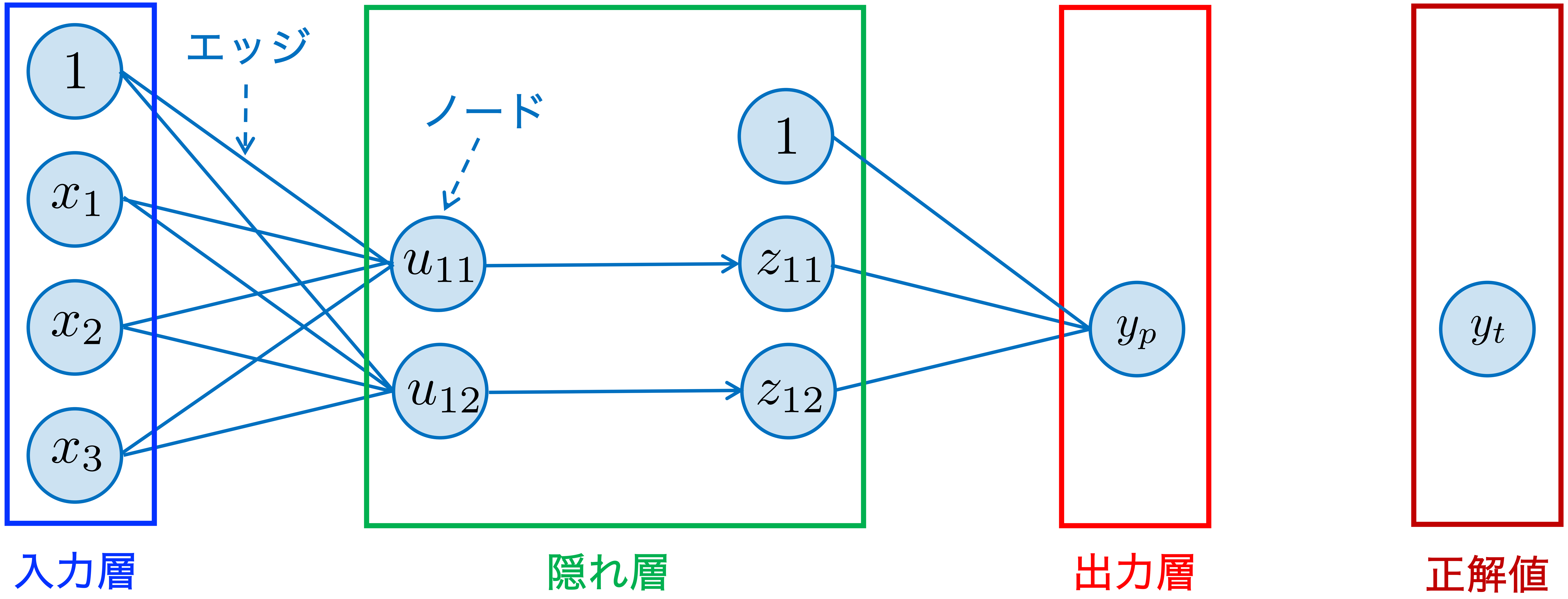
機械学習ゼミ
1. 機械学習の基礎 (ニューラルネットワーク)

別所秀将

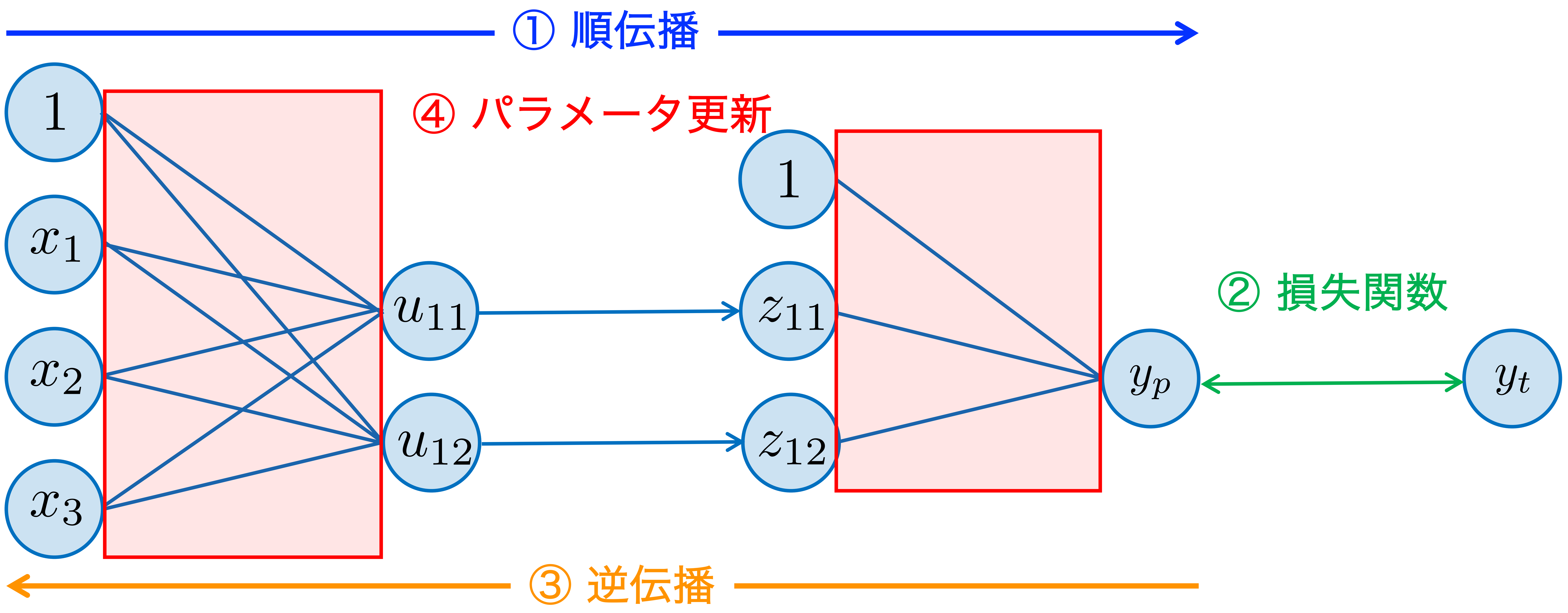
1. ニューラルネットワークとは

■ニューラルネットワークの構造

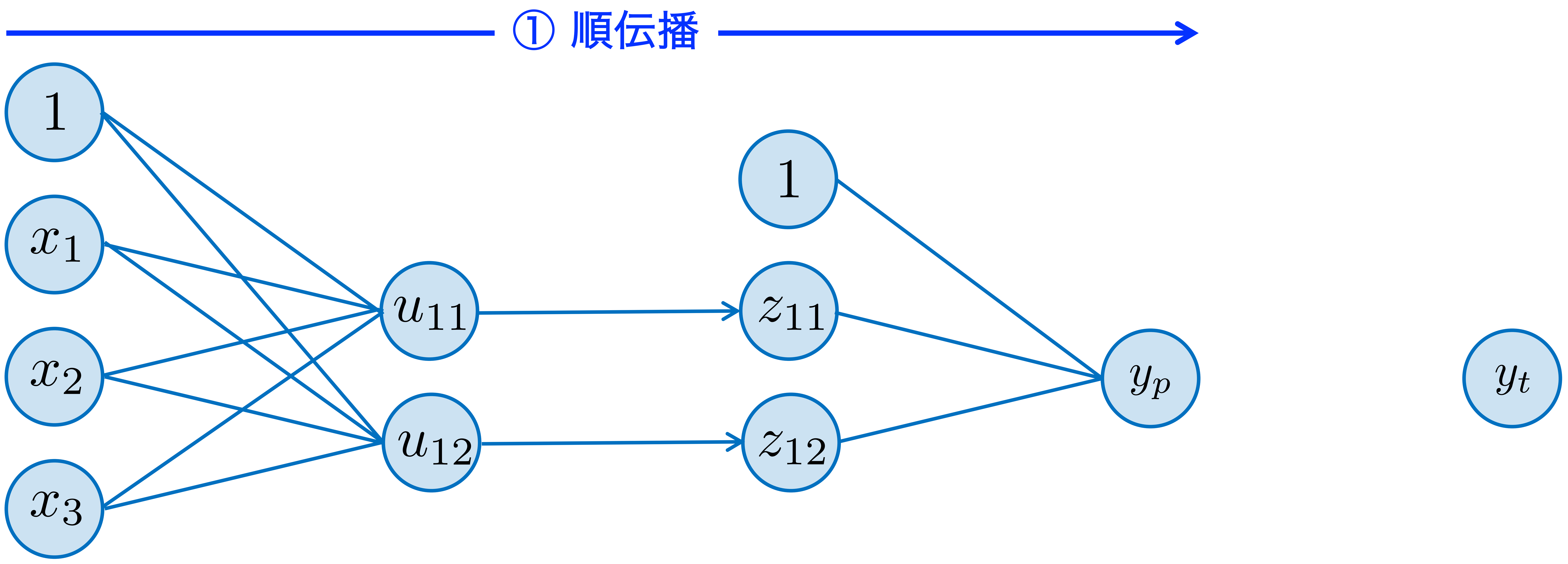


1. ニューラルネットワークとは

■ニューラルネットワークの構造

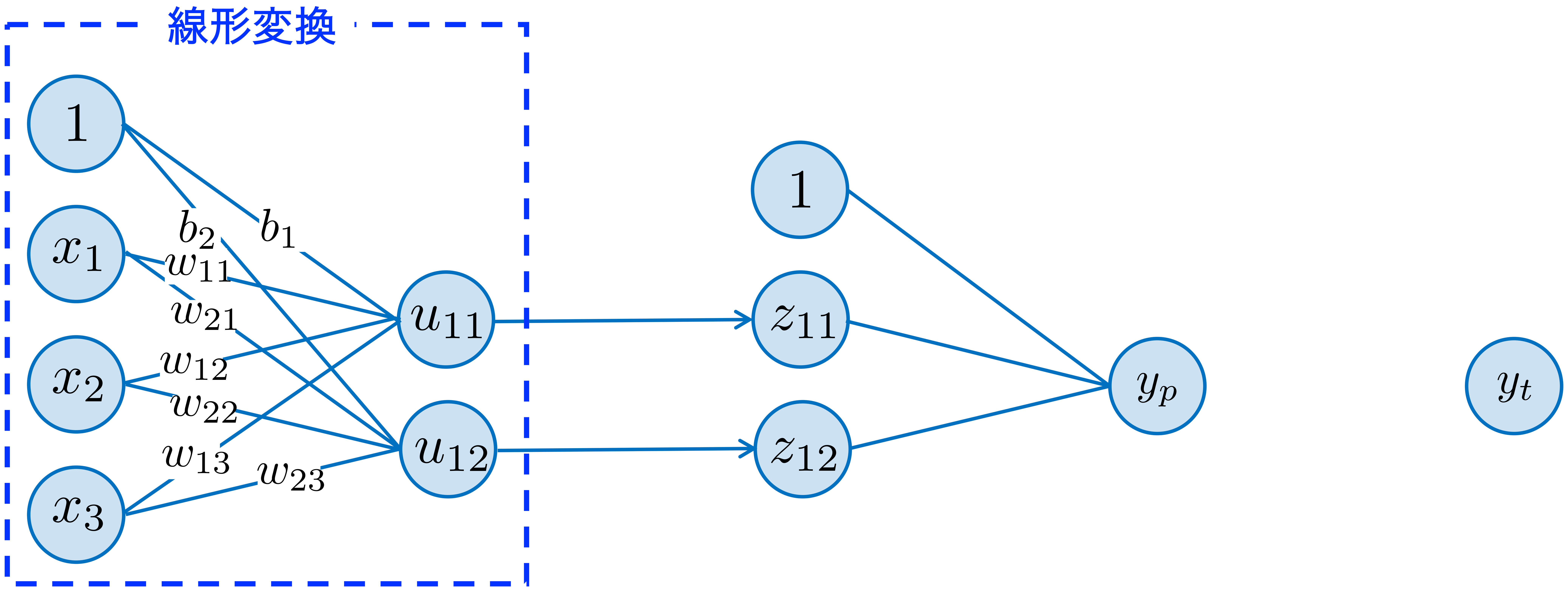


2. 順伝播



2. 順伝播

■線形変換

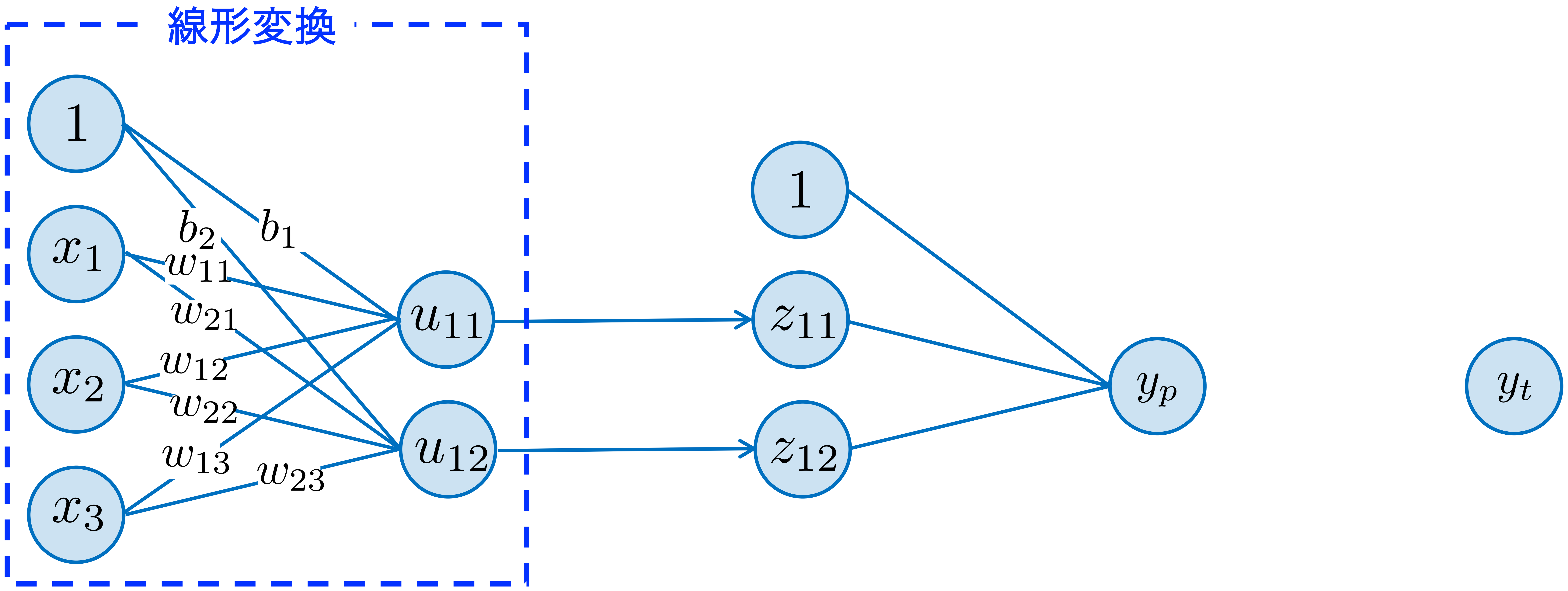


$$u_{11} = w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + b_1$$

$$u_{12} = w_{21}x_1 + w_{22}x_2 + w_{23}x_3 + b_2$$

2. 順伝播

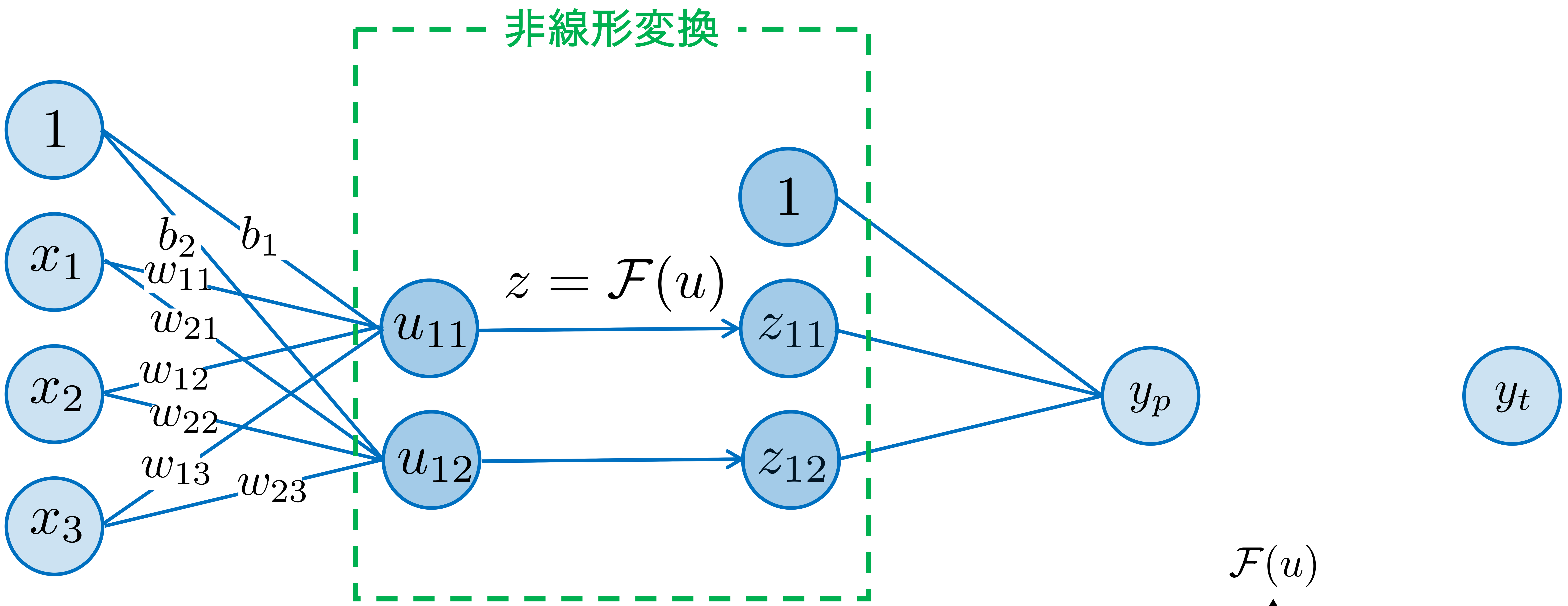
■線形変換



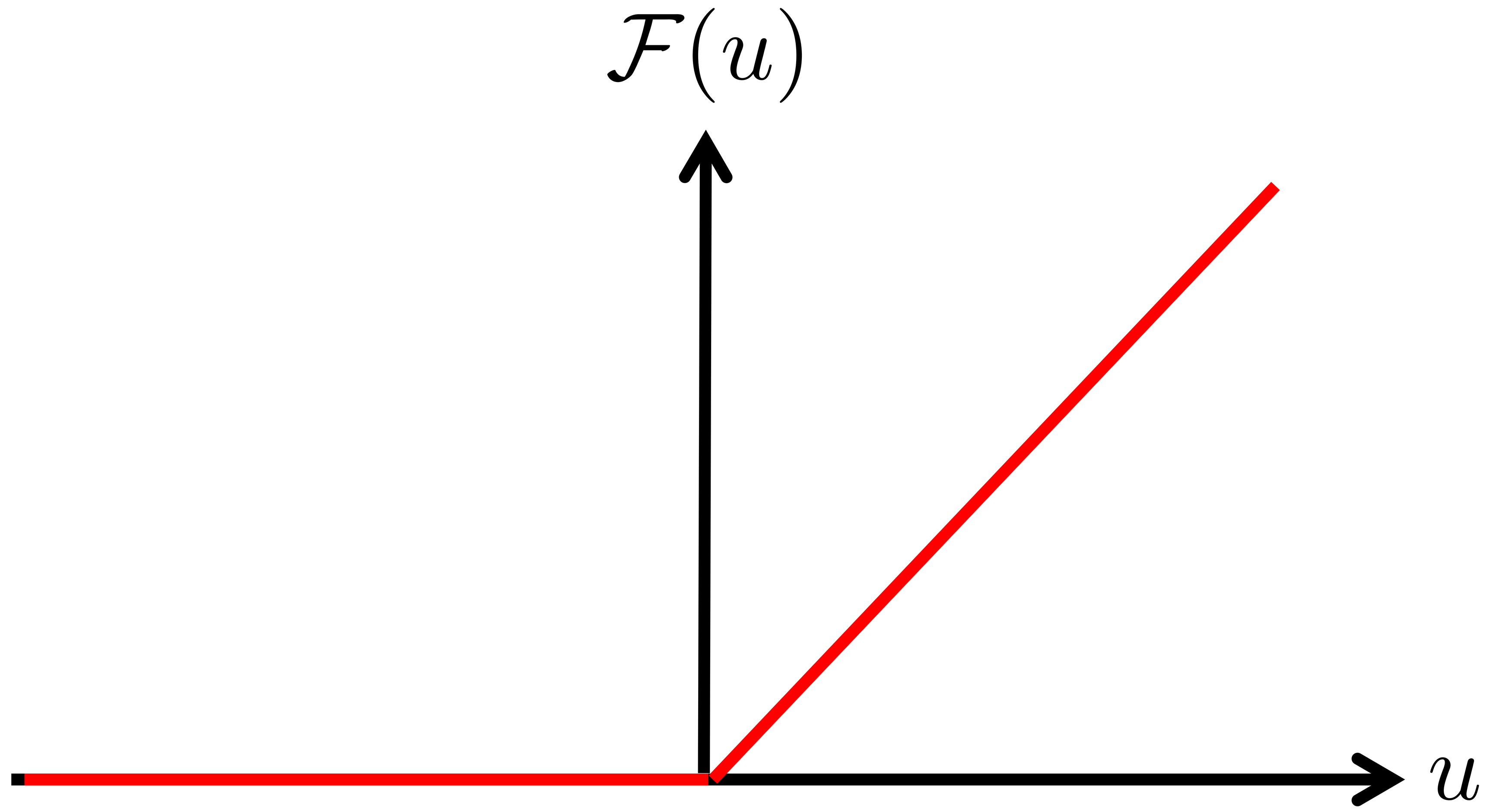
$$\begin{bmatrix} u_{11} \\ u_{12} \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \quad \text{or} \quad \mathbf{u} = \mathbf{wx} + \mathbf{b}$$

2. 順伝播

■非線形変換



● 非線形変換の例：ReLU関数 $\mathcal{F}(u) = \max(0, u)$

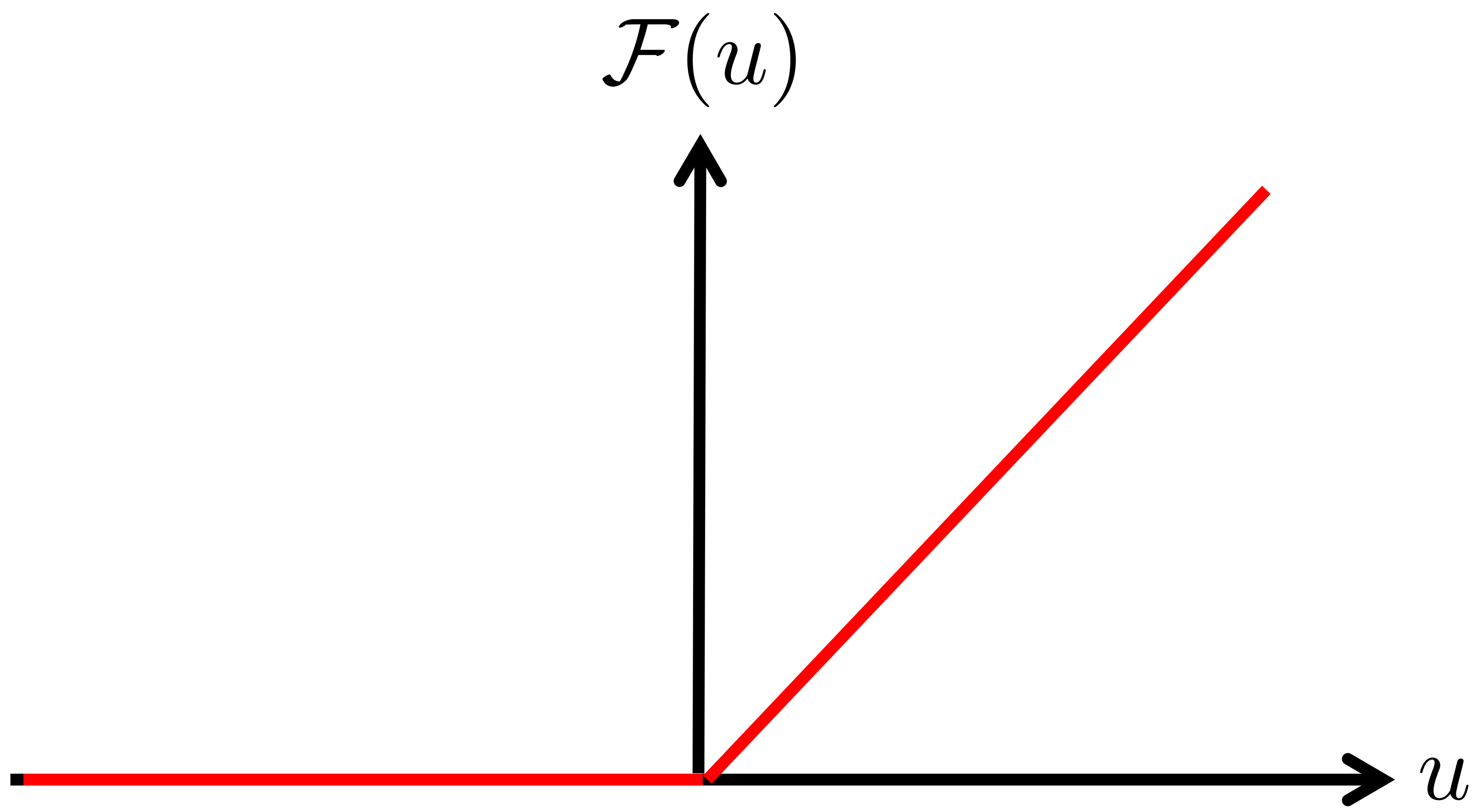


2. 順伝播

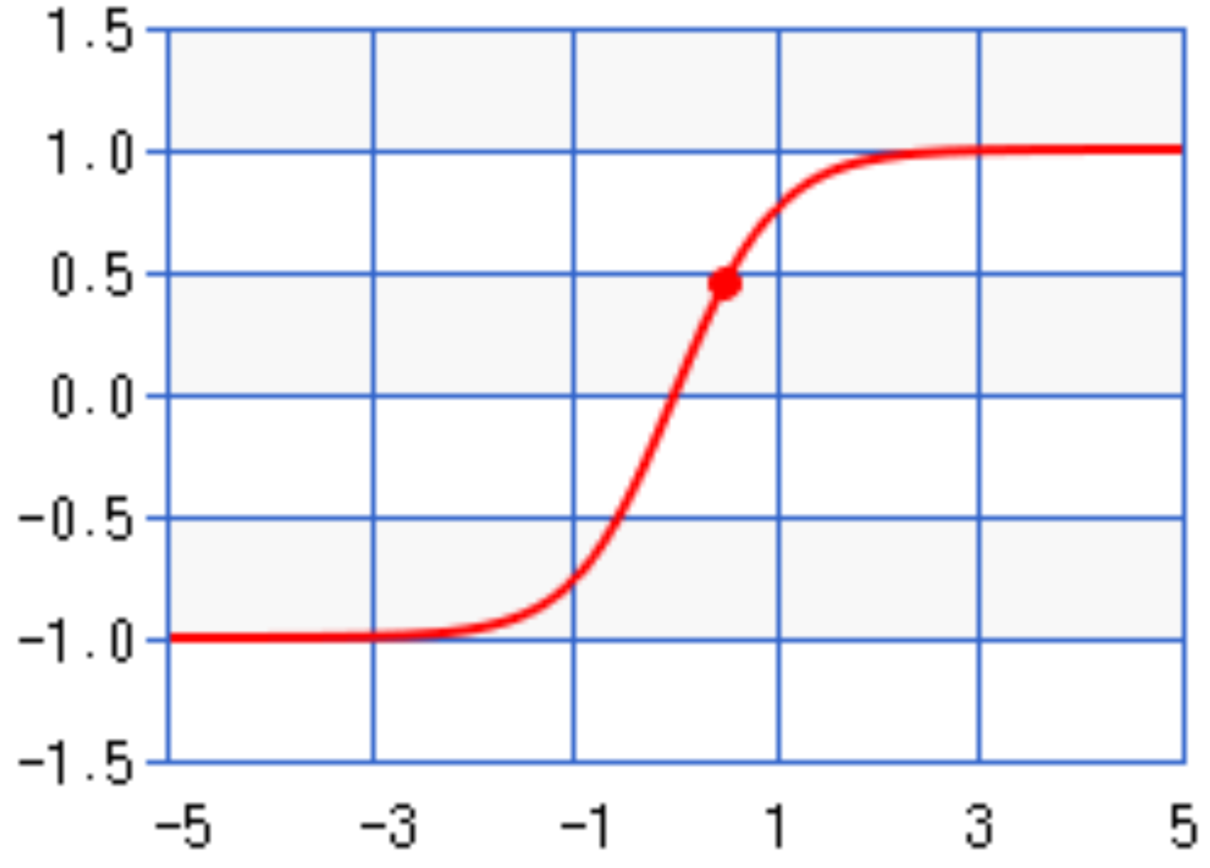
■非線形変換

● 非線形変換の例

➢ ReLU関数 $\mathcal{F}(u) = \max(0, u)$

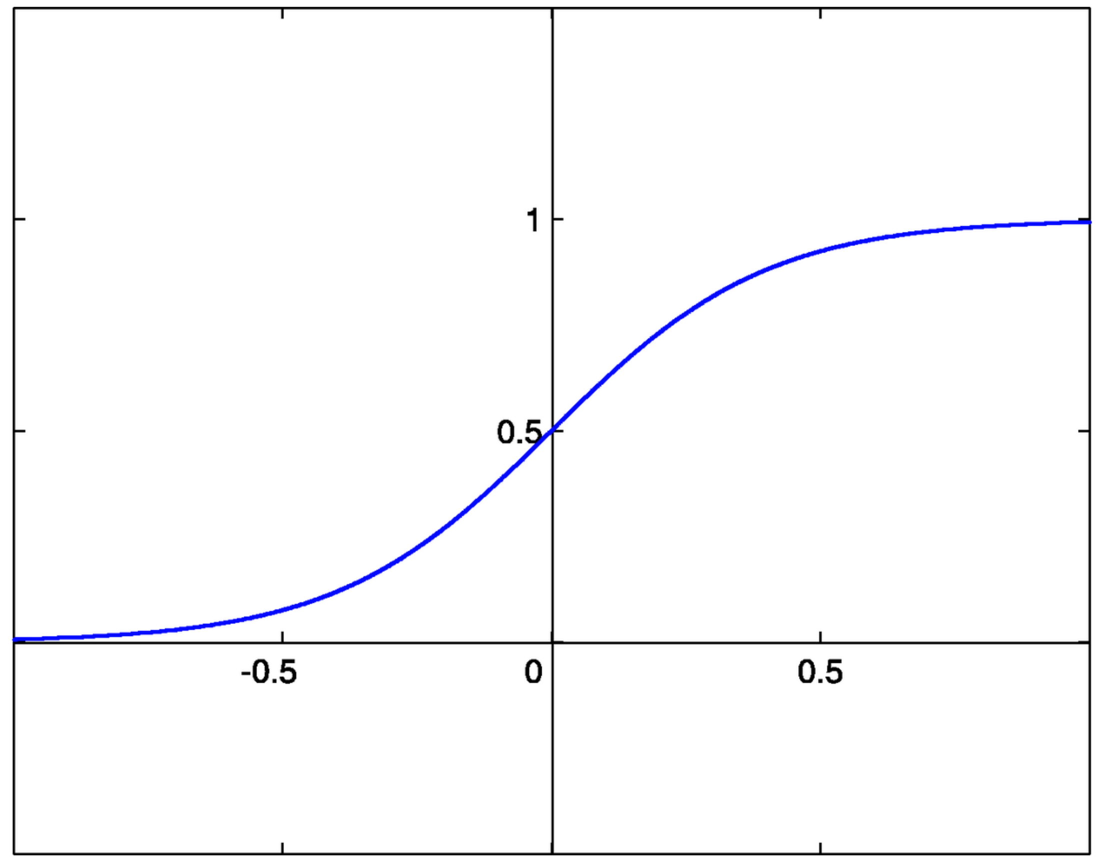


➢ tanh関数 $\mathcal{F}(u) = (e^u - e^{-u})/(e^u + e^{-u})$



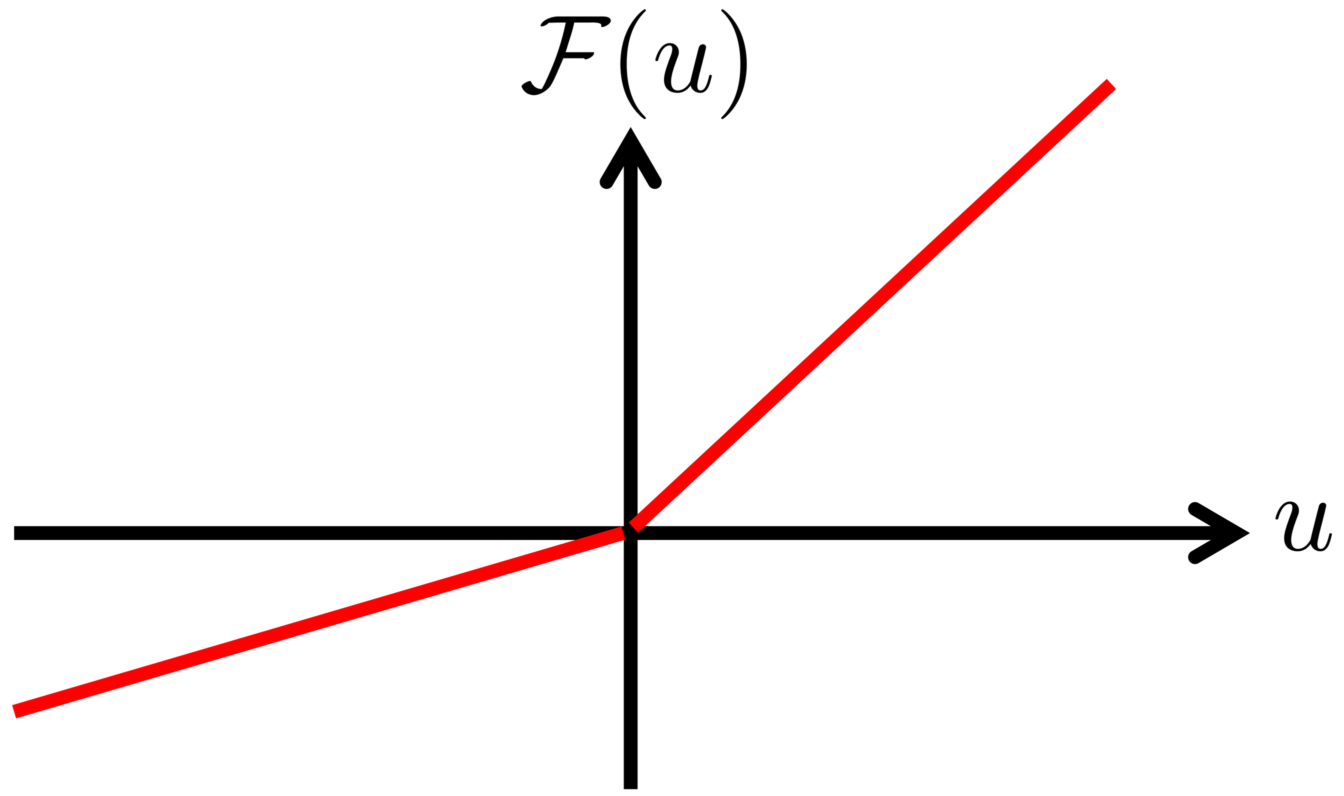
[https://keisan.casio.jp/exec/system/1541125775]

➢ シグモイド関数 $\mathcal{F}(u) = 1/(1 + e^{-u})$



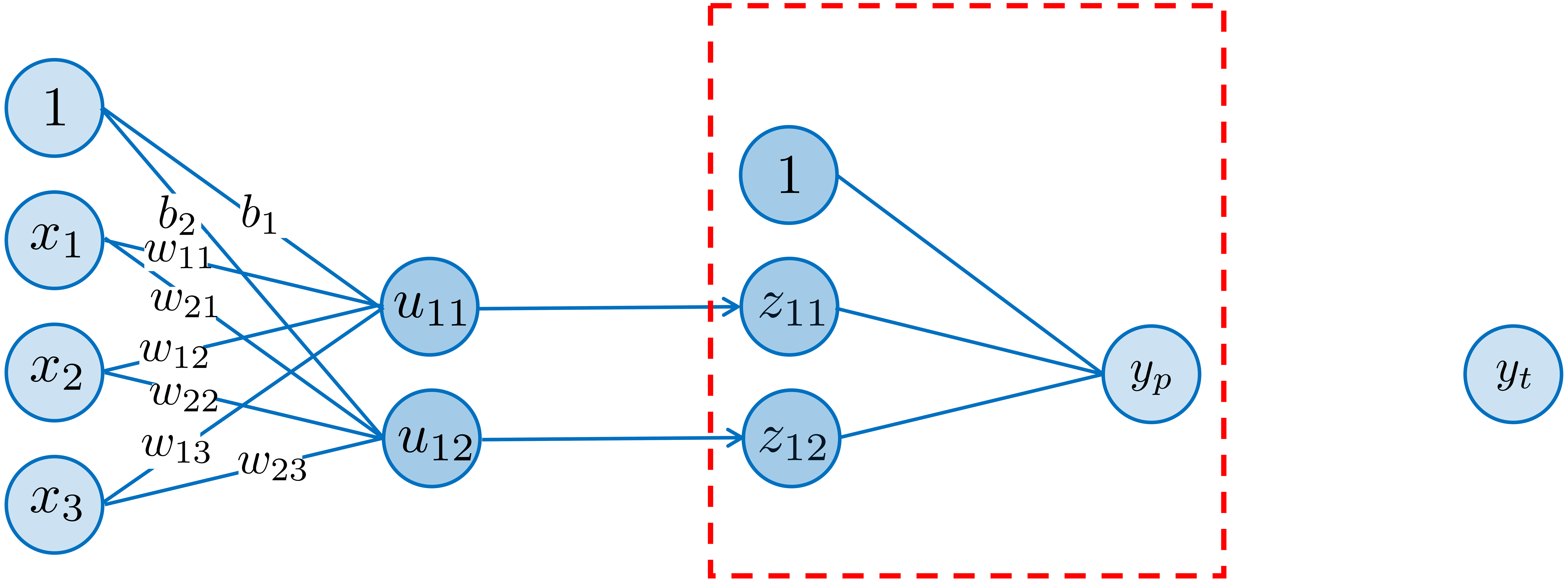
[ウィキペディア]

➢ Leaky ReLU関数



2. 順伝播

■出力層

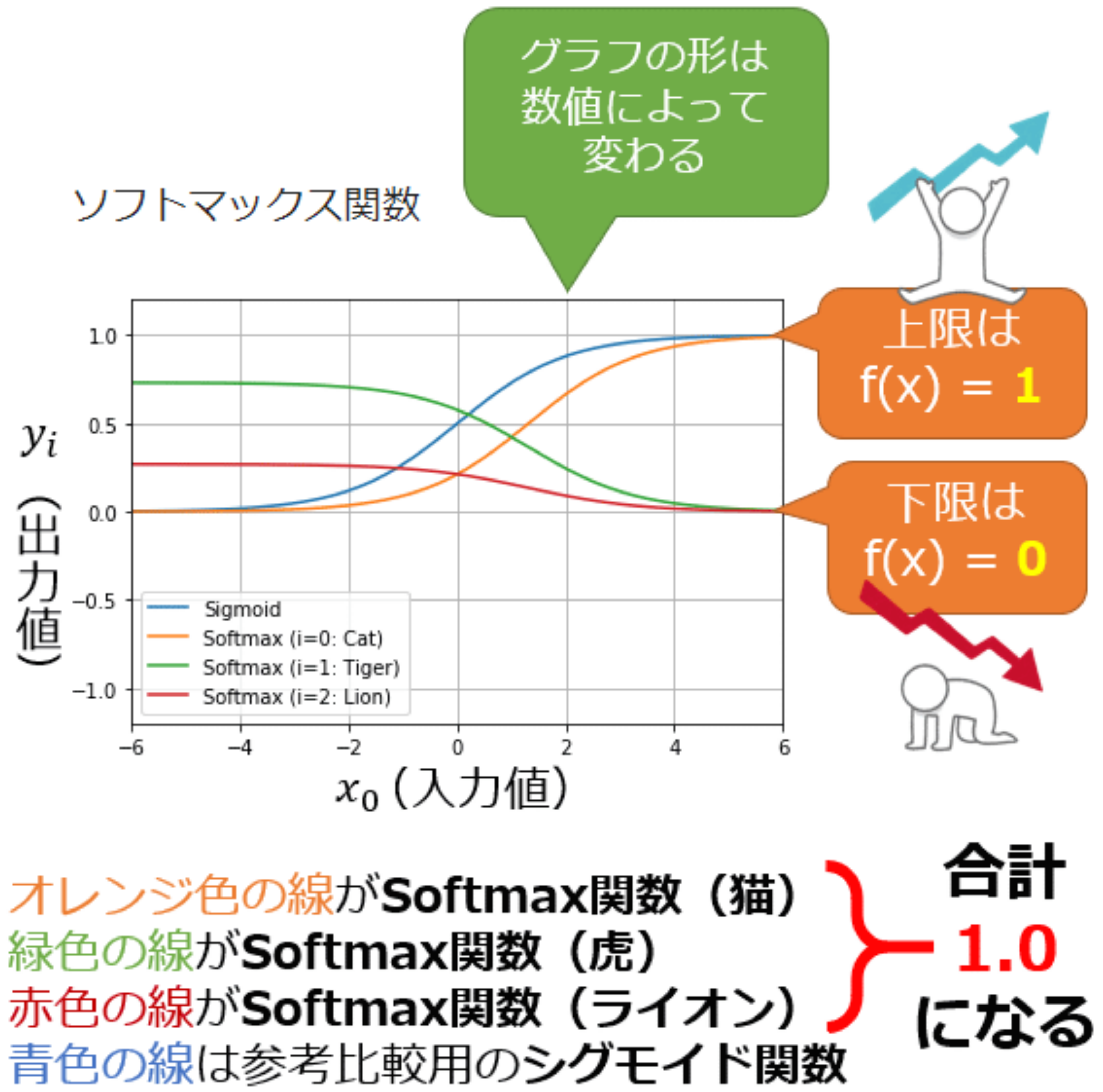


● 線形変換 + 非線形変換(出力層用)

2. 順伝播

■出力層

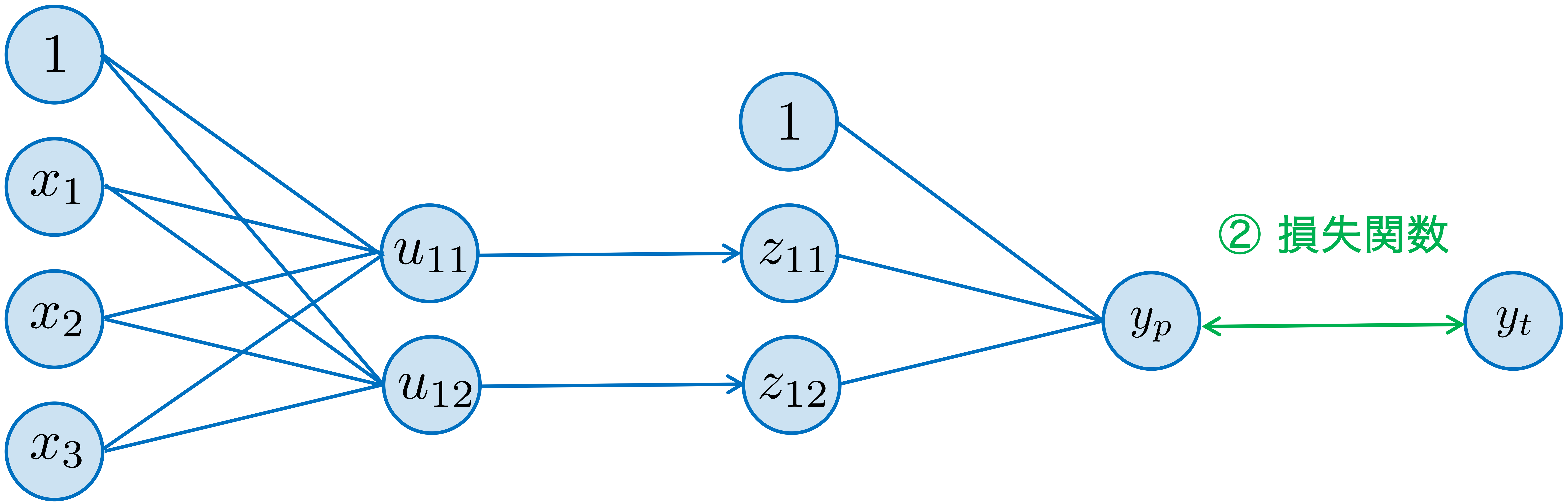
- 非線形変換(出力層用)
  - 回帰問題：恒等関数 $\mathcal{F}(u) = u$
  - 分類問題：ソフトマックス関数 $\mathcal{F}(u_j) = \dfrac{e^{u_j}}{\sum_j e^{u_j}}$



[https://atmarkit.itmedia.co.jp/ait/articles/2004/08/news016.html]

3. 損失関数

■損失関数：予測値と正解値の誤差を表す関数



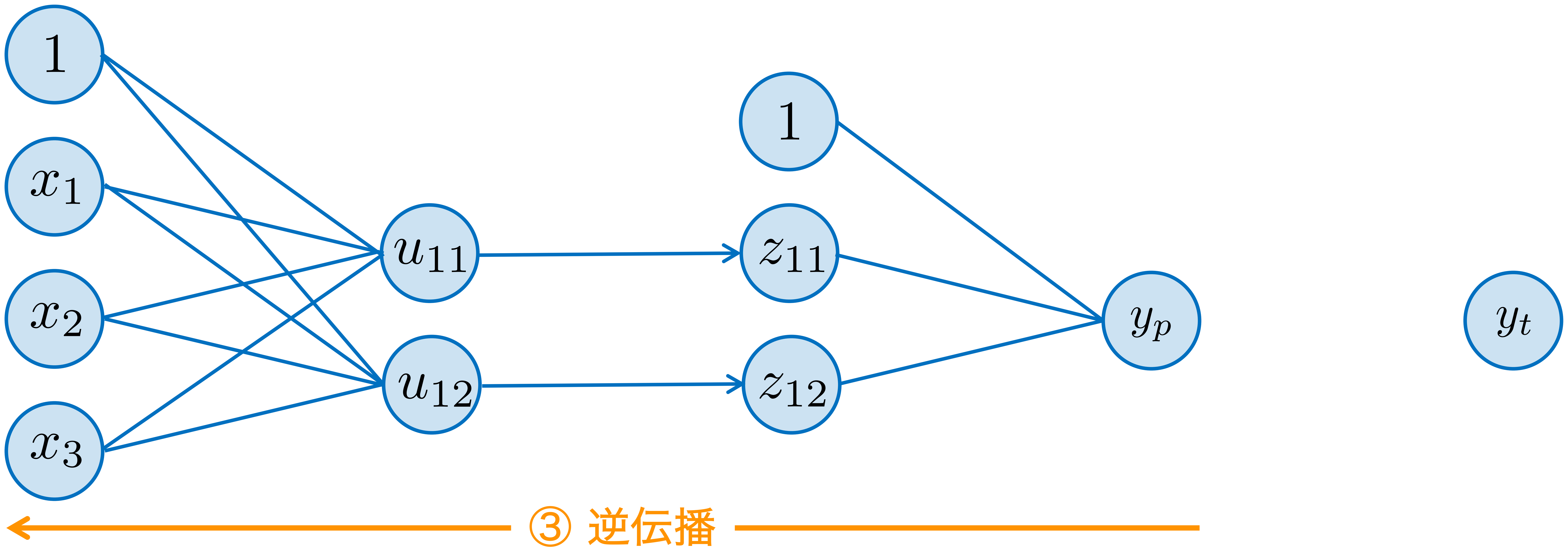
➢ 回帰問題：平均二乗誤差 (MSE) $\mathcal{L} = \frac{1}{N}\sum_{j=1}^{N}(y_p - y_t)^2$

➢ 分類問題：交差エントロピー誤差 $\mathcal{L} = -\sum_{j=1}^{N} y_t^j \log y_p^j$

4. 逆伝播

■なぜ逆伝播する？

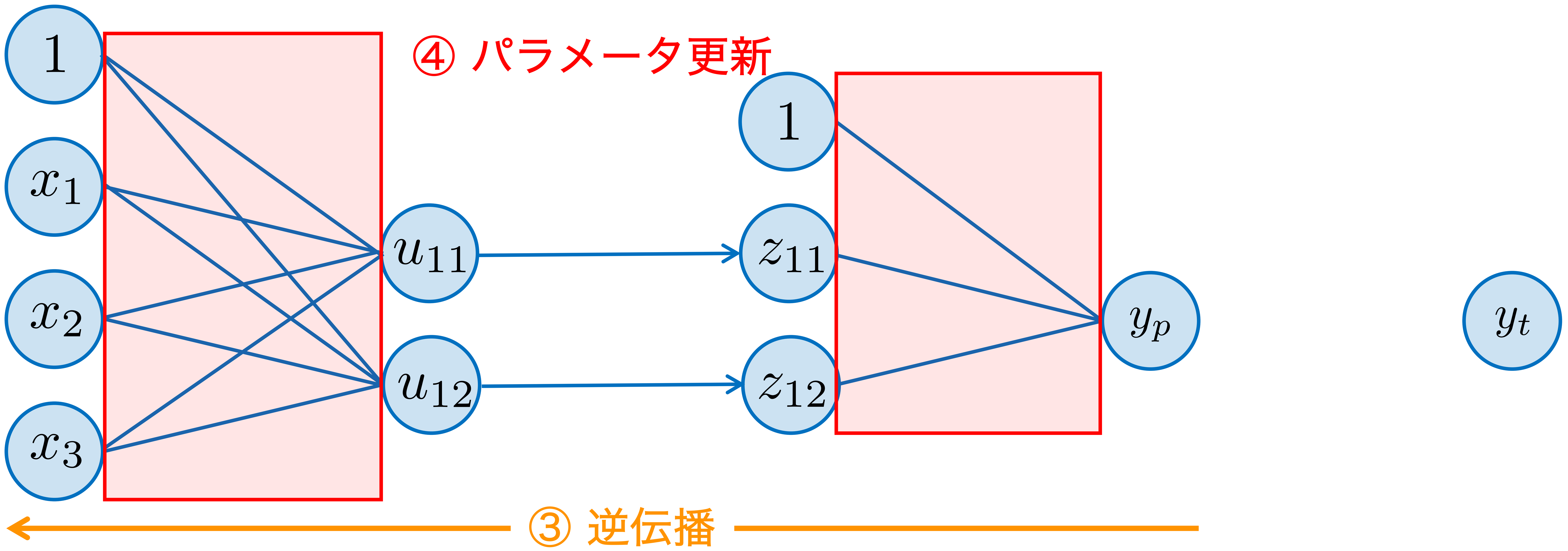
- (誤差)逆伝播：重みパラメータに対する損失関数の勾配を計算する



4. 逆伝播

■なぜ逆伝播する？

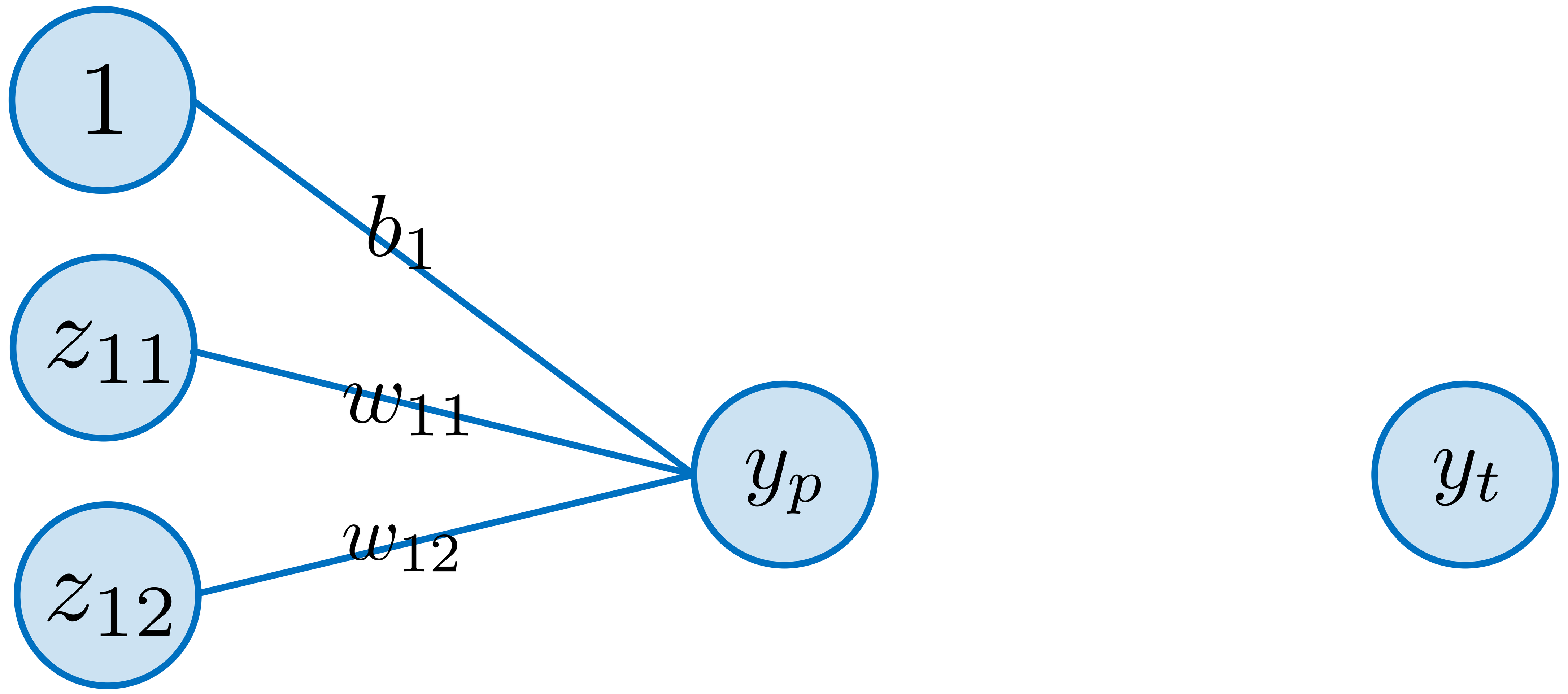
- (誤差)逆伝播：重みパラメータに対する損失関数の勾配を計算する



➢ パラメータ更新に損失関数の勾配が必要！

4.　逆伝播

■出力層



$$
\begin{aligned}
y_p &= w_{11}z_{11} + w_{12}z_{12} + b_1 \\
&= \sum_{j=1}^{2} w_{1j}z_{1j} + b_1
\end{aligned}
$$

$$\mathcal{L} = (y_p - y_t)^2$$

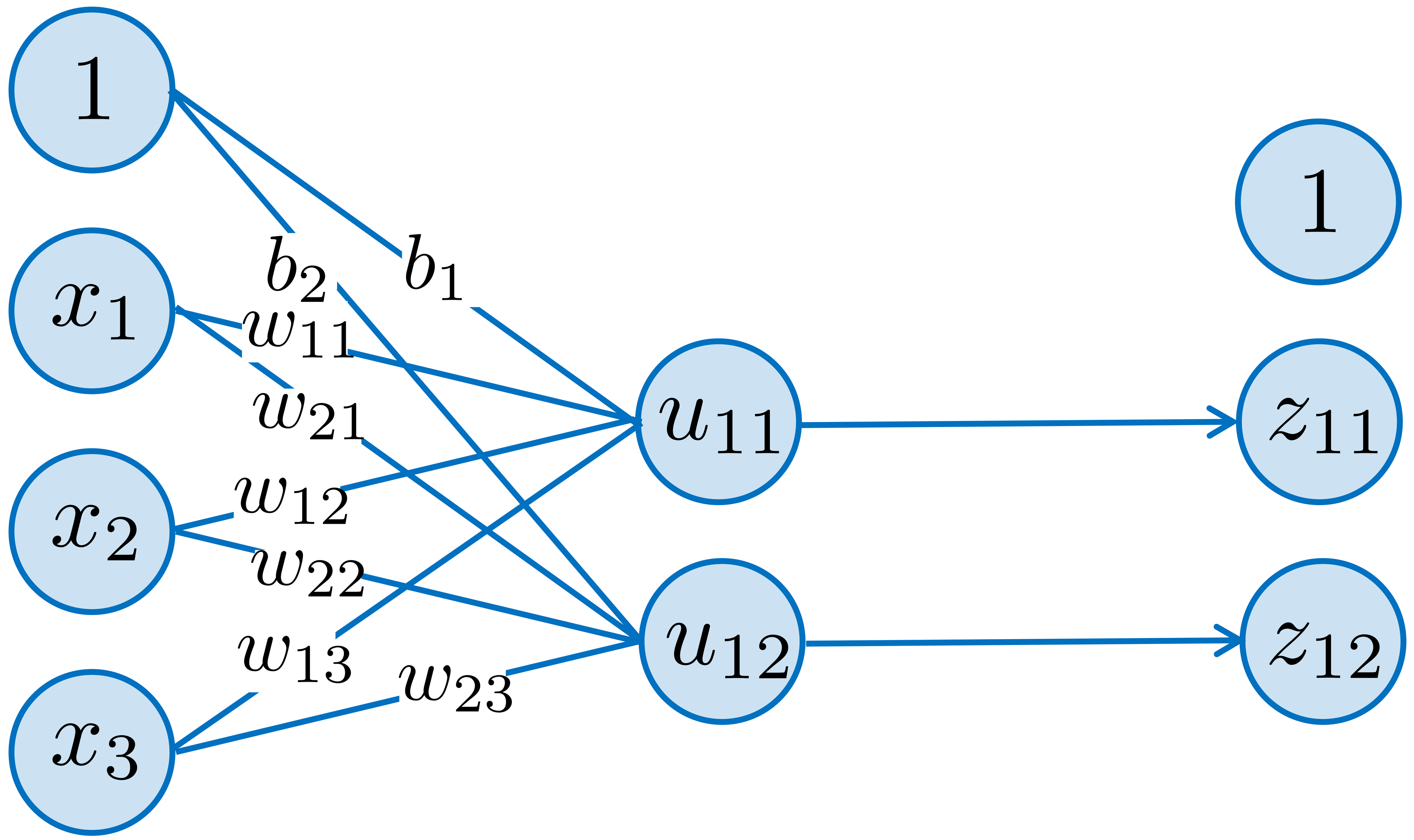
$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial w_{11}} &= \frac{\partial \mathcal{L}}{\partial y_p}\frac{\partial y_p}{\partial w_{11}} \\
&= 2(y_p - y_t)z_{11}
\end{aligned}
$$

$$\frac{\partial \mathcal{L}}{\partial w_{12}} = 2(y_p - y_t)z_{12}$$

$$\frac{\partial \mathcal{L}}{\partial b_1} = 2(y_p - y_t)$$

4. 逆伝播

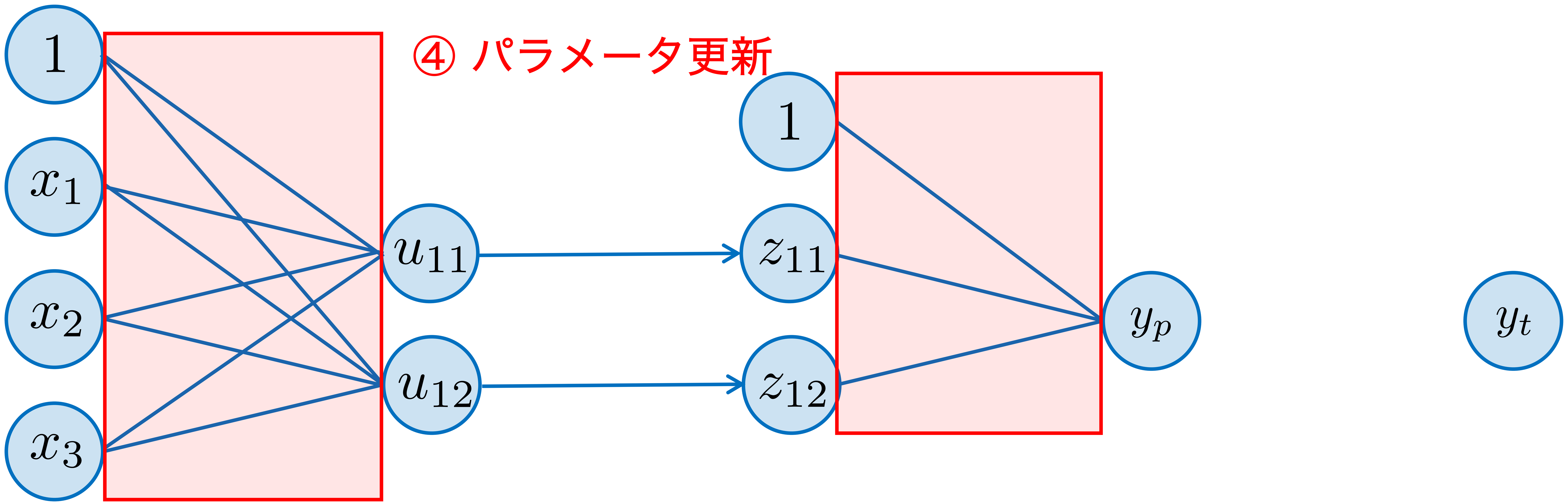
■隠れ層



$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial w_{11}} &= \frac{\partial \mathcal{L}}{\partial y_p}\frac{\partial y_p}{\partial w_{11}} \\
&= \frac{\partial \mathcal{L}}{\partial y_p}\frac{\partial y_p}{\partial z_{11}}\frac{\partial z_{11}}{\partial w_{11}} \\
&= \frac{\partial \mathcal{L}}{\partial y_p}\frac{\partial y_p}{\partial z_{11}}\frac{\partial z_{11}}{\partial u_{11}}\frac{\partial u_{11}}{\partial w_{11}} \\
&= 2(y_p - y_t)\cdot w_{11}\cdot \boxed{\Theta(u_{11})}\cdot x_1
\end{aligned}
$$

ReLUの微分

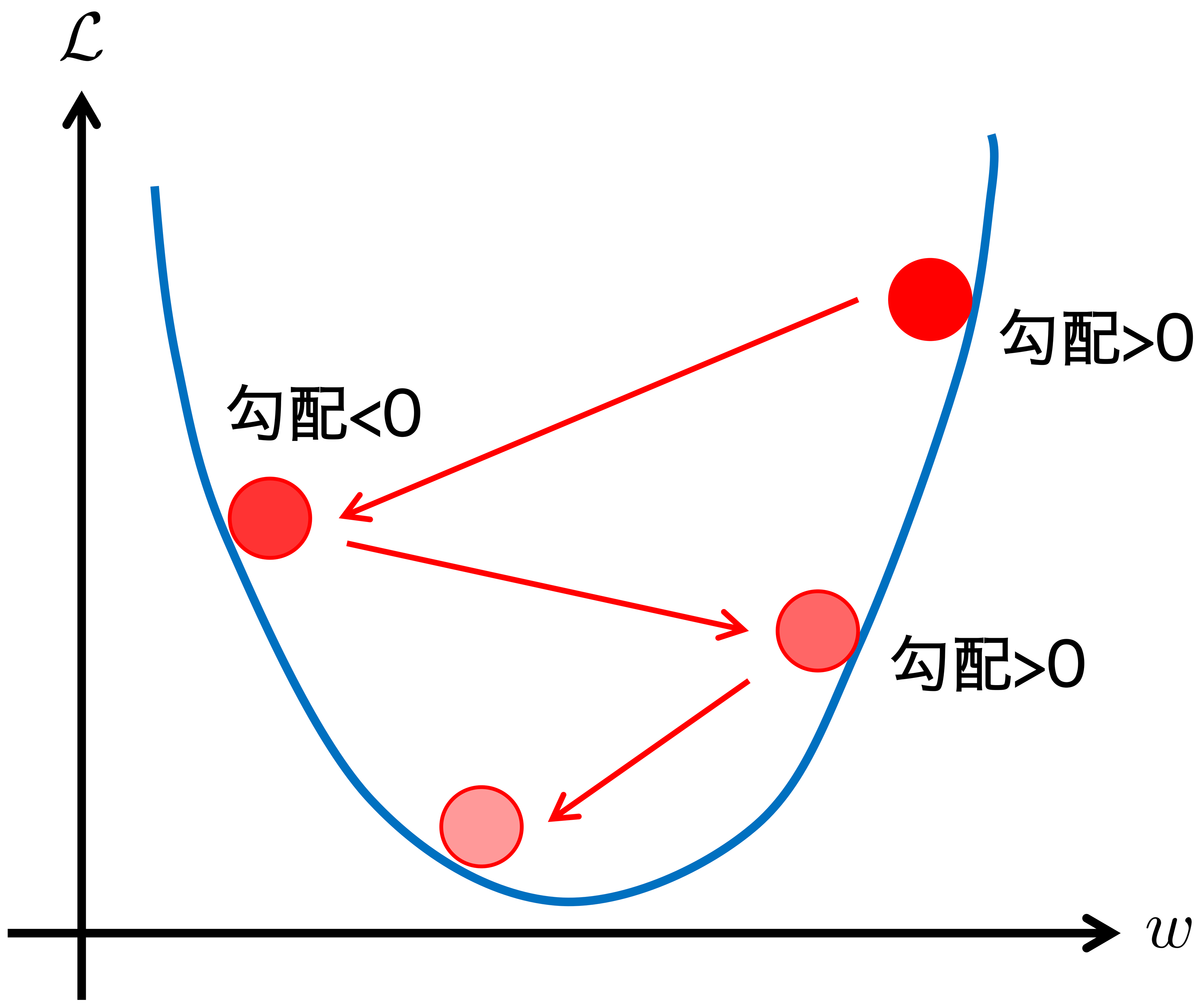
5. パラメータ更新



5. パラメータ更新

■勾配降下法

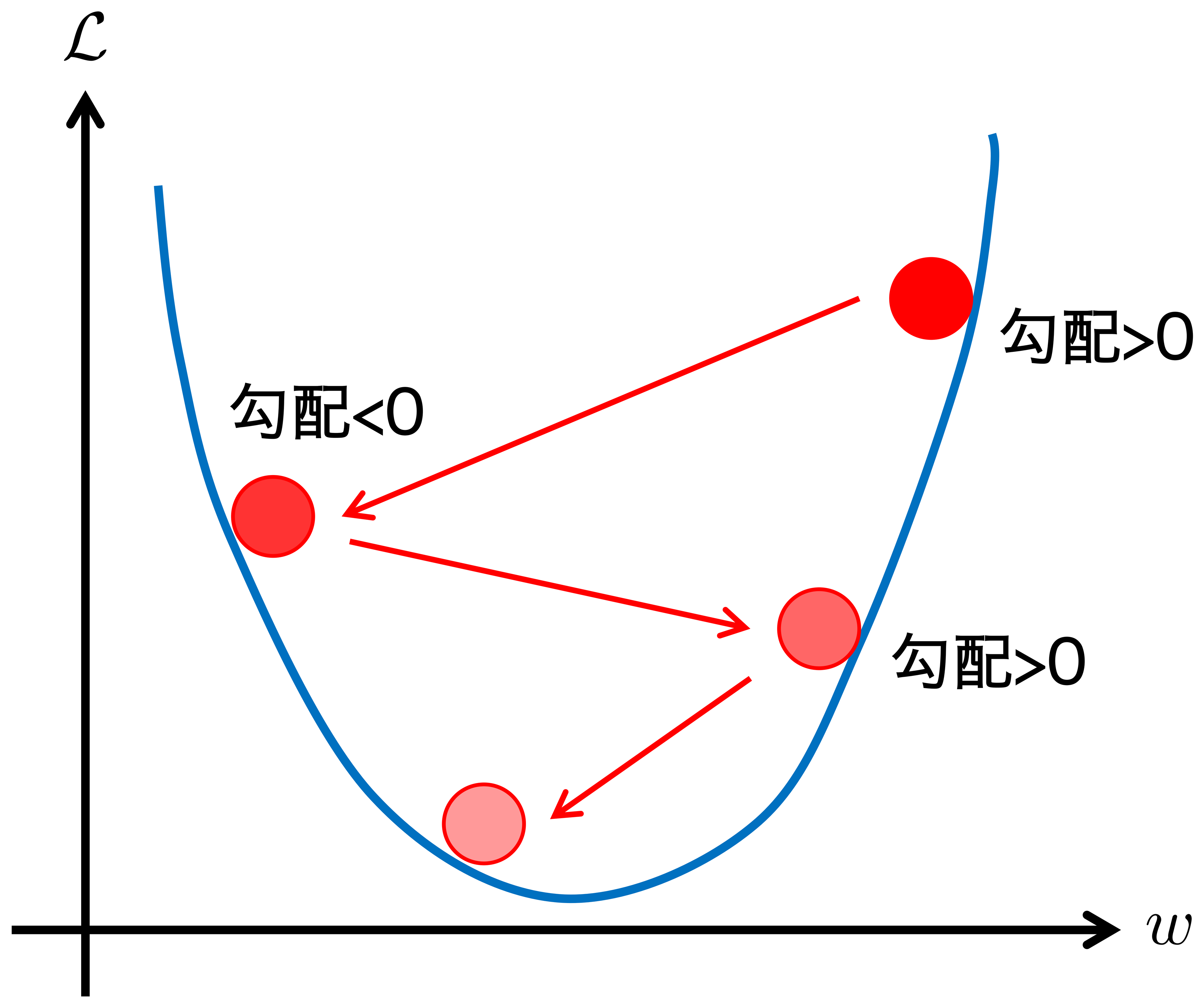
$$w = w - \eta \frac{\partial \mathcal{L}}{\partial w} \qquad \eta：学習率$$



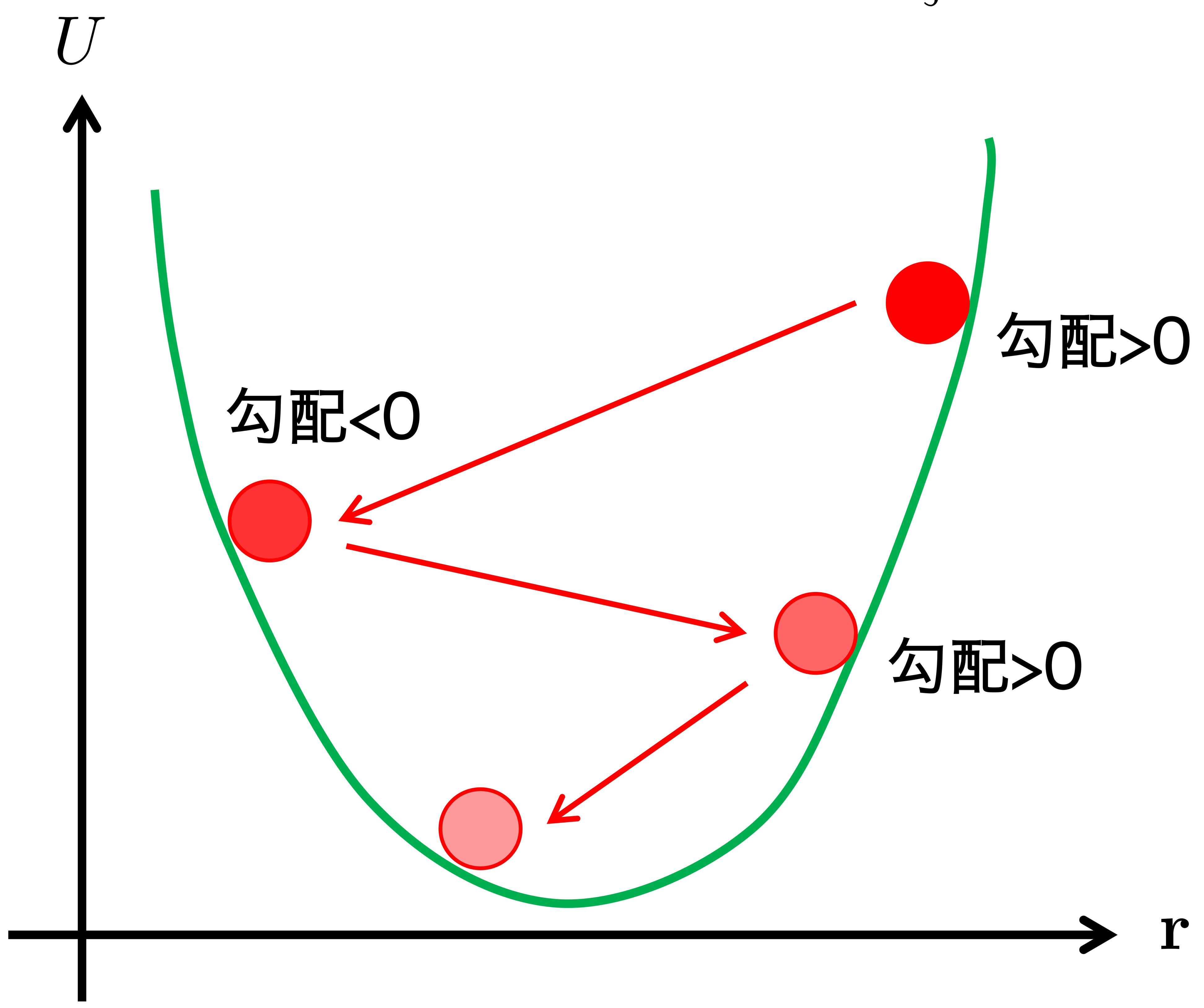
5. パラメータ更新

■勾配降下法

$$w = w - \eta \frac{\partial \mathcal{L}}{\partial w} \qquad \eta：学習率$$



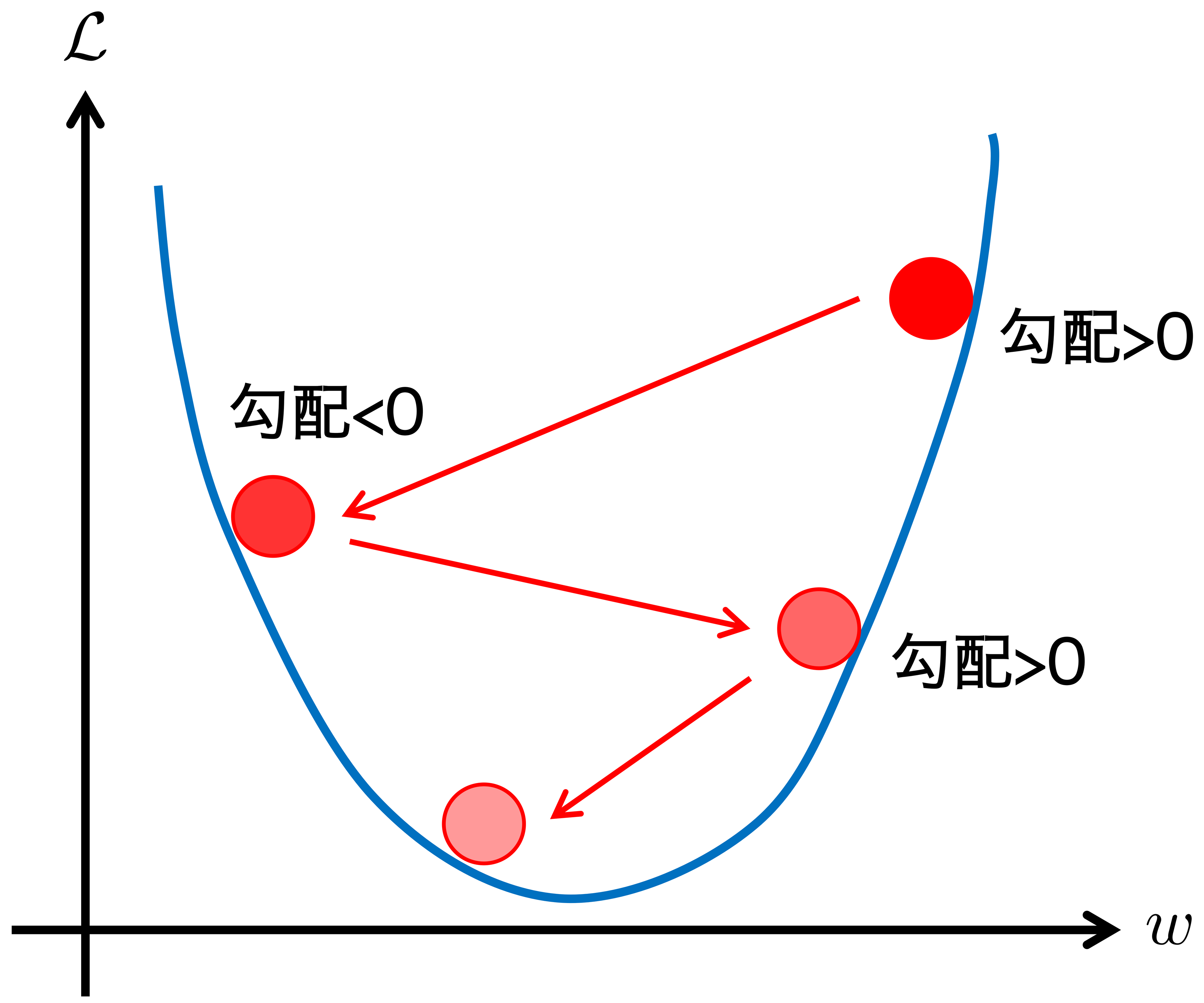
$$\zeta \mathbf{v} = -\nabla U(\mathbf{r}) \quad \rightarrow \quad \mathbf{r} = \mathbf{r} - \frac{\Delta t}{\zeta} \nabla U(\mathbf{r})$$



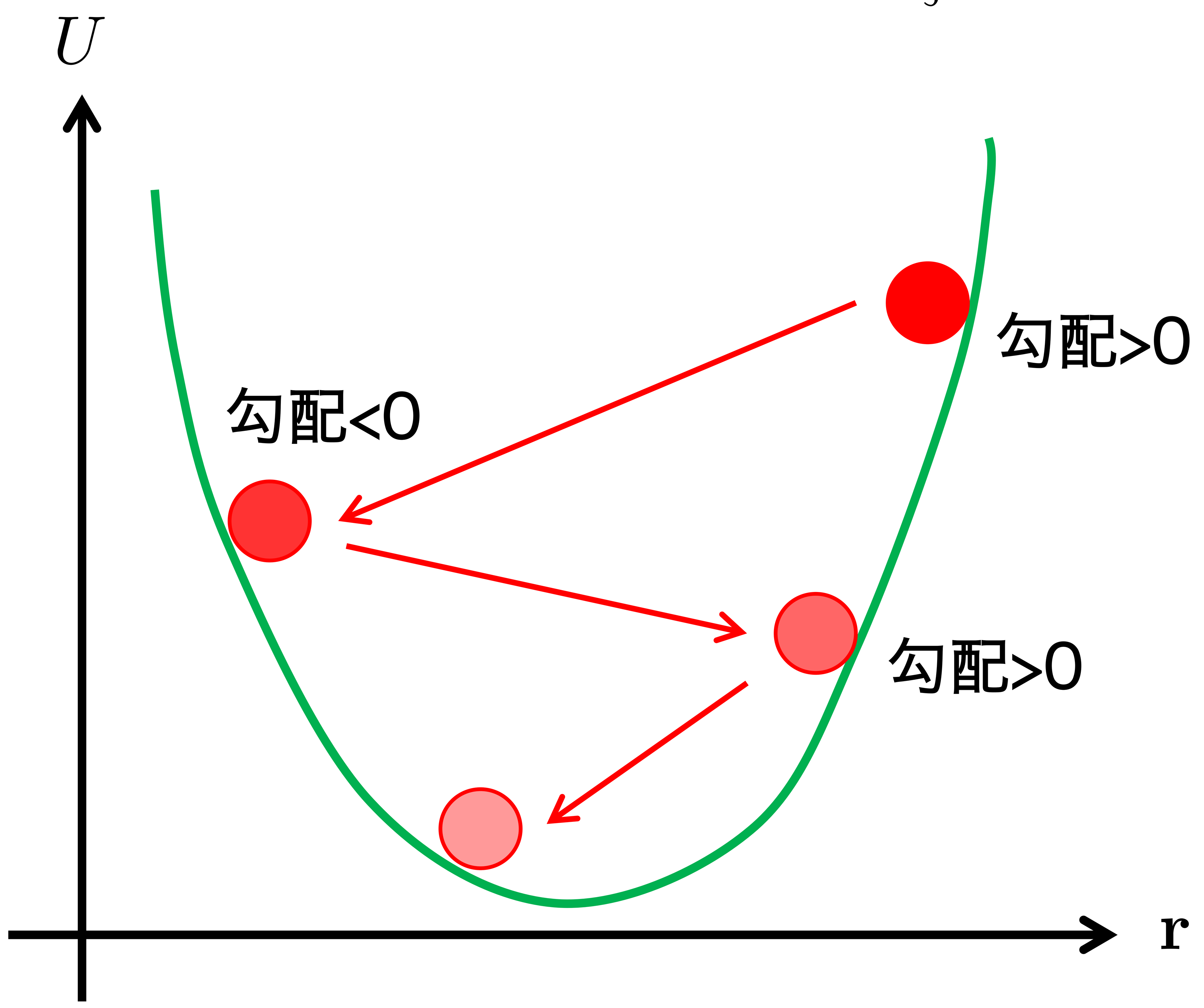
5. パラメータ更新

■勾配降下法

$$w = w - \eta \frac{\partial \mathcal{L}}{\partial w}$$ η：学習率



$$\zeta \mathbf{v} = -\nabla U(\mathbf{r}) \quad \rightarrow \quad \mathbf{r} = \mathbf{r} - \frac{\Delta t}{\zeta} \nabla U(\mathbf{r})$$



- 短所
  - 局所極小値から抜け出すことができない
  - 学習率の調整が難しい

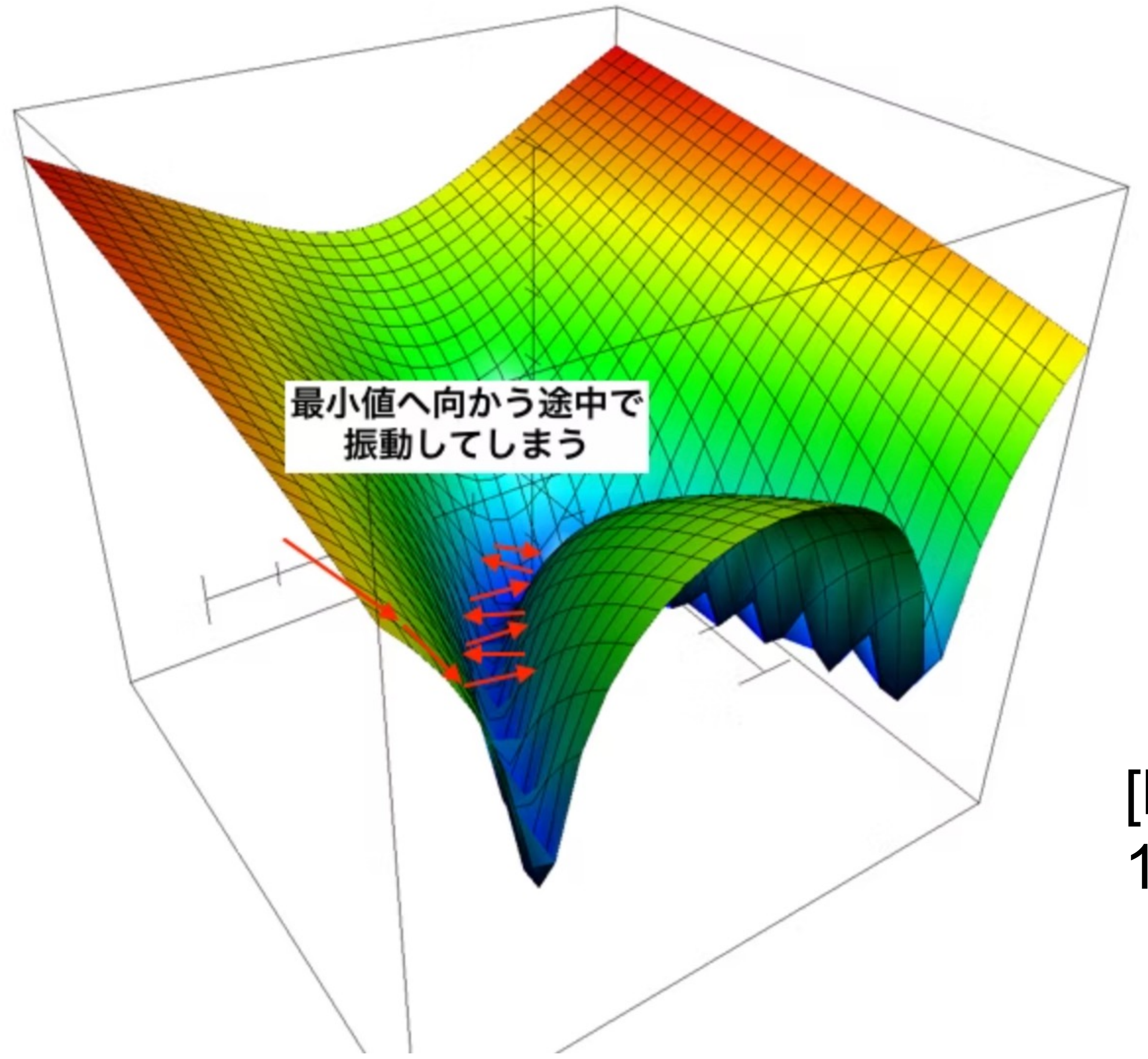
5. パラメータ更新

■確率的勾配降下法 (SGD)

- $\mathbf{w} = \mathbf{w} - \eta \nabla \mathcal{L}_d$
  - ➢ $\mathcal{L}_d$: ランダムに選んだデータのみの損失関数

- SGD : 一部のデータを用いて，パラメータを更新
  - ➢ ランダムにデータを選ぶ (← stochastic)
  - ➢ 局所極小値に陥っても，次のデータの損失関数が大きいので，局所極小値から抜け出せる．
  - ➢ パラメータは再び大きな値に更新され，極小値から抜け出す．

- 短所
  - ➢ 更新量が大きなため，学習が不安定.



Pathological Curvature

[https://qiita.com/omiita/items/1735c1d048fe5f611f80]

5. パラメータ更新

■Adam

- $$\begin{cases} \mathbf{v}_{t+1} = \beta_1 \mathbf{v}_t + (1-\beta_1)\nabla_{\mathbf{w}}\mathcal{L}_d(\mathbf{w}_t) \\ \mathbf{h}_{t+1} = \beta_2 \mathbf{h}_t + (1-\beta_2)\nabla_{\mathbf{w}}\mathcal{L}_d(\mathbf{w}_t) \odot \nabla_{\mathbf{w}}\mathcal{L}_d(\mathbf{w}_t) \\ \hat{\mathbf{v}}_{t+1} = \dfrac{\mathbf{v}_{t+1}}{1-\beta_1^{t+1}} \\ \hat{\mathbf{h}}_{t+1} = \dfrac{\mathbf{h}_{t+1}}{1-\beta_2^{t+1}} \\ \mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \dfrac{\hat{\mathbf{v}}_{t+1}}{\sqrt{\hat{\mathbf{h}}_{t+1}} + \epsilon} \end{cases}$$
- Momentum (慣性を導入) + RMSProp (学習率を調整)
- 現在最も機械学習において用いられている最適化手法

6. まとめ

■ニューラルネットワークによる学習の流れ

① 順伝播
線形変換($\mathrm{u = wx + b}$)+ 非線形変換(ReLU関数など)
↓
② 損失関数
(回帰)平均二乗誤差　(分類)交差エントロピー誤差
↓
③ 逆伝播
損失関数の勾配を計算
↓
④ パラメータ更新
勾配降下法，SGD，Adam

イテレーション
× バッチサイズ = エポック

- サンプルコード：リンク
- 問題：サンプルコードは回帰問題を扱っている．では，分類問題のコードを書いてみよう．