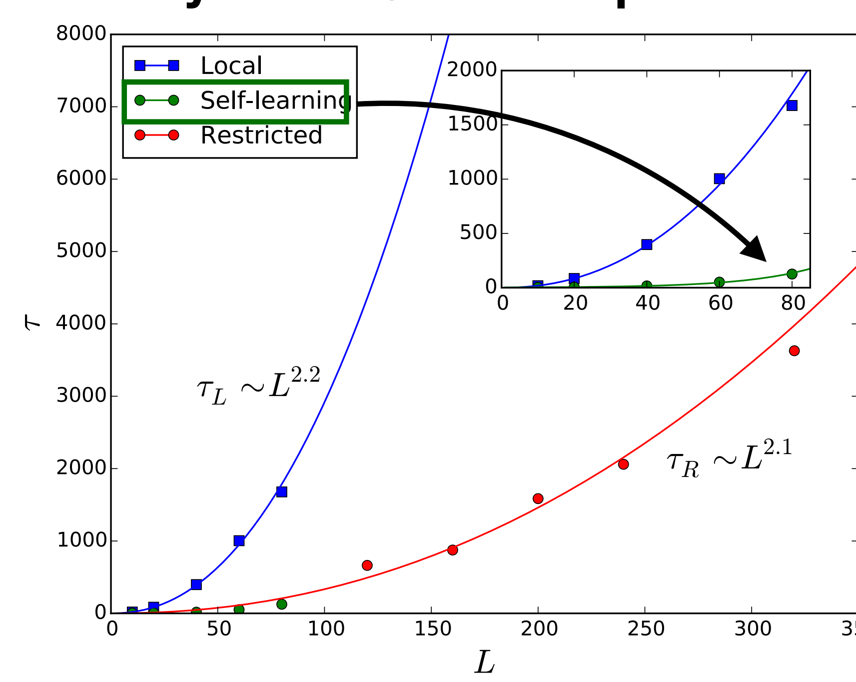
$$S_i = \pm 1$$

Autocorrelation function



24 time efficient

Dynamic Critical exponent



Very mild scaling

Exact algorithm is needed

QCD with Self-learning Monte Carlo

work in progress

Collaborate with
Akinori Tanaka (Riken AIP/ iTHENS)
Yuki Nagai (JAEA/ RIKEN AIP)

SLMC for lattice QCD

θ : tunable parameter = coupling

$$P(U_{k'} | U_k) = \min\left(1, \frac{e^{-(S[U_{k'}]-S^{\theta}_{eff}[U_{k'}])}}{e^{-(S[U_k]-S^{\theta}_{eff}[U_k])}}\right) Q^{\theta}_{eff}(U_{k'} | U_k)$$

Setup: SU(2) plaquette action + staggered quarks with ma = 0.5
Effective action = hopping parameter expanded action = pure-gluonic, heatbath

$$S[U] = \beta_{\text{pl}} \square + \bar{\psi}(D_{\text{stag}} + m)\psi$$

Our choice: $$S_{eff}[U] = \tilde{\beta}_{\text{pl}} \square + \tilde{\beta}_{\text{rec}} \square\!\square \cancel{+ \tilde{\beta}_*} + \cdots$$

Parameters determined by HMC with linear regression
or we can use SLMC (“self-learning” way of use)

Preliminary result

QCD with Self-learning Monte Carlo

work in progress

Collaborate with
Akinori Tanaka (Riken AIP/ iTHENS)
Yuki Nagai (JAEA/ RIKEN AIP)

SLMC for lattice QCD

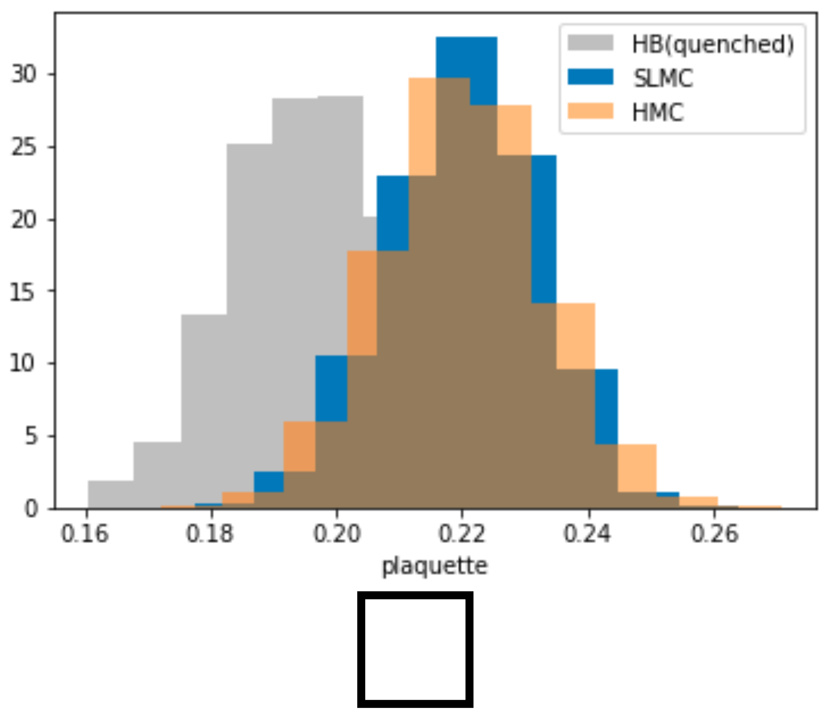
θ : tunable parameter = coupling

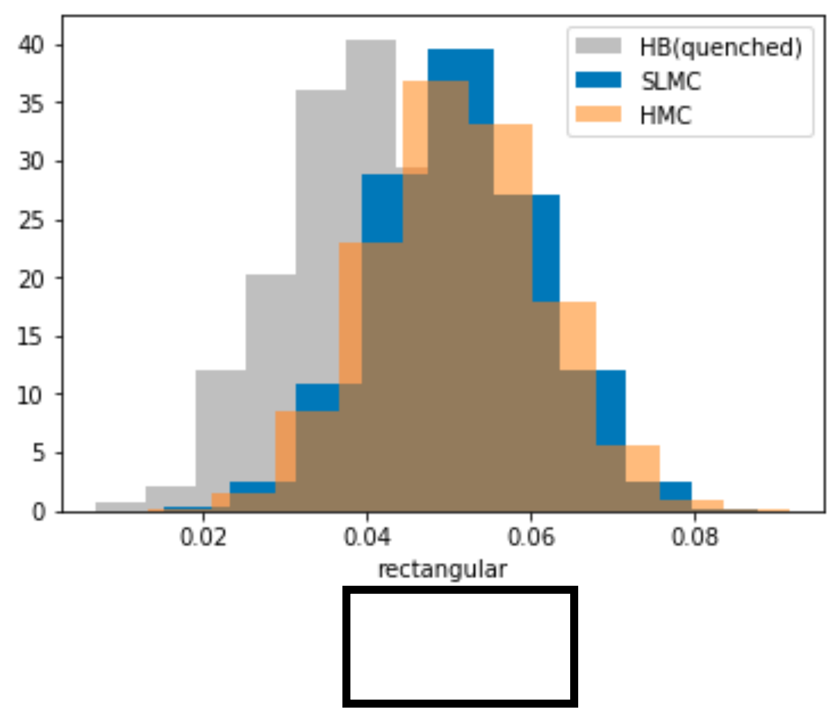
$$P(U_{k'} | U_k) = \min\left(1, \frac{e^{-(S[U_{k'}] - S^{\theta}_{eff}[U_{k'}])}}{e^{-(S[U_k] - S^{\theta}_{eff}[U_k])}}\right) Q^{\theta}_{eff}(U_{k'} | U_k)$$

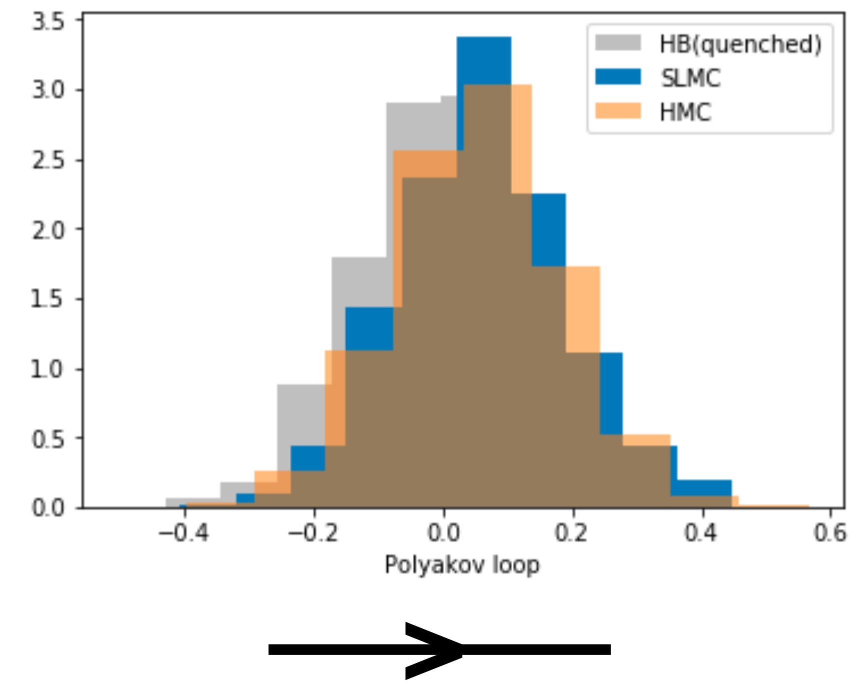
Setup: SU(2) plaquette action + staggered quarks with ma = 0.5
Effective action = hopping parameter expanded action = pure-gluonic, heatbath

Observables（■=SLMC, ■=HMC, ■=quenched）

ma=0.5, L=4







So far so good

3. QCD spectral function via sparse modeling

Work in progress, very preliminary

A. Tomiya

QCD spectral function?

It contains everything, but we cannot obtain

Two point functions G(τ) can be calculate on the lattice,

$$G(\tau) = \langle O(\tau) O^\dagger(0) \rangle \,.$$

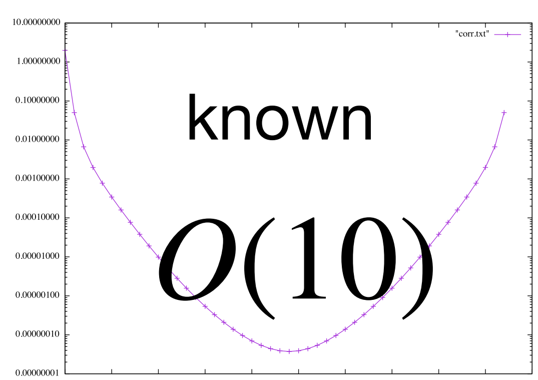
Corresponding QCD spectral function ρ(ω) contains
every information of QCD for that channel,

$$G(\tau) = \int d\omega K(\tau,\omega)\rho(\omega) \qquad \bar{K}(\tau,\omega) \sim \text{cosh: kernel}$$

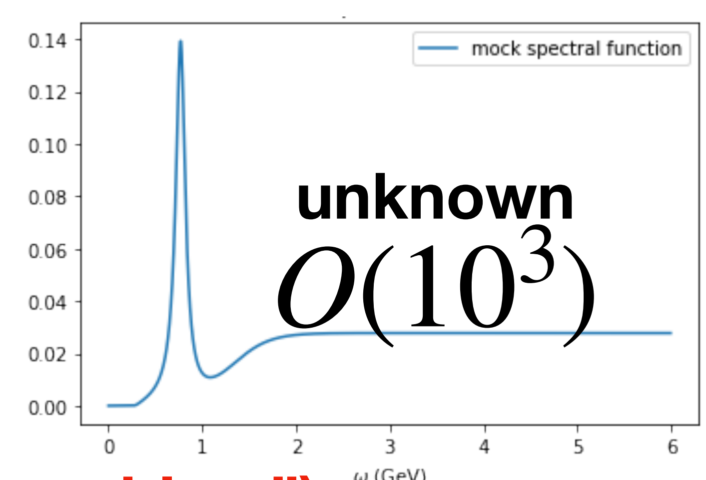
Practically, we can not obtain ρ because,

Discretize

$$G_\tau = \sum_\omega K_{\tau,\omega} \rho_\omega$$



$$\boxed{G} = \overset{\text{known}}{\boxed{K}} \cdot \boxed{\rho}$$



Lack of information to determine ρ from G ("ill-posed problem")

Maxima entropy method (MEM; Asakawa et al.) has been used = Bayesian analysis

Sparse Modeling?

Akio Tomiya

Fitting with L1 regulator (=LASSO, least absolute shrinkage and selection operator)

Apply the singular value decomposition (also used in MEM),

$$K = USV^\top$$

U, V **Orthogonal (unitary) mat.**

S **Rectangular diagonal matrix**

$$\begin{cases} \tilde{G} = U^\top G \\ \tilde{\rho} = V^\top \rho \end{cases}$$

Intermediate rep. (IR basis)

$$\tilde{G}_\tau = S_{\tau,\omega}\tilde{\rho}_\omega \quad \Longleftrightarrow \quad \tilde{G} = \begin{pmatrix} s_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & s_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & s_3 & 0 & 0 & 0 \end{pmatrix} \cdot \tilde{\rho}$$

$$\chi^2(\tilde{\rho}) = |\tilde{G} - S\tilde{\rho}|_2^2$$

Naive chi square
Overfit the noisy data

Modify

$$L(\tilde{\rho};\lambda) = |\tilde{G} - S\tilde{\rho}|_2^2 + \lambda|\tilde{\rho}|_1$$

Chi square with L1 regulator (LASSO) using Lagrange mult.

And minimize this L.

Why L1 regulator works?

Akio Tomiya

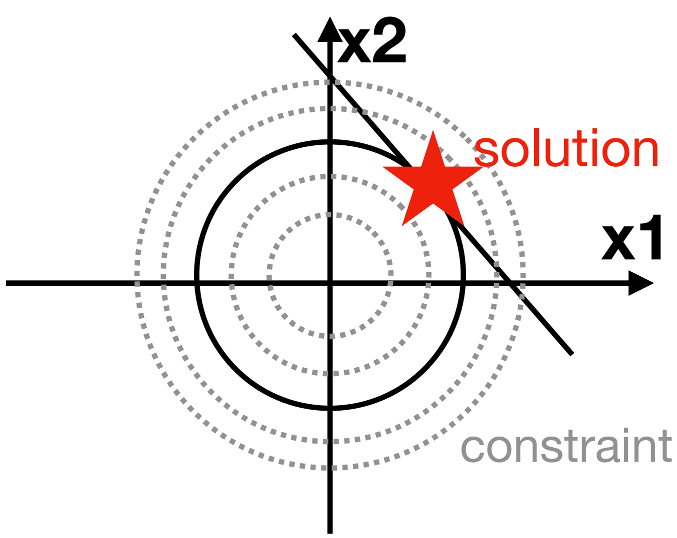
L1 regulator can kill ambiguities well = sparseness

Example:

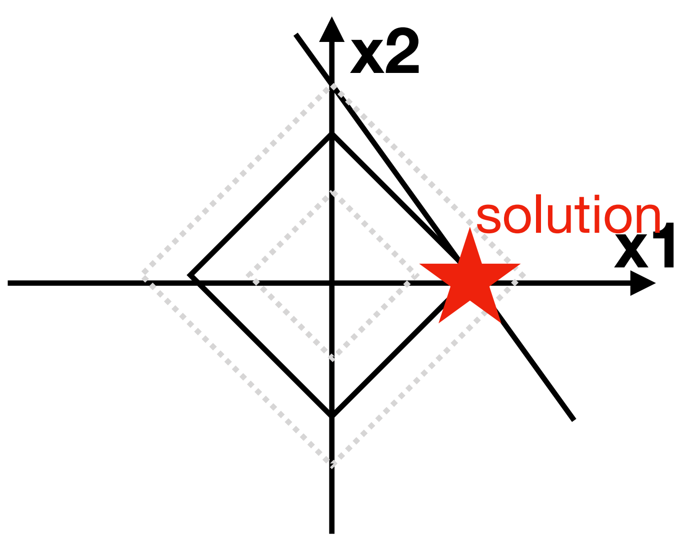
Given Eq. $(a_1 \quad a_2)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \textbf{constant}$

	w/ L2 constraint	w/ L1 constraint
"Constraint"	$\min_{x_i} \lvert \vec{x} \rvert_2$	$\min_{x_i} \lvert \vec{x} \rvert_1$
Definition	$\lvert \vec{x} \rvert_2 = \sqrt{x_1^2 + x_2^2}$	$\lvert \vec{x} \rvert_1 = \lvert x_1 \rvert + \lvert x_2 \rvert$

Solution:



x1≠0, x2≠0



x2 = 0 (sparse)

Even if the equation has statical noise, a solution with L1 constraint is robust (insensitive to noise)

Sparseness and LASSO

Minimize chi-square + L1 constraint in SVD basis

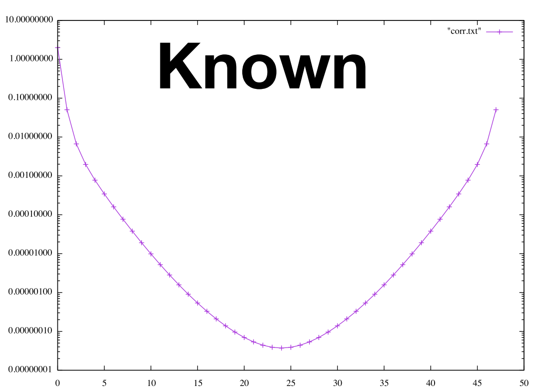
Akio Tomiya

Ohtsuki et al. 2017
for cond. mat.

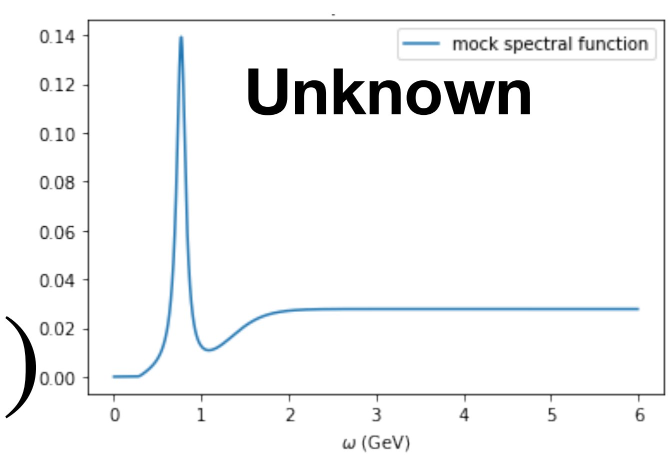
Original Problem:

We want to determine O(1000) points of ρ from O(10) data

This means, O(10) points of ρ in some basis can be determined
because we don't have information.
BUT, in SVD basis, the spectral function is sparse!



$$\underbrace{\tilde{G}}_{\text{known},\ O(10)} = \underbrace{\begin{pmatrix} s_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & s_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & s_3 & 0 & 0 & 0 \end{pmatrix}}_{\text{known}} \cdot \underbrace{\tilde{\rho}}_{\text{unknown},\ O(10^3)}$$



So, ρ can be obtained by minimizing,

$$L(\tilde{\rho};\lambda) = |\tilde{G} - S\tilde{\rho}|_2^2 + \lambda|\tilde{\rho}|_1$$

and,

$$\rho = V\tilde{\rho}$$

(In practice, we add positivity constraint for ρ to L.)

(very) Preliminary results

Akio Tomiya

Mock data + noise: it is well reconstructed …?

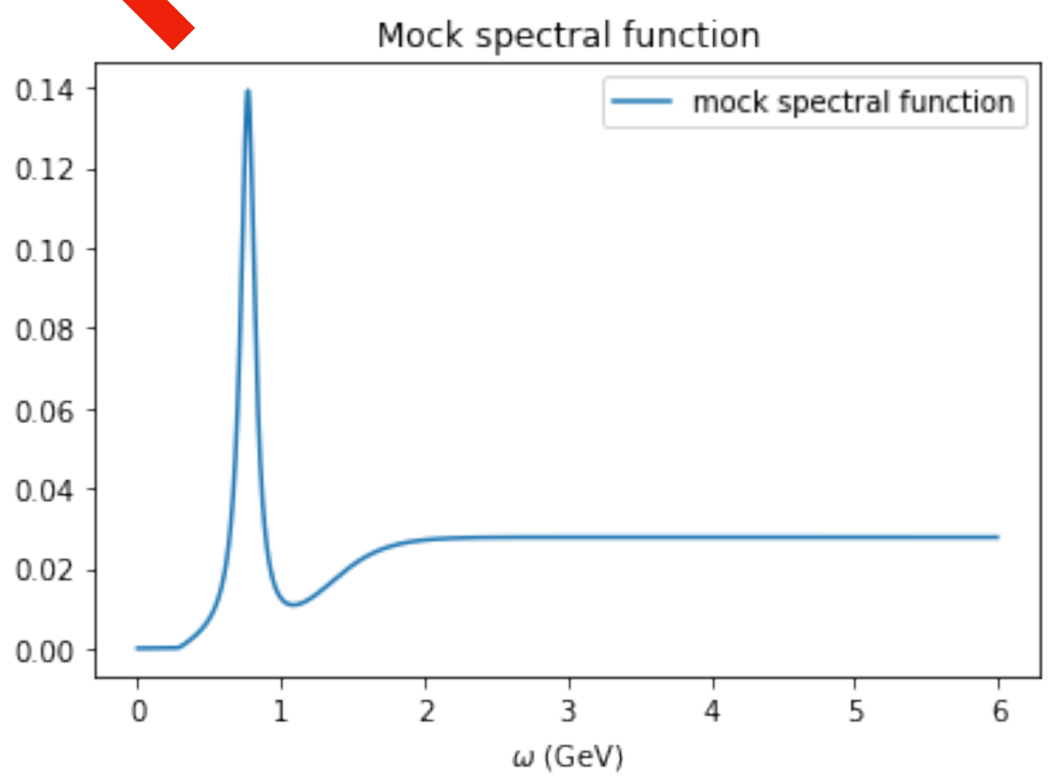
Mock data (vector ch.) from PRD65, 014501(CP-PACS), noise level from Asakawa et al.

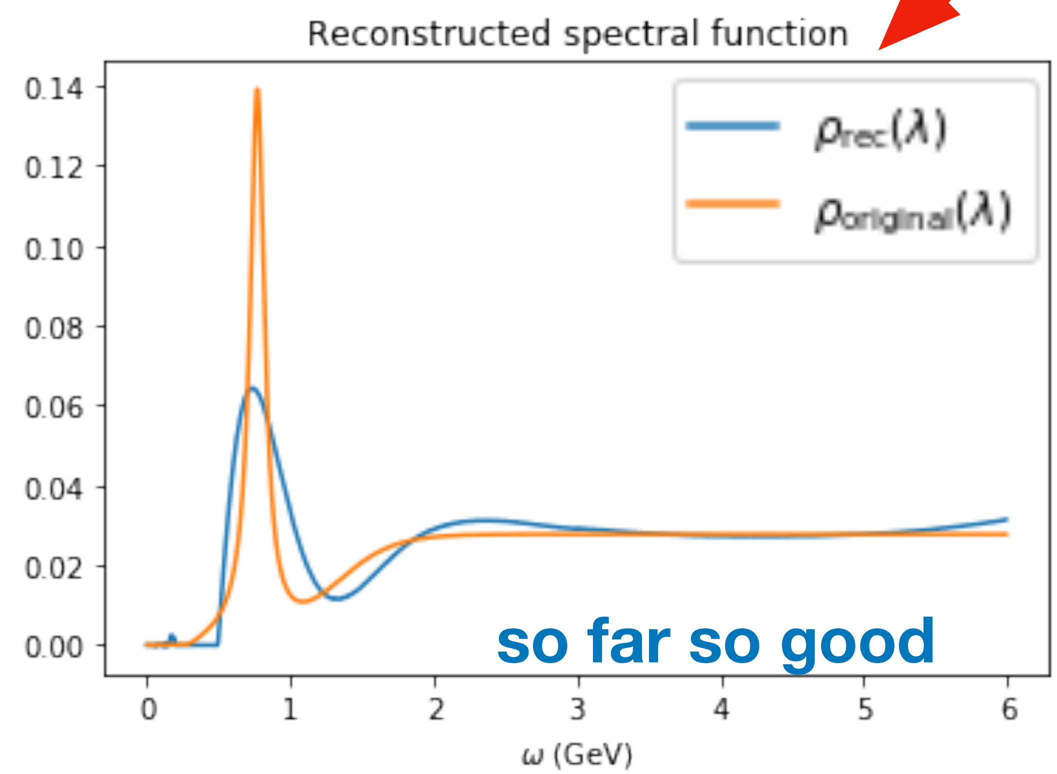
$$\rho_{in}(\omega) = \frac{2}{\pi}\left[F_\rho^2 \frac{\Gamma_\rho(\omega) m_\rho}{(\omega^2 - m_\rho^2)^2 + \Gamma_\rho^2(\omega) m_\rho^2} + \frac{1}{8\pi}\left(1 + \frac{\alpha_s}{\pi}\right)\frac{1}{1 + e^{(\omega_0 - \omega)/\delta}}\right].$$

$$\Gamma_\rho(\omega) = \frac{1}{48\pi}\frac{m_\rho^3}{F_\rho^2}\left(1 - \frac{4m_\pi^2}{\omega^2}\right)^{3/2}\theta(\omega - 2m_\pi).$$

$$m_\rho = 0.77,\quad m_\pi = 0.14,\quad F_\rho = 0.142,$$
$$\omega_0 = 1.3,\quad \delta = 0.2,\quad \alpha_s = 0.3,$$

ρ(mock data) $\xrightarrow[\text{w/ kernel}]{\text{integration}}$ **G** $\xrightarrow[\text{“Mimic MC”}]{\text{+Gaussian noise}}$ **G+noise** $\xrightarrow[\text{in SVD basis}]{\text{Sparse modeling}}$ **ρ(reconstruct)**





$$|\rho_{rec}(\omega) - \rho_{mock}(\omega)|_2 = 0.1071$$

Summary

Machine learning provides us new techniques

1. Neural network can detect phase transition in classical spin chain
2. SLMC can generate meaningful gauge configurations
3. Sparse modeling can reconstruct QCD spectral function (for mock data though)

Todo:

5. Application of SLMC to physical system; right top corner of the Columbia plot
6. Improve SLMC by adding more and more terms, neural net may help + extend to SU(3)
7. More test on sparse modeling and apply to real lattice qcd data

Comment:
- Our community should discuss systematic error from ML techniques, if it is not exact (Benefit of LQCD is quantitativity).
- How can we control or evaluate error?

Thanks!

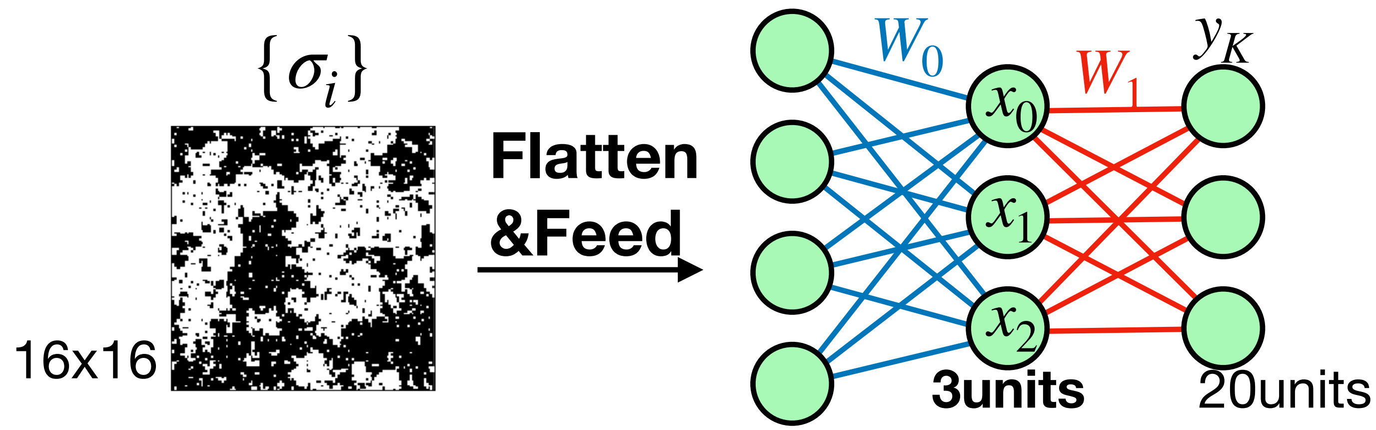
Backup

Results 2/2

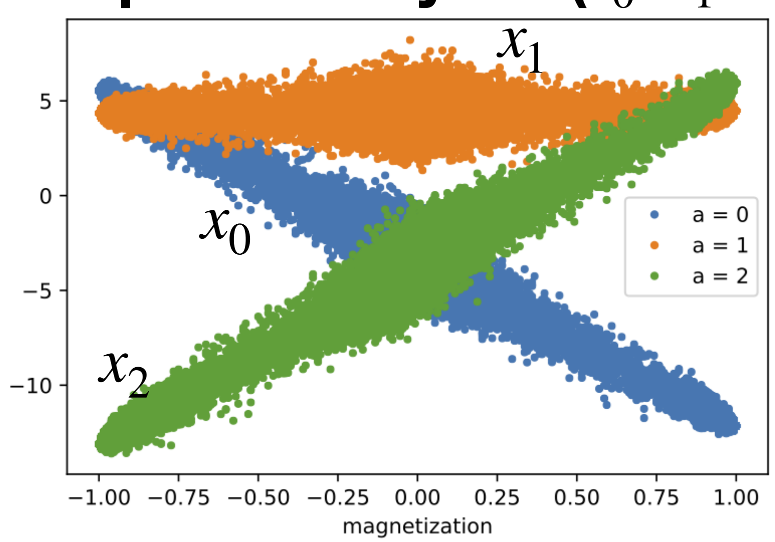
Akio Tomiya

Output of first layer ~ Magnetization

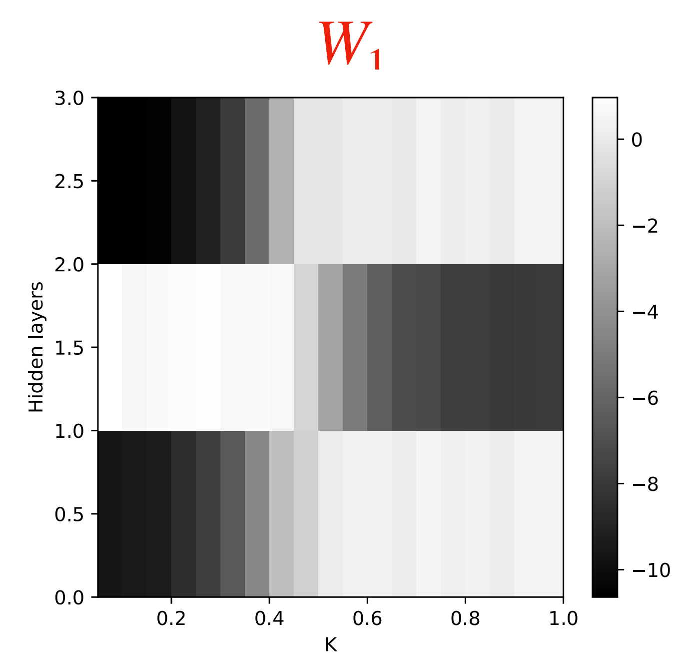
K. Kashiwa, Y. Kikuchi AT 1812.01522



Output 1st layer= (x_0 x_1 x_2)



Scatter plot for output of 1st layer and magnetization
-> x is correlated to the magnetization
(automatically captured)



This means, W_1 represents correlation between temperature and magnetization!

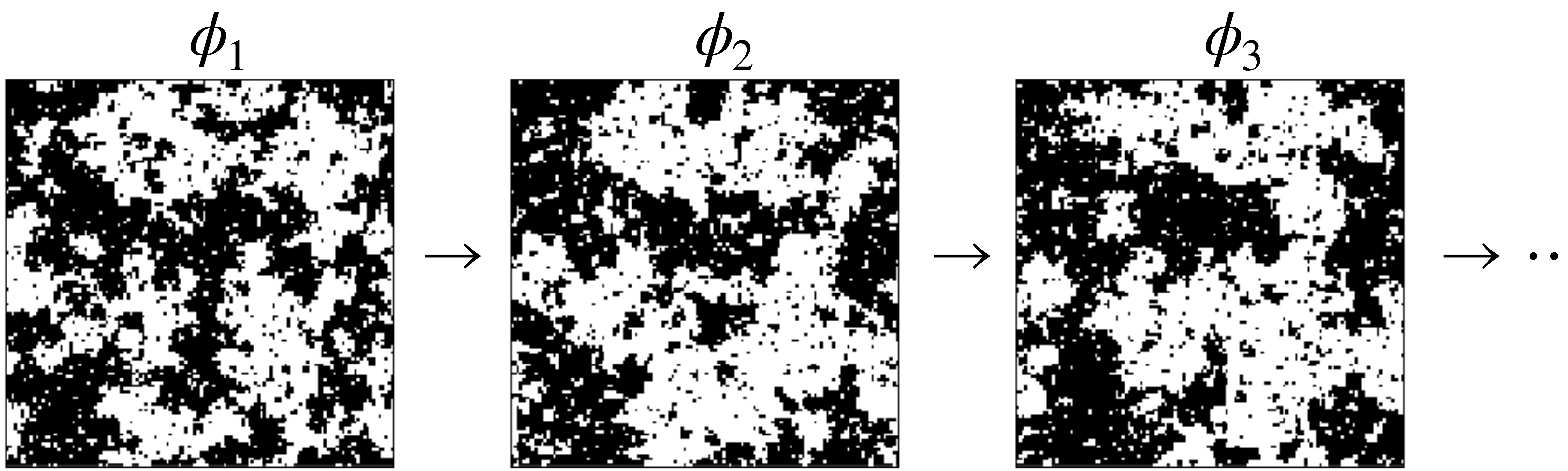
Markov chain Monte-Carlo

It has inefficiency from correlation between samples

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

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

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



Γ or τ_{ac} measures similarity of configurations

Effect of long autocorrelation

Akio Tomiya

Autocorrelation makes signals/noise ratio bad

Data from
Nf=3, standard staggered
with magnetic field

$$L^3 \times N_t = 16^3 \times 4$$
$$ma = 0.03$$

β	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 we find an algorithm with smaller z (or shorter τ_{ac}),
it enables us precise/large scale research around the critical regime!

Markov chain Monte-Carlo?

Akio Tomiya

If detailed balance satisfied, we can sample using it

A key concept is the detailed balance condition:

If an update algorithm P(.|.) satisfies

$$P(\phi_{k'} | \phi_k) e^{-S[\phi_k]} = P(\phi_k | \phi_{k'}) e^{-S[\phi_{k'}]}$$

it will give configurations with a desired distribution (skip proof)

$$P_{eq}(\phi) = \frac{1}{\int \mathcal{D}\phi' e^{-S[\phi']}} e^{-S[\phi]}$$

We have to fight for autocorrelation!

Akio Tomiya

Machine Learning techniques could reduce autocorrelation

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

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

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

τ_{ac} is given by an update algorithm (N. Madras et. al 1988)

- Correlation between generated configurations are estimated by autocorrelation time τ_{ac}
- Autocorrelation time τ_{ac} depends on an update algorithm
- If τ_{ac} becomes half, statistics becomes effectively double in same cost in time!

I attempt to generate configurations using machine learning!

Exact algorithm is needed

QCD with Self-learning Monte Carlo

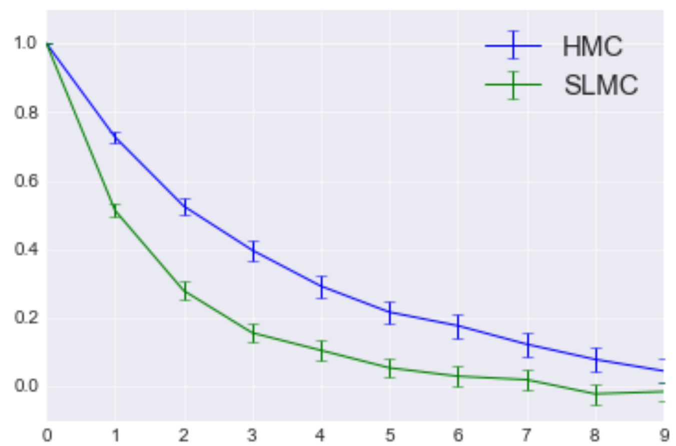
work in progress

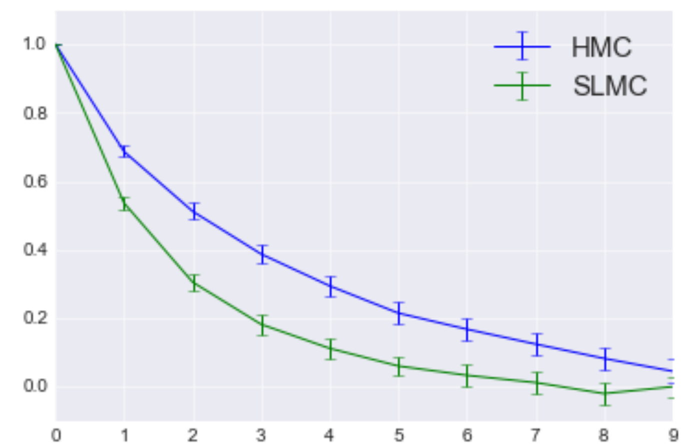
Collaborate with
Akinori Tanaka (Riken AIP/ iThems)
Yuki Nagai (JAEA/ RIKEN AIP)

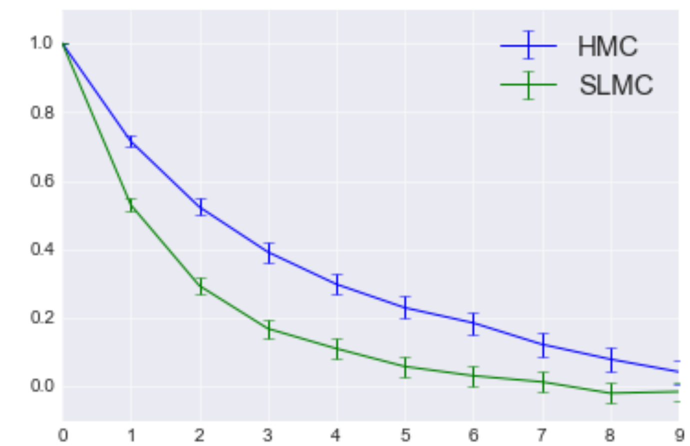
$$P(U_{k'} | U_k) = \min\left(1, \frac{e^{-(S[U_{k'}]-S_{eff}[U_{k'}])}}{e^{-(S[U_k]-S_{eff}[U_k])}}\right) Q_{eff}(U_{k'} | U_k)$$

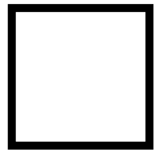
Setup: SU(2) plaquette action + staggered quarks
Effective action = hopping parameter expanded action, heatbath

Autocorrelation（■=HMC, ■=SLMC）

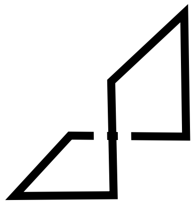








Also good



Exact algorithm is needed

QCD with Self-learning Monte Carlo

work in progress

Collaborate with
Akinori Tanaka (Riken AIP/ iThems)
Yuki Nagai (JAEA/ RIKEN AIP)

$$P(U_{k'}|U_k) = \min\left(1, \frac{e^{-(S[U_{k'}]-S_{eff}[U_{k'}])}}{e^{-(S[U_k]-S_{eff}[U_k])}}\right) Q_{eff}(U_{k'}|U_k)$$

Setup: SU(2) plaquette action + staggered quarks
Effective action = hopping parameter expanded action, heatbath

Acceptance	# of operation of $(D[U]+m)^{-1}$
HMC : 90%	**26**
SLMC : 60%	**1**

Bad news: now it works only for m > 1
Stay tuned

QCD spectral function?

QCD with Self-learning Monte Carlo

$$G(\tau) = \int d\omega \bar{K}(\tau, \omega)\rho(\omega) \qquad \bar{K}(\tau, \omega) = \omega^2(e^{-\tau\omega} + e^{-(N_\tau - \tau)\omega})$$

$$G_\tau = \sum_\omega \bar{K}_{\tau,\omega}\rho_\omega$$

$$G_\tau = \bar{K}_{\tau,\omega} \cdot \rho_\omega$$

$O(10)$ $\qquad$ $O(10^3)$

Introduction

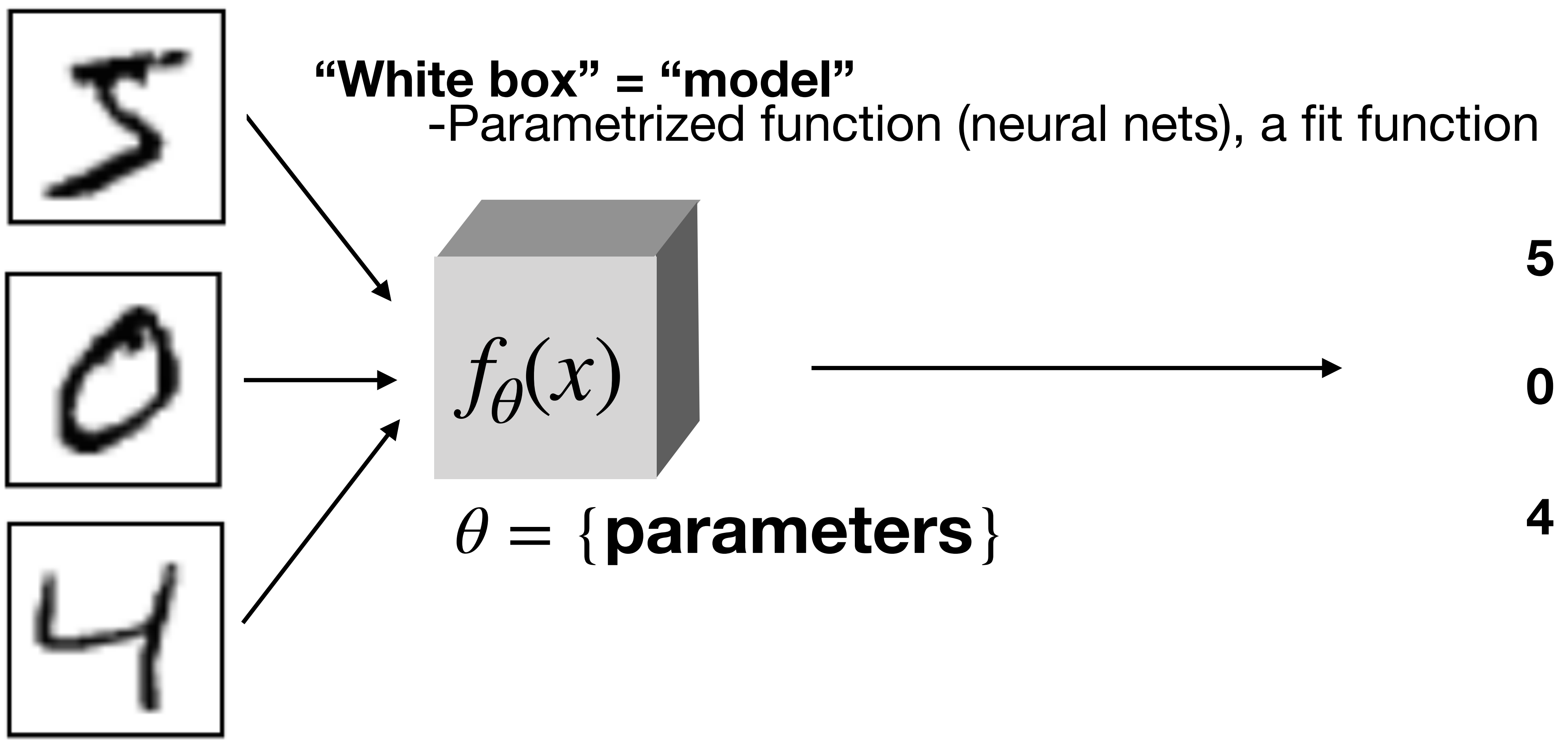
What does “Machine learning” give us?

What supervised learning does (~ deep learning, neural nets):

Introduction

What does “Machine learning” give us?

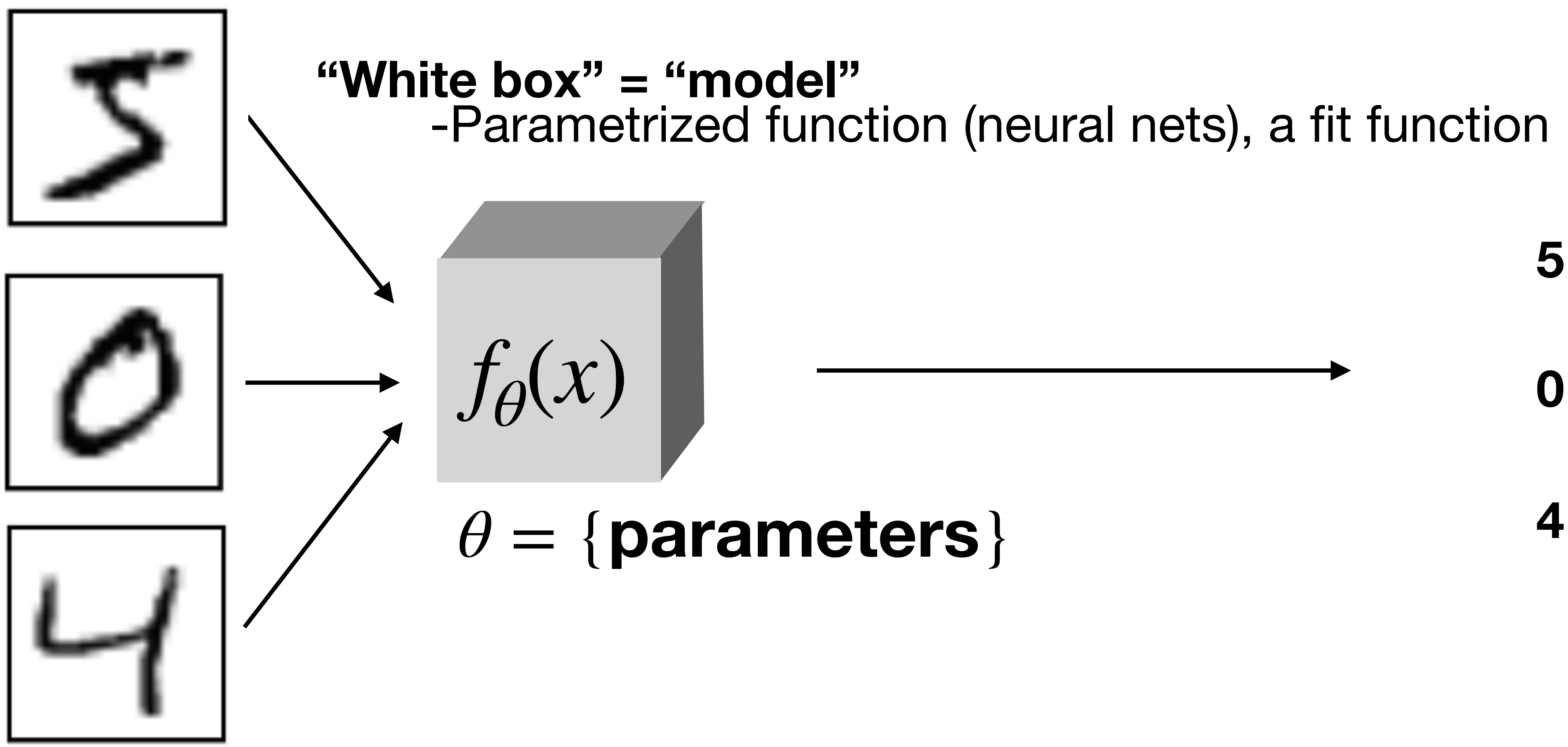
What supervised learning does (~ deep learning, neural nets):



Introduction

What does “Machine learning” give us?

What supervised learning does (~ deep learning, neural nets):



$$D_{KL}(\theta) \geq 0$$: “Distance between correct answer to current output”
(cf: chi square in fitting, θ = a set of parameters)

- Machine learning (∋ deep learning) basically, make a “map” between data and output
- It could be deterministic (neural nets) or stochastic (generative models, later)

Unsupervised learning in a nutshell

Akio Tomiya

Information comes from a probability distribution

$$\partial_\theta D_\theta(P \| P_\theta) = -\int \mathscr{D}x P(x) \partial_\theta \log(P_\theta(x))$$

$$P_\theta[x] = \frac{1}{Z_\theta} e^{-H_\theta[x]}$$

$$= -\int \mathscr{D}x P(x) \partial_\theta(-H_\theta[x] - \log Z_\theta)$$

$$= \int \mathscr{D}x P(x)(\partial_\theta H_\theta[x] + \frac{1}{Z_\theta}(\int \mathscr{D}y \partial_\theta e^{-H_\theta[y]}))$$

$$= \int \mathscr{D}x P(x) \partial_\theta H_\theta[x] - \frac{1}{Z_\theta} \int \mathscr{D}y H_\theta[y] e^{-H_\theta[y]}$$

Unsupervised learning in a nutshell

Akio Tomiya

Inverse of Ising model ~ Boltzmann machine

Eg. Generalized Ising model (spin glass)

$$H_\theta[\sigma] = \beta \sum K_{ij}\sigma_i\sigma_j$$

σ_i: **spin**

K_{ij}: **coupling**

What physicists want to do ↓

↑ **Boltzmann machine: guess a set of parameters θ**

$$\langle O[\sigma] \rangle_\theta = \frac{1}{Z_\theta} \sum_{\{\sigma\}} e^{-H_\theta[\sigma]}$$

$\theta = \{K_{ij}, \beta |$ **parameters** $\}$

Unsupervised learning in a nutshell

Unsupervised

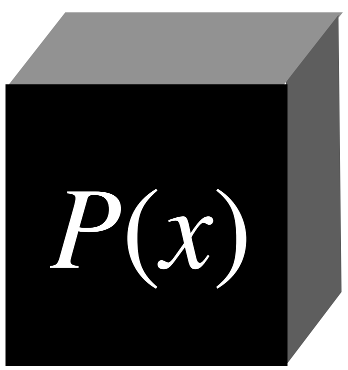
Eg. Generalized Ising model (spin glass)

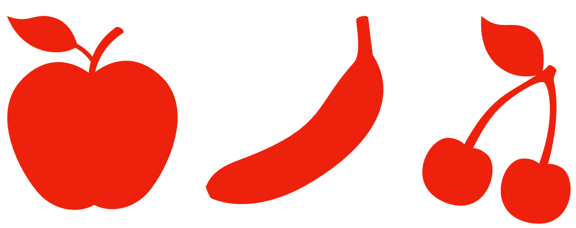
$$P_\theta[\sigma] = \frac{1}{Z_\theta} e^{-H_\theta[\sigma]}$$

σ_i: **spin**
K_{ij}: **parameters**

Generative models fit (guess) the coupling

sampling(some easier way)





- Once determine a coupling K (= training), we can use it.
- Sampling from parametrized distribution is needed