深層学習と物理学

新野康彦

奈良工業高等専門学校

2024 年 3 月 16 日

この話の流れ

1. はじめに
   - AI とは
   - 情報科学の歴史
   - 深層学習とは
2. ニューラルネットワーク
   - 脳の神経細胞と数理モデル
   - ニューラルネットの学習法
   - 誤差逆伝播法
3. 物理学への応用
   - 量子色力学の符号問題への機械学習的アプローチ [1]
   - ~~深層学習は統計系の配位から何をどう学ぶか~~ [2]

[1] Y. Mori, K. Kashiwa and A. Ohnishi, PTEP 2018(2), 023B04
[2] K-I. Aoki, T. Fujita and T. Kobayashi, J. Phys. Soc. Jpn **88** 054002

1. はじめに

AI とは

AI: Artificial Intelligence の略. 人間の知的活動をコンピューター上で実現したもの.

John McCarthy: 知的な機械, 特に知的なコンピュータープログラムを作る科学や工学全般

Alan M. Turing: 人間の質問者がその”モノ”と会話をして, その相手が人間か機械か判別できないときの, その”モノ”のこと (チューリング · テスト)

[2] AI を題材とした映画としては「エクス・マキナ」(2015 年 イギリス) が面白い.

情報科学の歴史

第 1 次 AI ブーム (1950∼1970 年代): **探索と推論の時代**

1956 年: ダートマス会議での人工知能の提唱
1958 年: F. Rosenblatt によるパーセプトロンの考案
1969 年: M. L. Mnski & S. A. Papert(パパート)による「パーセプトロン」の限界の提示 ⇒ 冬の時代へ

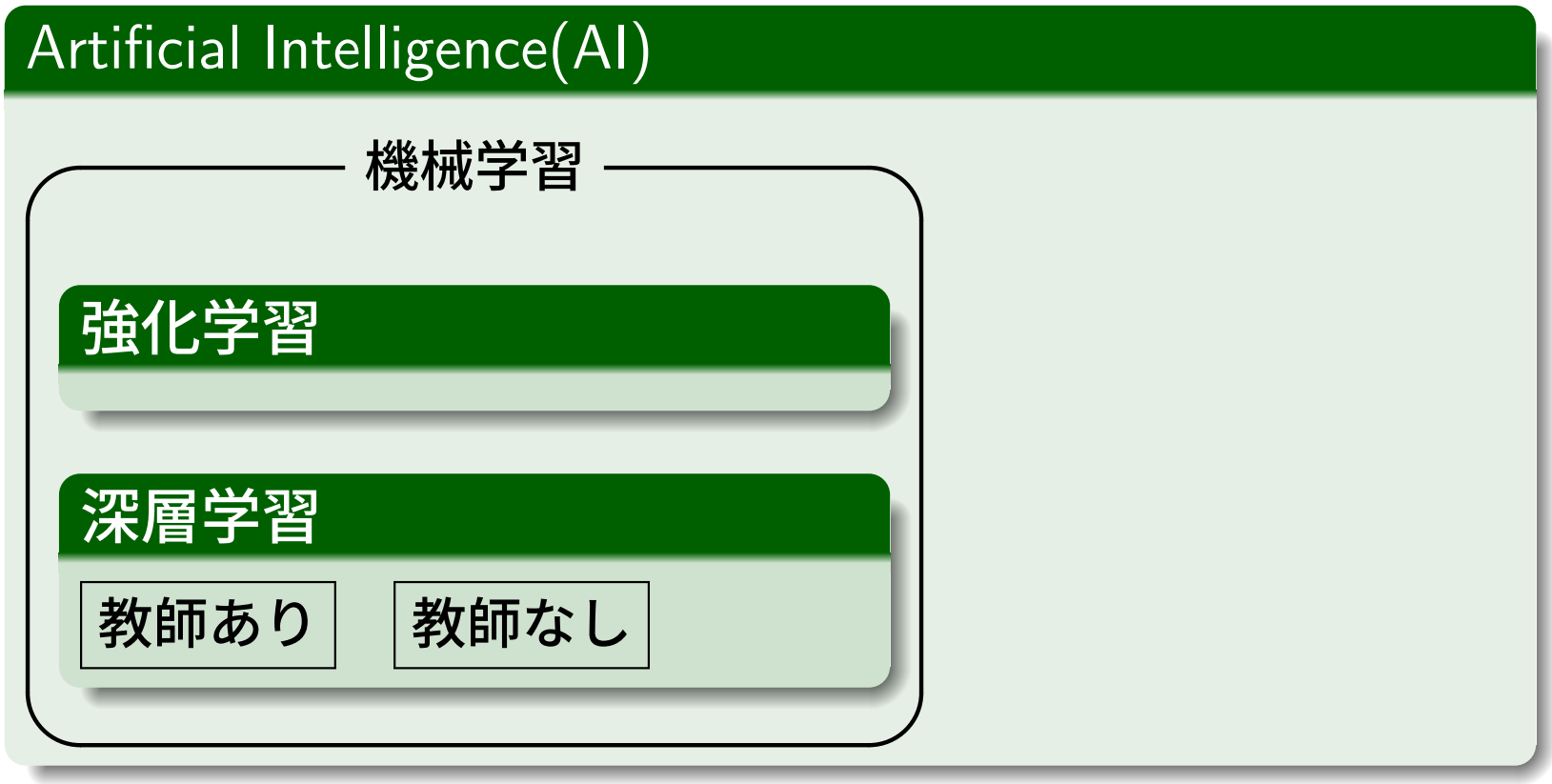
第 2 次 AI ブーム (1980∼1990 年代): **知識表現の時代**

1979 年: 福島邦彦氏によるネオコグニトロン (畳み込みニューラルネット) の提唱
1980 年代: エキスパートシステムの流行
1982 年: J. J. Hopfeld による Hopfield 模型 (再帰的ニューラルネットワーク) の提唱
現実問題への適用限界 ⇒ 冬の時代へ

第 3 次 AI ブーム (2000 年代 ∼): **機械学習 · 深層学習の時代**

World Wide Web の発明とビッグデータの収集 · 取得
GPU(Graphical Processing Unit) の発展
コンピューターによる自律的な特徴量の導出

深層学習とは



機械学習: コンピューターに大量のデータを読み込ませ，自ら学習することでルールやパターンを発見する技術.

深層学習: 機械学習の手法の 1 つであるニューラルネットワークを多層構造に構築したもの.

$\longrightarrow$ 深層学習の発展

画像認識: 2012 年, 深層学習の登場で認識精度が高精度化 (認識誤差 26%$\rightarrow$16%へ), 2015 年に人間の認識能力 (認識誤差 5%程度) を超える

音声認識: 音声認識誤差を 30%前後まで改善 $\rightarrow$ スマートフォン等で音声認識が一般的に利用可能へ
2016 年 10 月, Microsoft が音声認識技術において人並みの性能を実現したと発表

囲碁: 2015 年 10 月, Google 傘下の Deep Mind が開発した囲碁プログラム Alpha Go がプロ棋士に勝利
2017 年 5 月, 世界棋士レート 1 位の柯潔氏に 3 局全勝

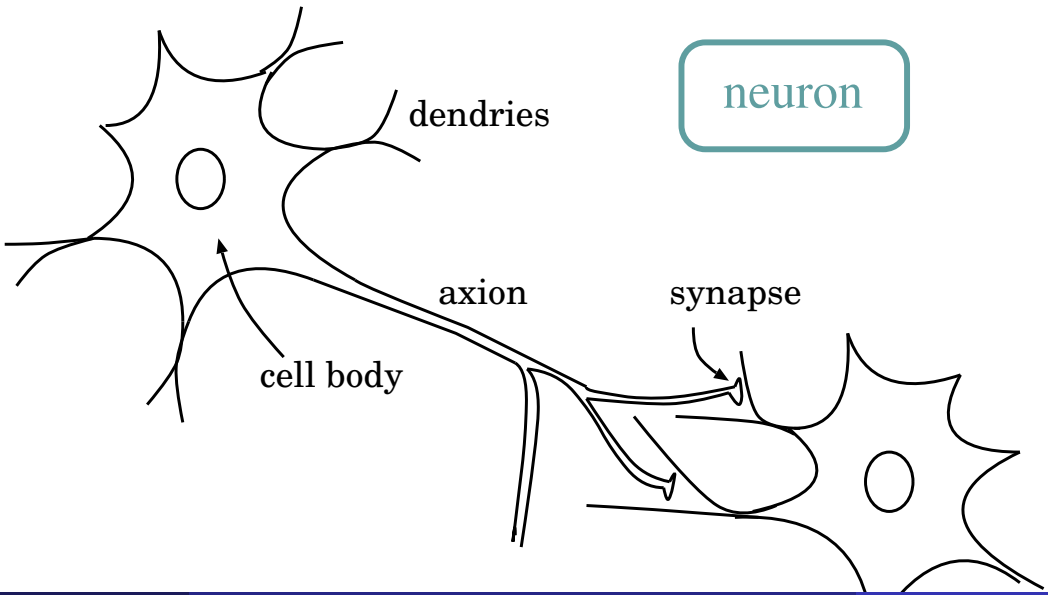
2. ニューラルネットワーク

脳の神経細胞 (neuron)

軸索 (axion): 細胞体から伸びた, 他の神経細胞に信号を伝える出力線維.
樹状突起 (dendries): 細胞体から伸びた, 他の神経細胞からの信号を受け取る入力線維.
シナプス (synapse): 軸索の先端と樹状突起の間にある, 信号伝達などの神経活動に関わる接合部分.

信号電波のメカニズム

各樹状突起に到達した信号がニューロンに入力され, その総和がある閾値を超えることで発火, 軸索を通じて別のニューロンに信号を出力する。

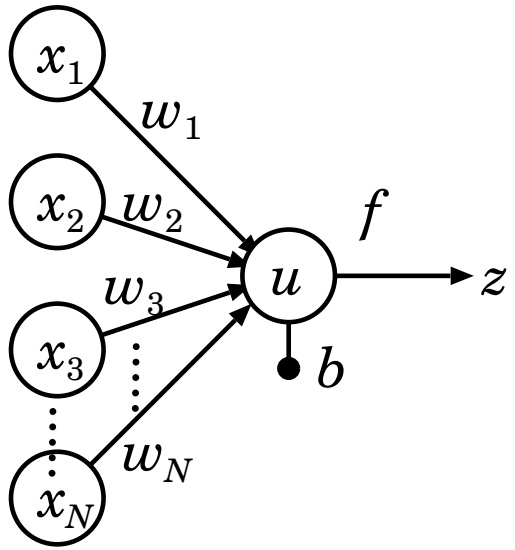


ニューラルネットワーク

ニューラルネットワーク (NN): 脳の神経回路網を数理モデル化したもの.

ニューロンの数理モデル化

パーセプロトン



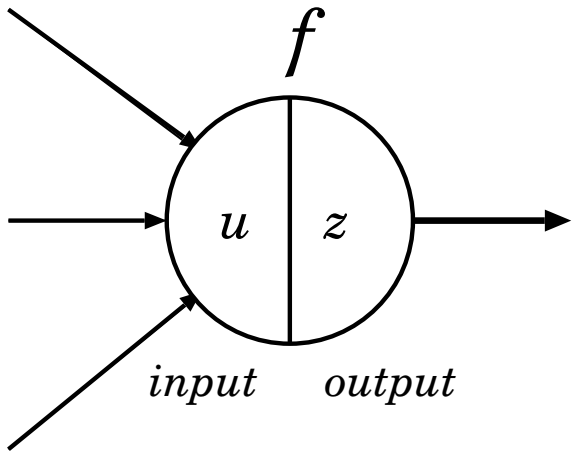
$x_i (i = 1, 2, \cdots, N)$: 入力信号

$w_i (i = 1, 2, \cdots, N)$: 重み (重要度)

$b (< 0)$: バイアス (閾値)

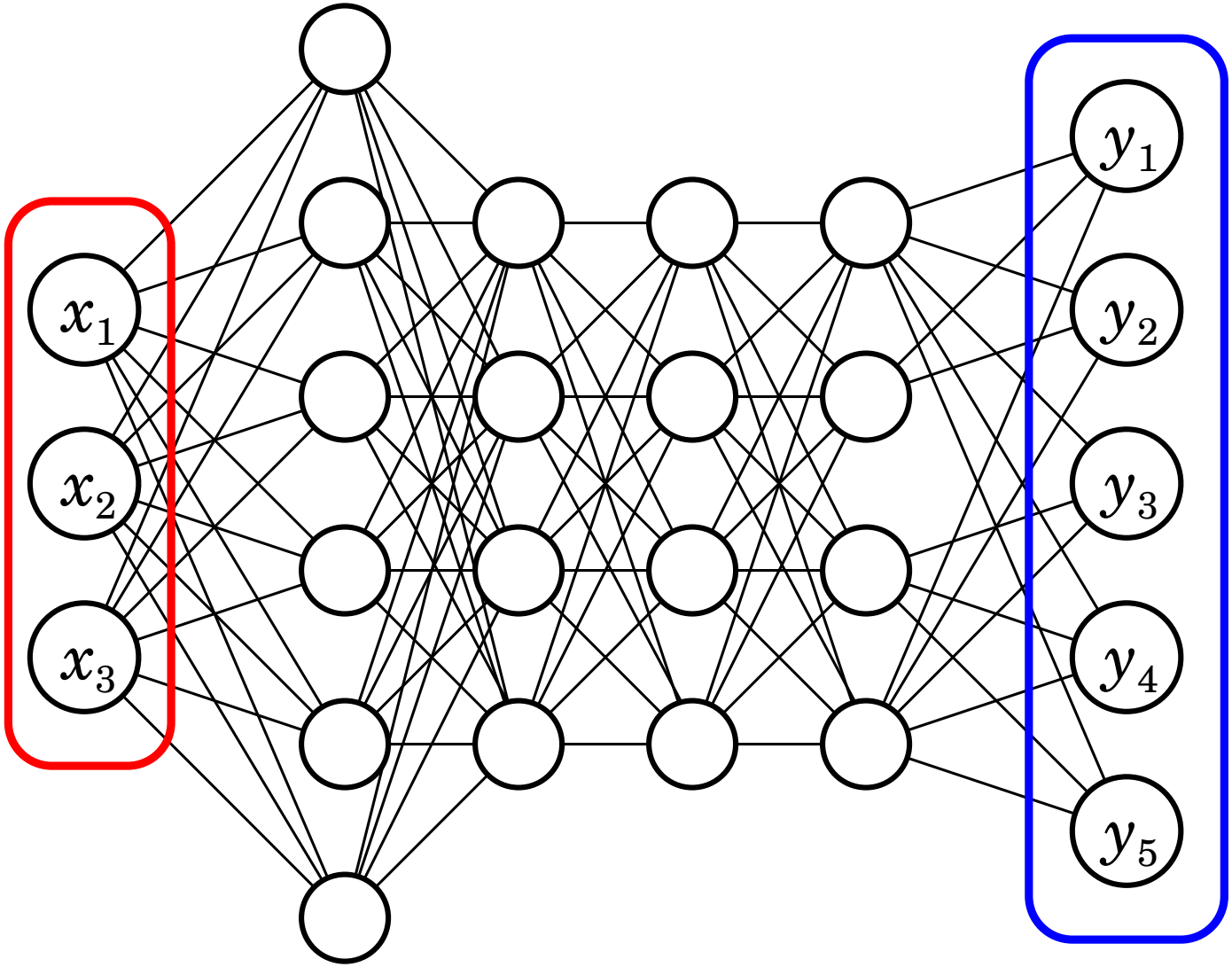
$$u = \sum_{i=1}^{N} w_i x_i + b.$$

$f(\cdot)$: 活性化関数



$$z = f(u) = \begin{cases} 1 & (u \geq 0) \quad \text{発火}, \\ 0 & (u \leq 0) \quad \text{抑制}. \end{cases}$$

⟶ 全ての多層ニューロンへ拡張 (教師あり学習)



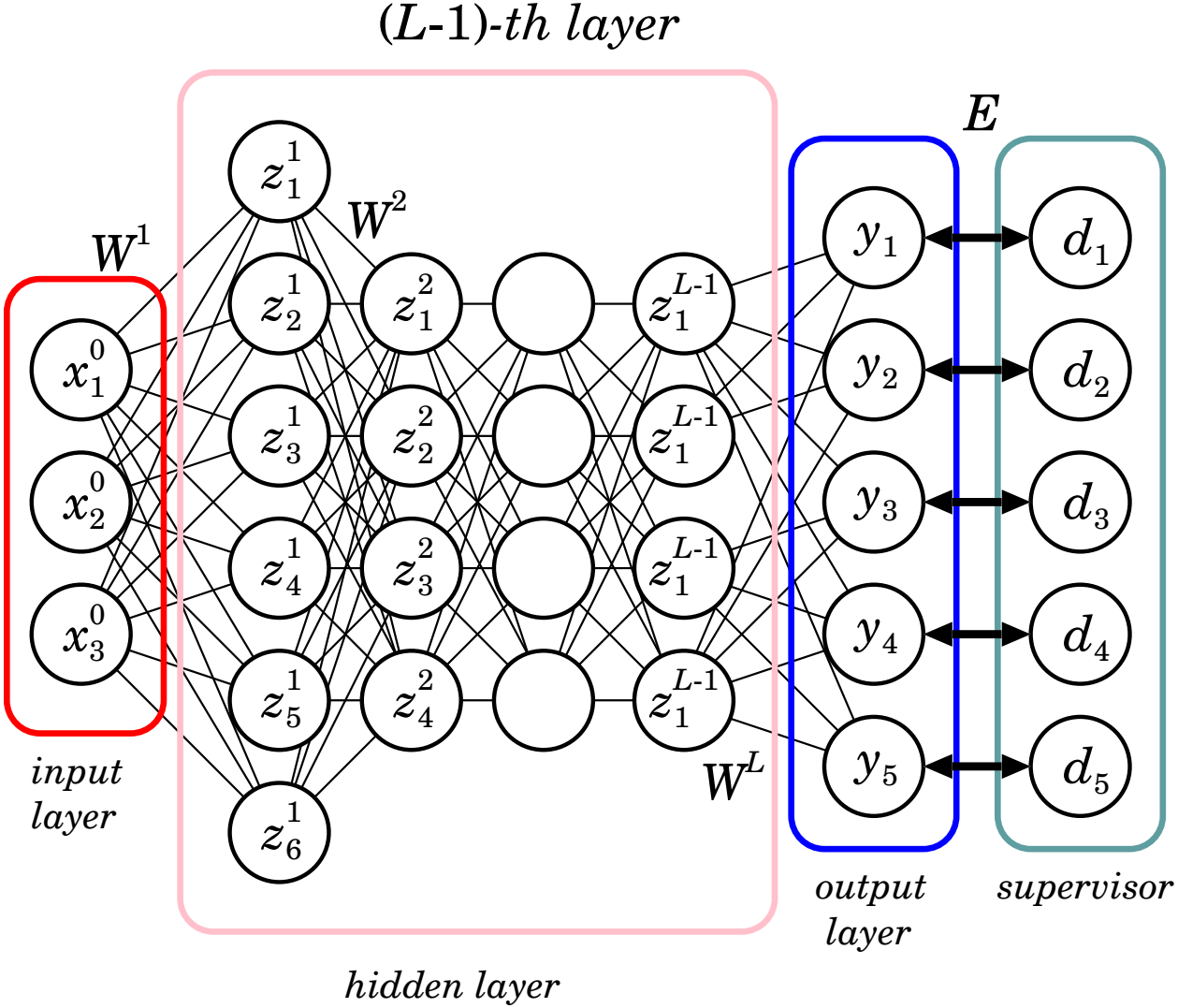
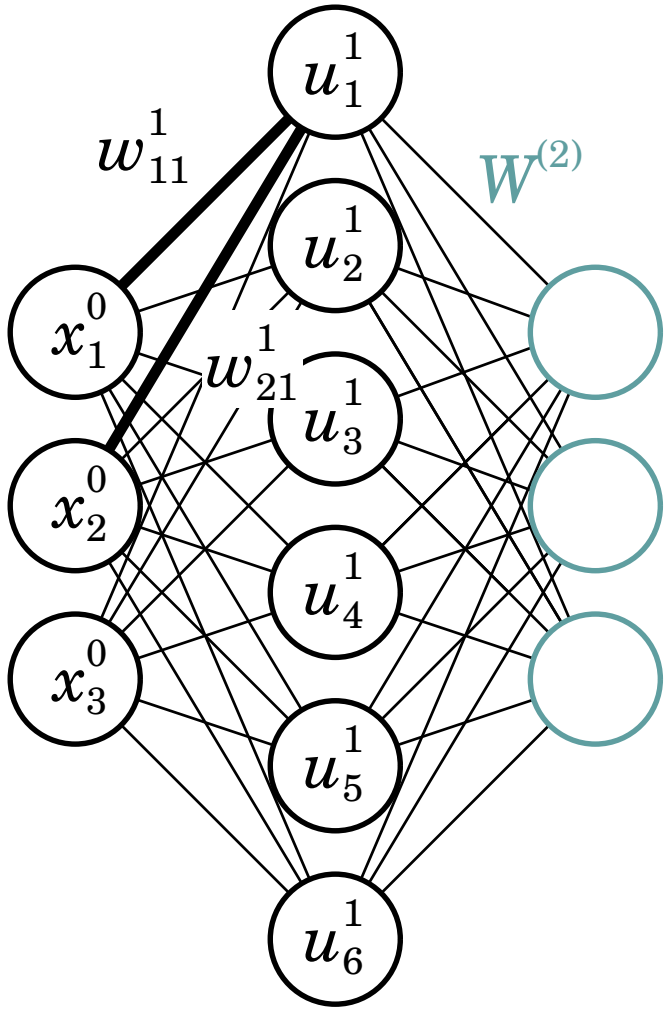


Figure: 教師あり深層学習

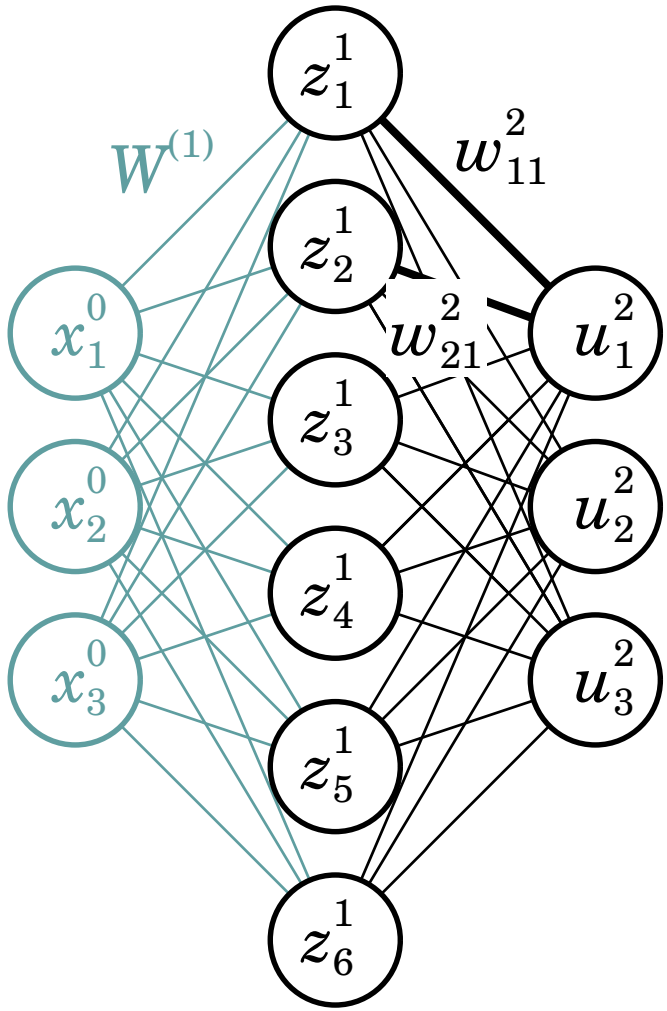
第 1 層

$$
\begin{aligned}
u_1^1 &= w_{11}^1 x_1^0 + w_{12}^1 x_2^0 + \cdots + b_1^1 \\
&= \sum_k w_{k1}^1 x_k^0 + b_1^1, \\
& z_1^1 = f^1(u_1^1). \\
u_2^1 &= w_{12}^1 x_1^0 + w_{22}^1 x_2^0 + \cdots + b_2^1 \\
&= \sum_k w_{k2}^1 x_k^0 + b_2^1, \\
& z_2^1 = f^1(u_2^1). \\
& \cdots \\
u_j^1 &= \sum_k w_{kj}^1 x_k^0 + b_j^1, \\
& z_j^1 = f^1(u_j^1). \\
& \cdots
\end{aligned}
$$



第 2 層

$$
\begin{aligned}
u_1^2 &= w_{11}^2 z_1^1 + w_{21}^2 z_2^1 + \cdots + b_1^2 \\
&= \sum_k w_{k1}^2 x_k^0 + b_1^2, \\
& z_1^2 = f^2(u_1^2). \\
u_2^2 &= w_{12}^2 z_1^1 + w_{22}^2 z_2^1 + \cdots + b_2^2 \\
& z_2^2 = f^2(u_2^2). \\
& \cdots \\
u_j^2 &= \sum_k w_{kj}^1 x_k^0 + b_j^1, \\
& z_j^2 = f^2(u_j^2). \\
& \cdots
\end{aligned}
$$



第 1 層

$$u_j^1 = \sum_k w_{kj}^1 x_k^0 + b_j^1,$$

$$z_j^1 = f^1(u_j^1).$$

第 2 層

$$u_j^2 = \sum_k w_{kj}^2 z_k^1 + b_j^2,$$

$$z_j^2 = f^2(u_j^2).$$

$$\cdots$$

出力層

$$u_j^L = \sum_k w_{kj}^L z_j^{L-1} + b_j^L,$$

$$\underline{y_j = z_j^L = f^L(u_j^L).}$$

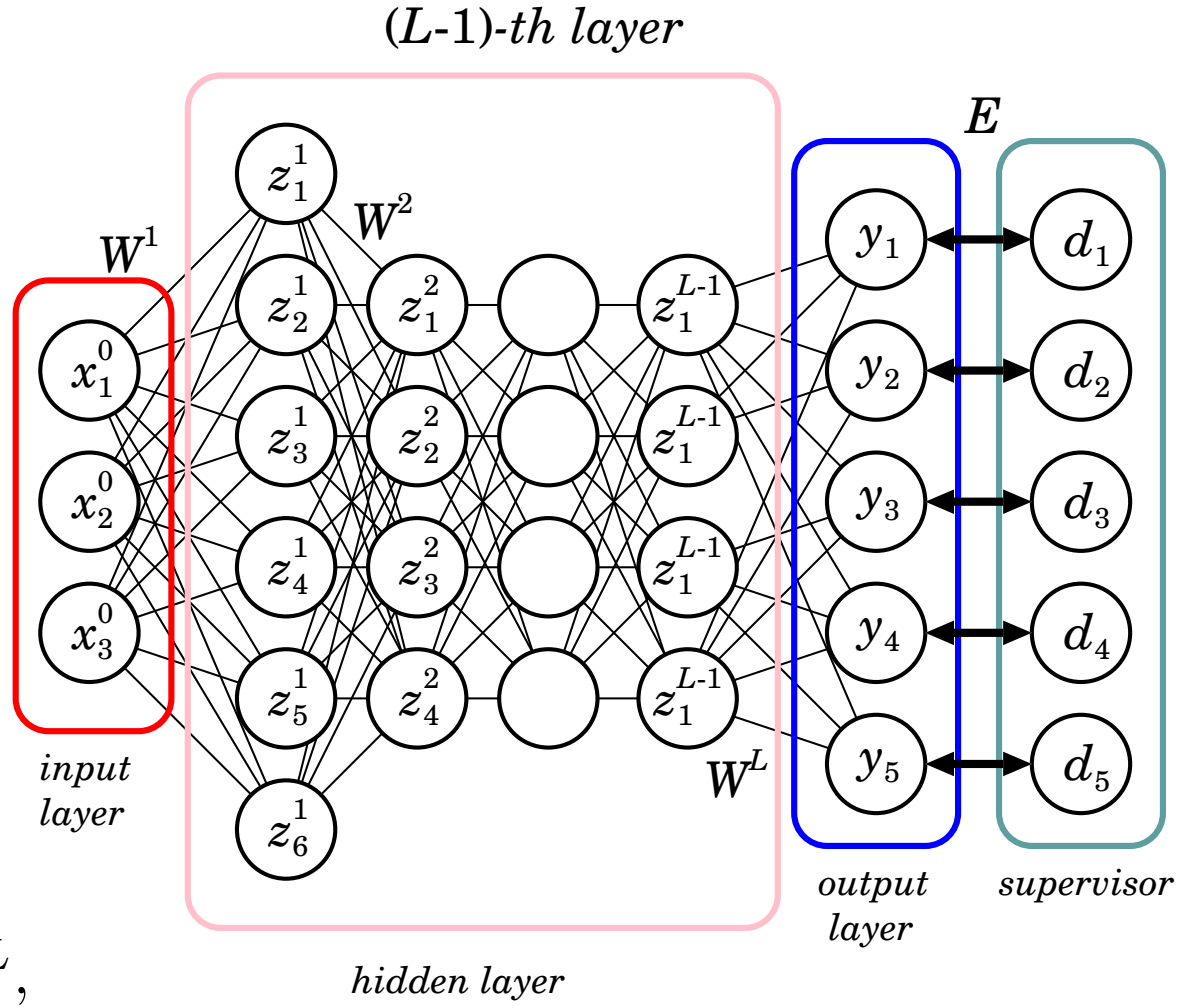
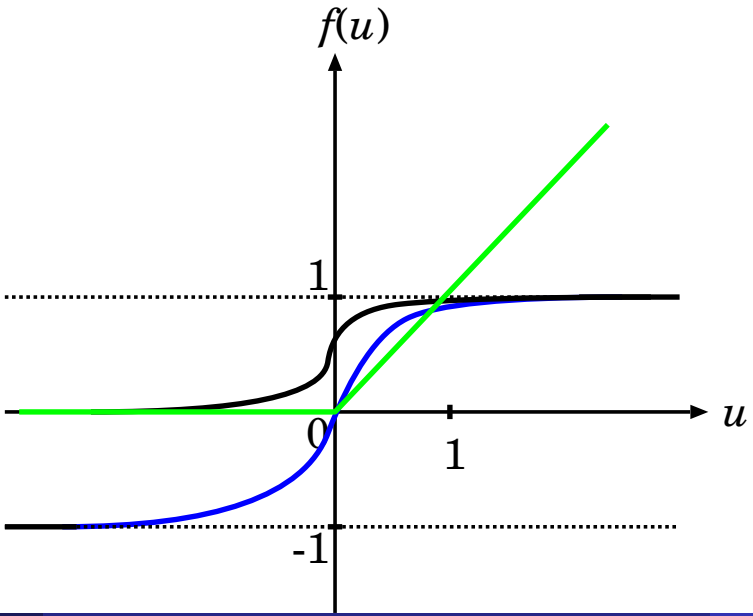


Figure: 教師あり深層学習

活性化関数 $f(u) = \begin{cases} \dfrac{1}{1+e^{-u}}, & \text{シグモイド関数} \\ \tanh u, & \text{ターンエイチ} \\ max\{u, 0\} & \text{Rectified Linear Unit (ReLU)} \\ \dfrac{e_i^u}{\sum_k e^{-u_k}} \ (i = 1, 2, \cdots, n), & \text{ソフトマックス関数} \end{cases}$



NN 学習の仕組み

学習

出力 $\boldsymbol{y}^T = (y_1, y_2, \cdots, y_N)$ が正解 (教師) $\boldsymbol{d}^T = (d_1, d_2, \cdots, d_N)$ と等しくなる，或いは十分近くになるようにパラメーター $\boldsymbol{w} = (W^1, W^2, \cdots, W^L, \boldsymbol{b}^1, \boldsymbol{b}^2, \cdots, \boldsymbol{b}^L)$ を調節すること.

1. 学習データ x からミニバッチを取得する
2. ミニバッチを用いて forward 計算を行い，現在のパラメーター w (初期値は乱数で設定する) による出力 y と損失関数 E を求める

$$E = \frac{1}{2}\sum_{i=1}^{n}(y_i - d_i)^2.$$

3. backward 計算を行い，損失関数 E のパラメーター w に関する勾配を計算する
4. 求めた勾配をもとにパラメーター w を更新する

ミニバッチ学習: すべての学習データ x の中から, ランダムに選び出した集合 (ミニバッチ) を使って, 損失関数 E を計算する
経験的にミニバッチの個数は 256 個程度が選ばれる
1 回毎に実施する学習に用いるデータ数を制限するため，学習データの中から，毎回ランダムに M 個選び出す.
特に $M = 1$ の場合をオンライン学習という.

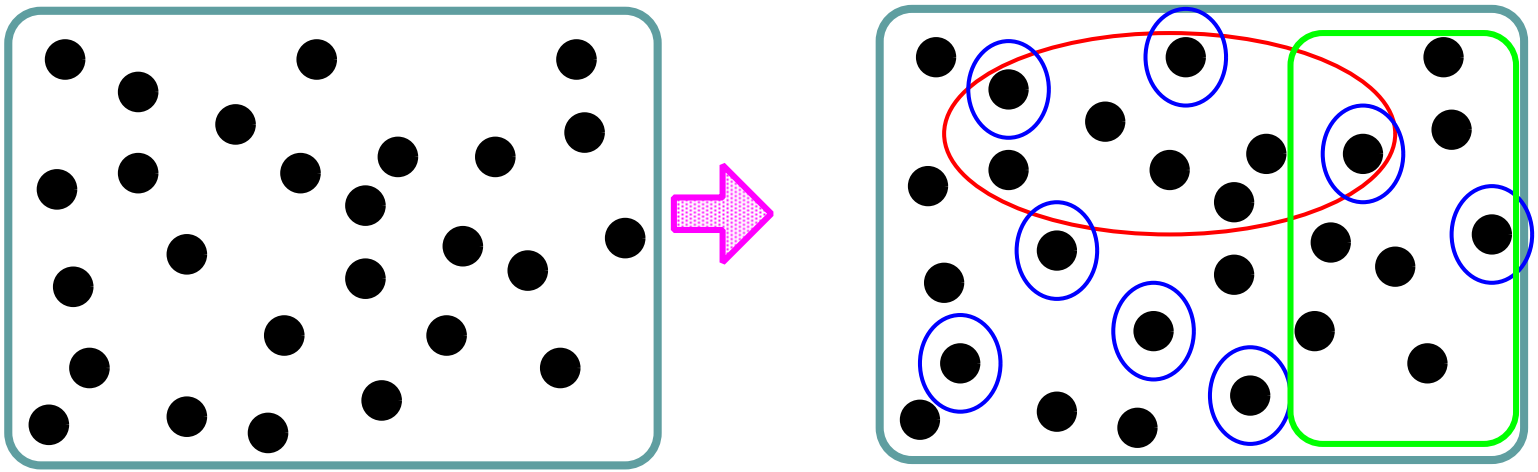
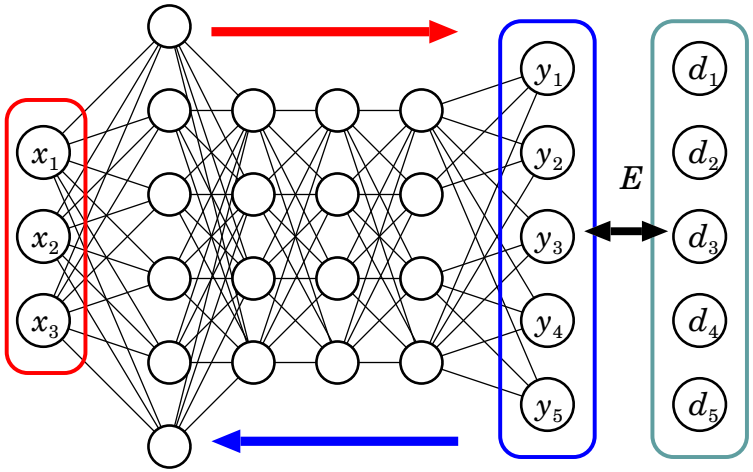


Figure: 用意したデータの中から学習データを選択する.

Figure: forward 計算と backward 計算 (左図)



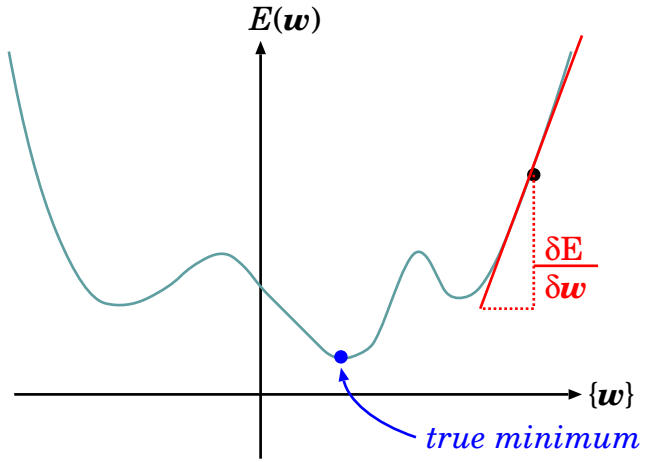
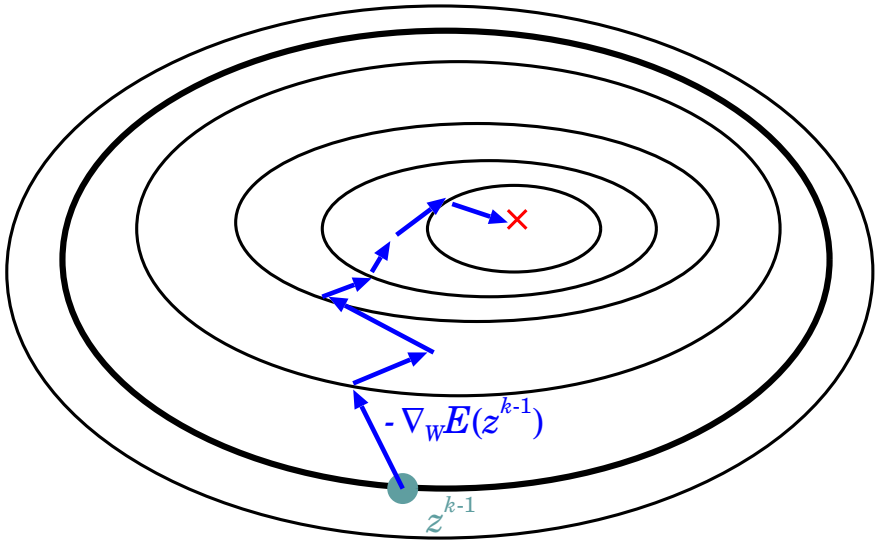


Figure: 勾配降下法 (gradient descent)



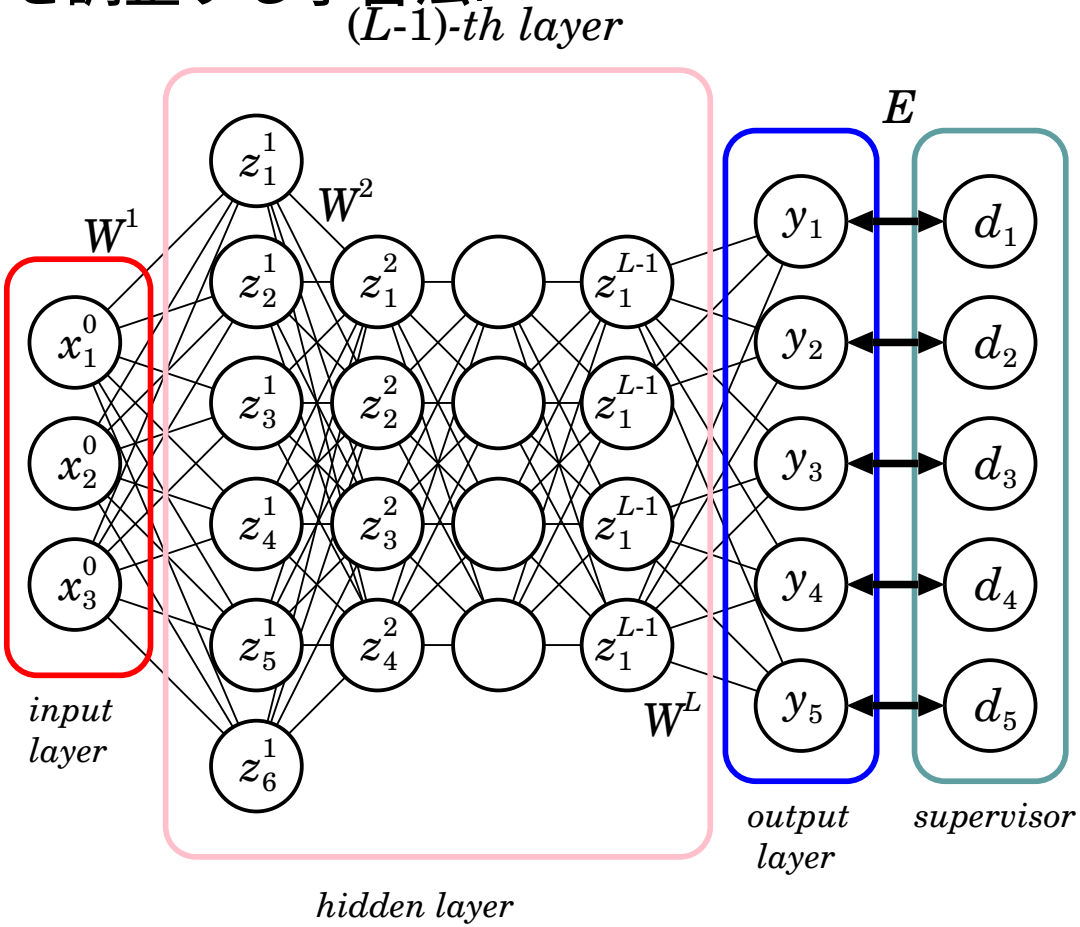
パラメーター w の学習

$$\boldsymbol{w} \longrightarrow \boldsymbol{w}' = \boldsymbol{w} - \eta \nabla_{\boldsymbol{w}} E.$$

η: 学習率 (learning rate)

誤差逆伝播法 (backpropagation)

誤差逆伝播法: ニューラルネットワーク (NN) の学習に利用される学習アルゴリズムの一つ. 学習データ (教師あり) が与えられたとき, NN の出力が教師データと一致するようにパラメーター w を調整する学習法.



学習とは，パラメーター w を調整して損失関数 E を最小にすることであるので，

$$\Delta w_{ij}^{l} = -\eta \frac{\partial E}{\partial w_{ij}^{l}},$$

を計算する必要がある. ここで，

$$u_j^l = \sum_k w_{kj}^l z_k^{l-1} + b_j^l \equiv \sum_{k=1} w_{kj}^l z_k^{l-1} + w_{0j}^l = \sum_{k=0} w_{kj}^l z_k^{l-1}, \qquad z_0^{l-1} \equiv 1,$$

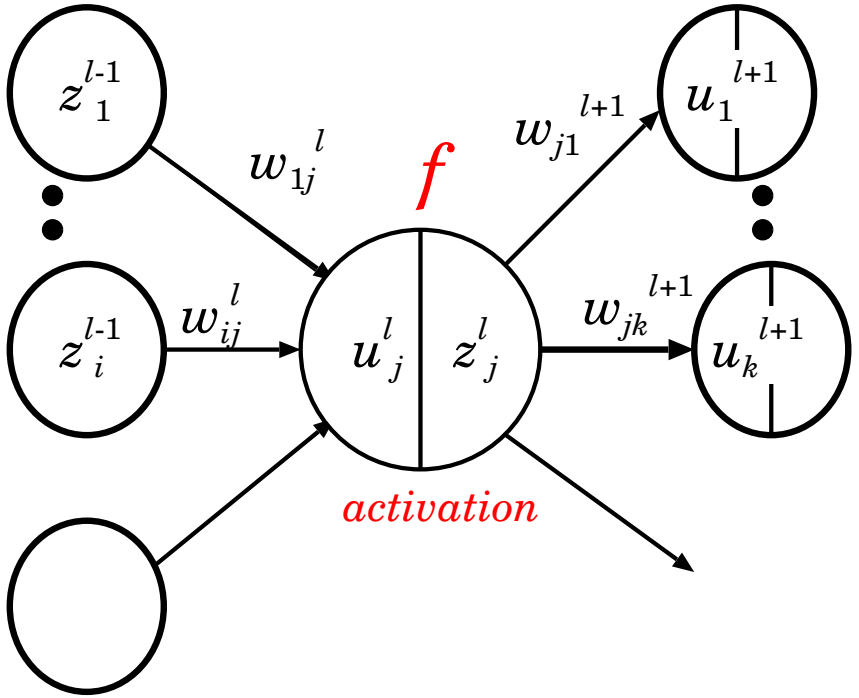
出力層

$$\begin{aligned}
\frac{\partial E}{\partial w_{ij}^L} &= \frac{\partial E}{\partial u_j^L}\frac{\partial u_j^L}{\partial w_{ij}^L} = \delta_j^L \frac{\partial u_j^L}{\partial w_{ij}^L}. \\
\frac{\partial u_j^L}{\partial w_{ij}^L} &= z_i^{L-1}. \\
\delta_j^L &= \frac{\partial E}{\partial z_j^L}\frac{\partial z_j^L}{\partial u_j^L} = \frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial u_j^L} = \underbrace{(y_j - d_j)}_{\text{残差 (誤差)}} \left(f^L(u_j^L) \right)'
\end{aligned}$$

中間層

$$\frac{\partial E}{\partial w_{ij}^l} = \frac{\partial E}{\partial u_j^l}\frac{\partial u_j^l}{\partial w_{ij}^l} = \delta_j^l \frac{\partial u_j^l}{\partial w_{ij}^l} = \delta_j^l z_i^{l-1}.$$

$$\begin{aligned}
\delta_j^l &= \frac{\partial E}{\partial u_j^l} = \sum_k \frac{\partial E}{\partial u_k^{l+1}}\frac{\partial u_k^{l+1}}{\partial u_j^l} \\
&= \sum_k \frac{\partial E}{\partial u_k^{l+1}}\frac{\partial u_k^{l+1}}{\partial z_j^l}\frac{\partial z_j^l}{\partial u_j^l} \\
&= \sum_k \delta_k^{l+1} w_{jk}^{l+1} \left(f^l(u_j^l)\right)'
\end{aligned}$$



$$\frac{\partial E}{\partial w_{ij}^l} = \sum_k \delta_k^{l+1} w_{jk}^{l+1} \left(f^l(u_j^l)\right)' z_i^{l-1}.$$

既に δ_j^L が求められていることにより，$\delta_j^L \to \delta_j^{L-1} \to \cdots \to \delta_j^l \to \cdots$ と出力層から遡ることによって δ_j^l は全て求められる.

3. 物理学への応用

符号問題

符号問題: モンテカルロシミュレーションにおける実変数の多重積分

$$\mathcal{Z} = \int_{\mathbb{R}^n} d^n x \; e^{-S(x)}, \quad S(x) = S_{\mathrm{R}}(x) + iS_{\mathrm{I}}(x),$$

の Boltzmann 因子 $e^{-S(x)}$ が確率解釈できない.

$\longrightarrow$ **回避法**

$$\begin{aligned}
\langle \mathcal{O}(x) \rangle &= \frac{1}{\mathcal{Z}} \int_{\mathbb{R}^n} d^n x \; \mathcal{O}(x) e^{-S(x)} = \frac{\int_{\mathbb{R}^n} d^n x \; \mathcal{O}(x) e^{-iS_{\mathrm{I}}(x)} e^{-S_{\mathrm{R}}}}{\int_{\mathbb{R}^n} d^n x \; e^{-iS_{\mathrm{I}}(x)} e^{-S_{\mathrm{R}}(x)}} \\
&= \frac{\int_{\mathbb{R}^n} d^n x \; \mathcal{O}(x) e^{-iS_{\mathrm{I}}(x)} e^{-S_{\mathrm{R}}} / \int_{\mathbb{R}^n} d^n x \; e^{-S_{\mathrm{R}}(x)}}{\int_{\mathbb{R}^n} d^n x \; e^{-iS_{\mathrm{I}}(x)} e^{-S_{\mathrm{R}}(x)} / \int_{\mathbb{R}^n} d^n x \; e^{-S_{\mathrm{R}}(x)}} \\
&= \frac{\langle e^{-iS_{\mathrm{I}}(x)} \mathcal{O}(x) \rangle_{\mathrm{R}}}{\langle e^{-iS_{\mathrm{I}}(x)} \rangle_{\mathrm{R}}}. \qquad \langle \mathcal{O}(x) \rangle_{\mathrm{R}} \equiv \frac{\int_{\mathbb{R}^n} d^n x \; \mathcal{O}(x) e^{-S_{\mathrm{R}}(x)}}{\int_{\mathbb{R}^n} d^n x \; e^{-S_{\mathrm{R}}(x)}}
\end{aligned}$$

$$\langle e^{-iS_{\mathrm{I}}(x)} \rangle_{\mathrm{R}} \quad = \quad \frac{\mathcal{Z}}{Z_{\mathrm{R}}} = e^{-\beta V(f(x)-f_{\mathrm{R}}(x))},$$

により，統計的に有意な結果を得るためには，

$$\text{statistical error} \sim \frac{1}{\sqrt{\#\text{ sample}}} < \text{signal}, \qquad \#\text{ sample} \simeq e^{2\beta V(f(x)-f_{\mathrm{R}}(x))}$$

が成り立つ必要がある.

経路最適化法

経路最適化法: 符号問題が弱くなるように元の積分経路を複素空間まで拡張，変形する.

1. 損失関数を定義する

$$E = \frac{1}{2}\int d^n t |e^{i\theta(t)} - e^{i\theta_0}|^2 \times |J(t)e^{-S(z)}| = |\mathcal{Z}| \left(\frac{1}{|\langle e^{i\theta(t)} \rangle_{\mathrm{R}}} - 1 \right)$$

$$\mathcal{Z}(z) = \int d^n t \; J(t) e^{-S(z(t))} = |\mathcal{Z}| e^{i\theta_0}, \quad J(t) = \det\left(\frac{\partial z_i}{\partial t_j}\right),$$

2. 変形前の積分経路で損失関数 E を計算する
3. ニューラルネットワークを利用し，損失関数が小さくなるように積分経路を変形する

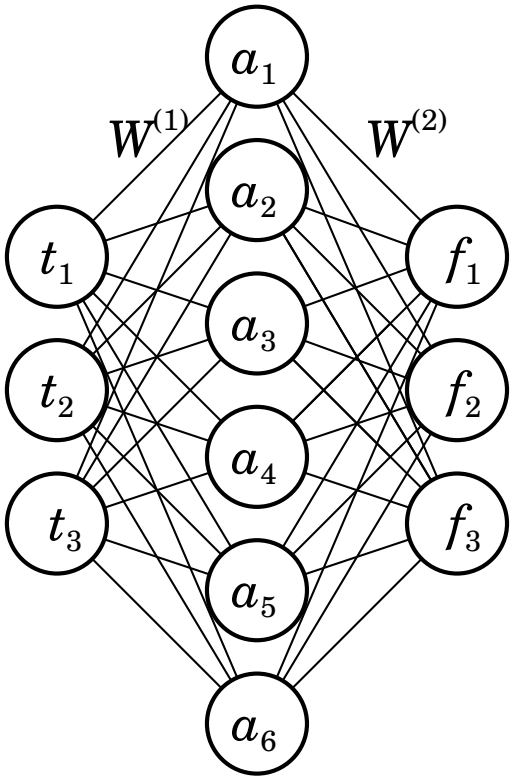
$$z_k(t) = t_k + i(\alpha_k f_k(t) + \beta_k) \Leftarrow \begin{cases} a_i = g(W_{ij}^{(1)} t_j + b_i^{(1)}), \\ f_k = g(W_{ki}^{(2)} a_i + b_k^{(2)}), \end{cases}$$

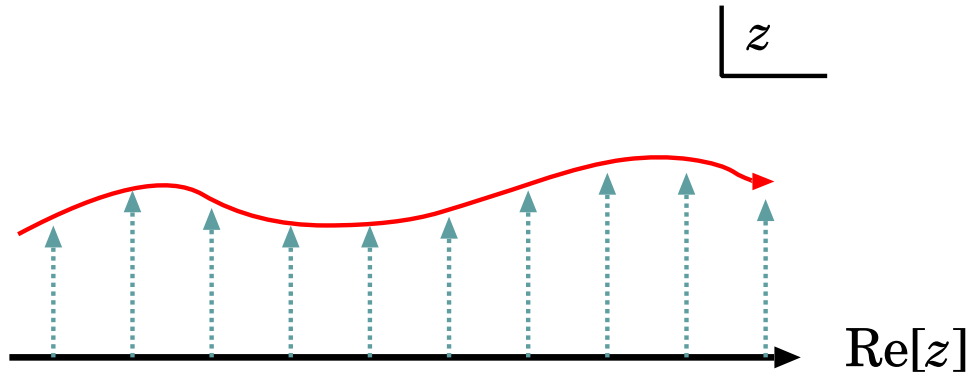
4. 誤差逆伝播法によって損失関数が十分小さくなるまで変形を繰り返す (教師なし学習)

- 2 次元有限密度複素 $\lambda\phi^4$ 理論を採用 [3]

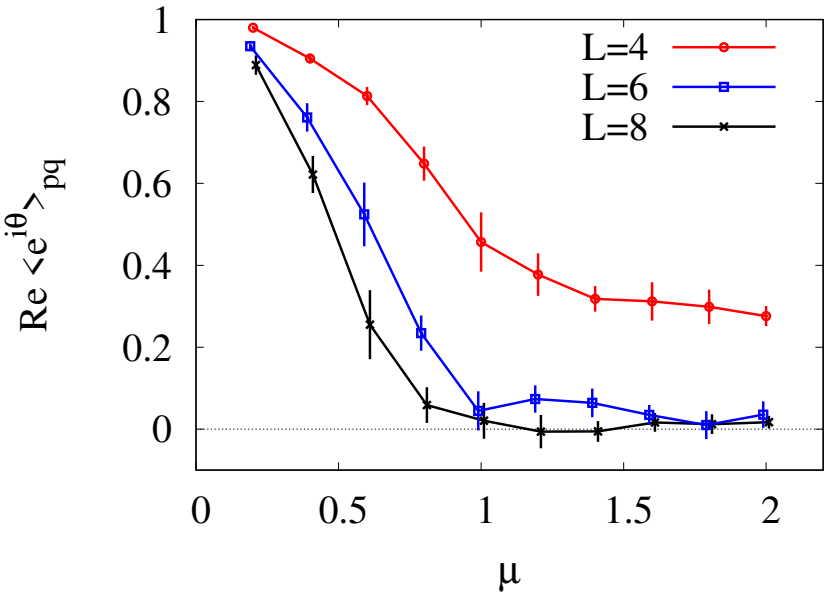
$$S = \sum_x \left[\frac{4+m^2}{2} \phi_{a,x}\phi_{a,x} + \frac{\lambda}{4}\left(\phi_{a,x}\phi_{a,x}\right)^2 - \phi_{a,x}\phi_{a,x+\hat{1}} \right.$$
$$\left. -\phi_{a,x}\phi_{b,x+\hat{0}}(\delta_{a,b}\cosh(\mu) - i\epsilon_{a,b}\sinh(\mu)) \right]$$

- 活性化関数 $g(u)$ は $g(u) = \tanh(u)$ を採用





[3] Y. Mori, K. Kashiwa and A. Ohnishi, PTEP 2018(2), 023B04



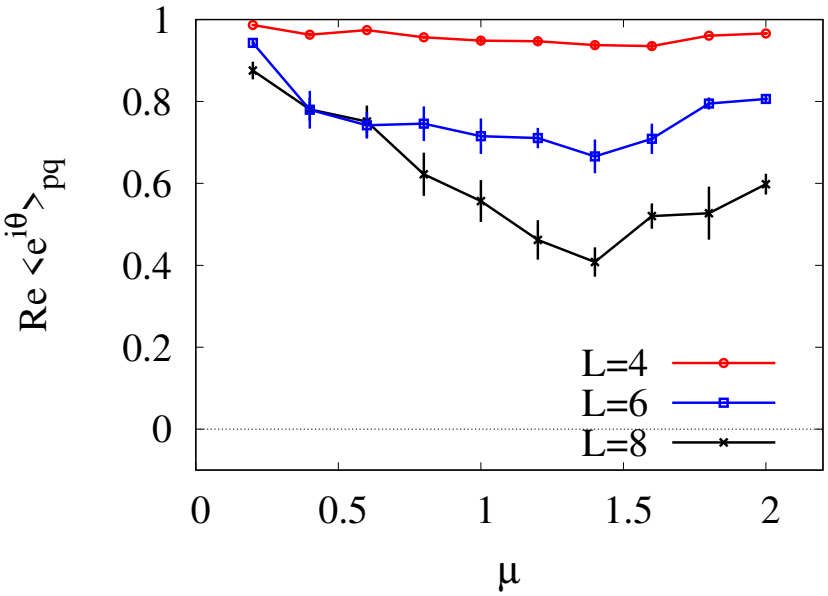
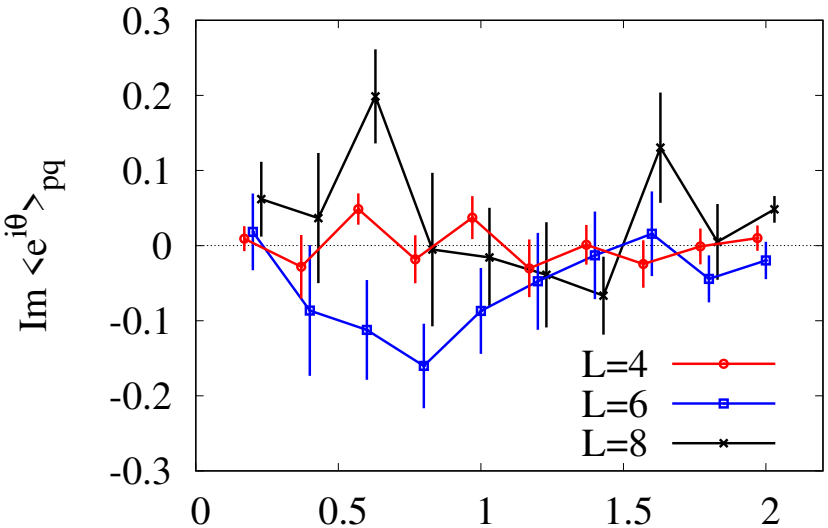


Figure: The real part of the average phase factor without(left) and with the optimization(right) as a function of μ.

Figure: The imaginary part of the average phase factor with the optimization as a function of μ.



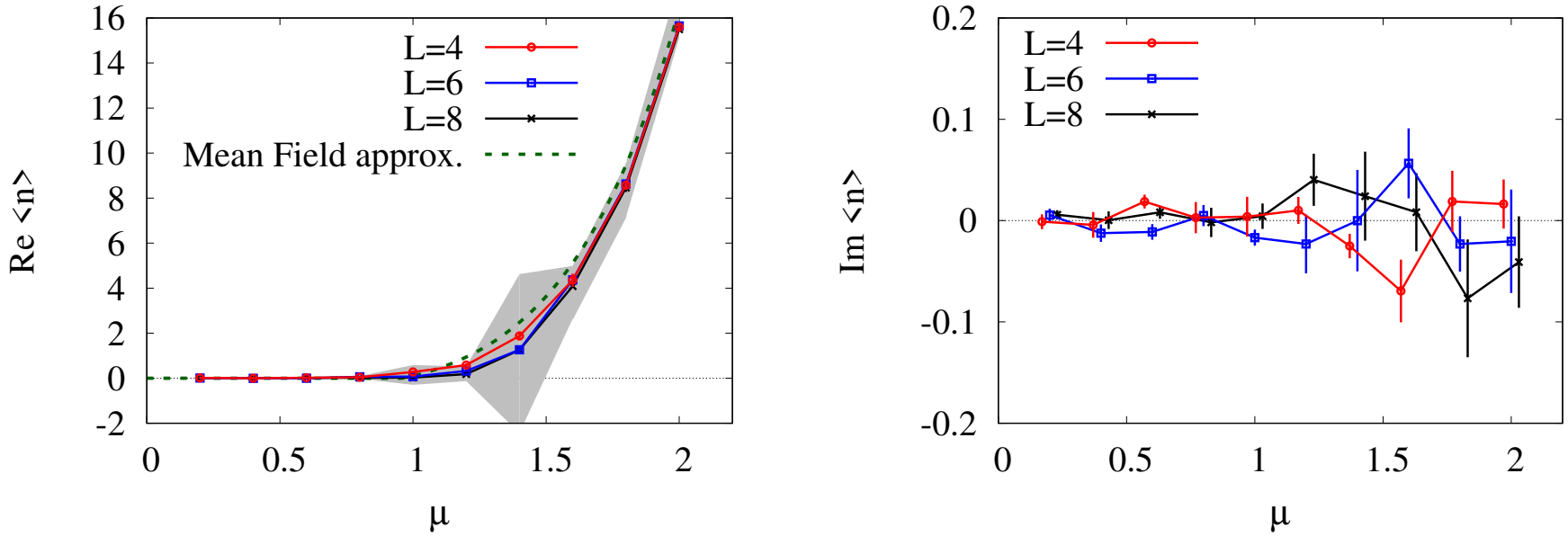


Figure: The expectation value of the number density n in the Monte Carlo calculations and in the mean field approximation(dashed line).

$$n = \frac{1}{V} \sum_x \left[\delta_{a,b} \sinh(\mu) - i\epsilon_{ab} \cosh(\mu) \right] z_{a,x} z_{b,x+\hat{0}}, \quad V = L_t L_x.$$

話せなかったこと

- **万能近似定理** (universal approximation theorem)
- **畳み込みニューラルネットワーク** (convolution neural network)+ **プーリング** (pooling)
- **教師なし学習**

  etc.$\cdots$