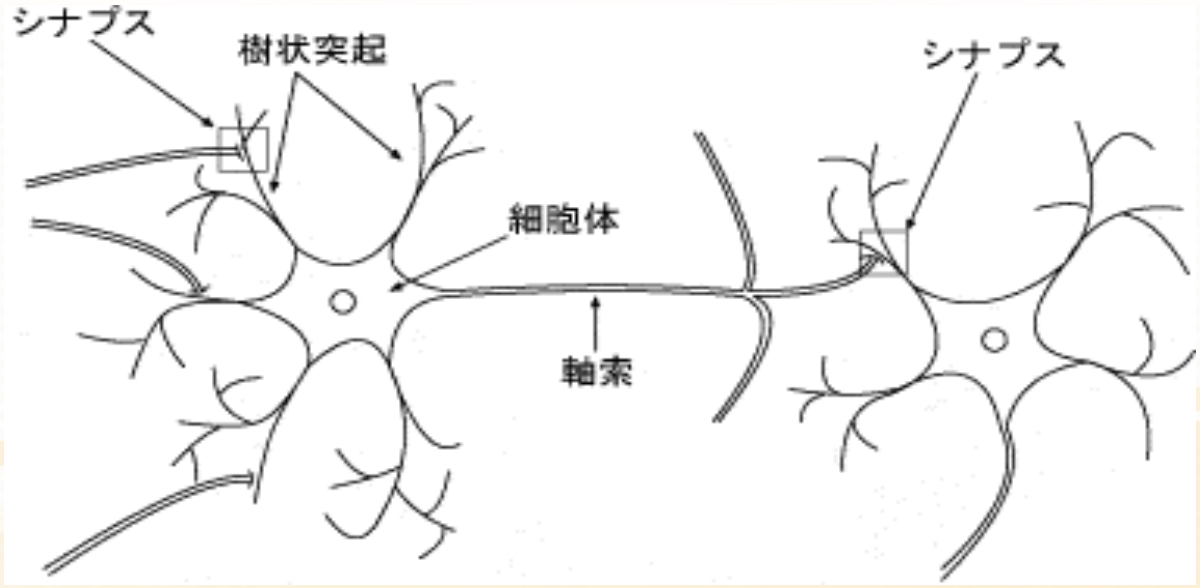
データサイエンス応用コース
ニューラルネットワークの構造と学習

高野　渉
大阪大学

脳神経系と人工知能

人間のような賢い意思決定・判断・推論・予測ができる計算をコンピュータの中で実現するにはどうすればいいだろうか？

人間の脳神経系を模擬した数理モデルを設計することを考える。この数理モデルがニューラルネットワークである。



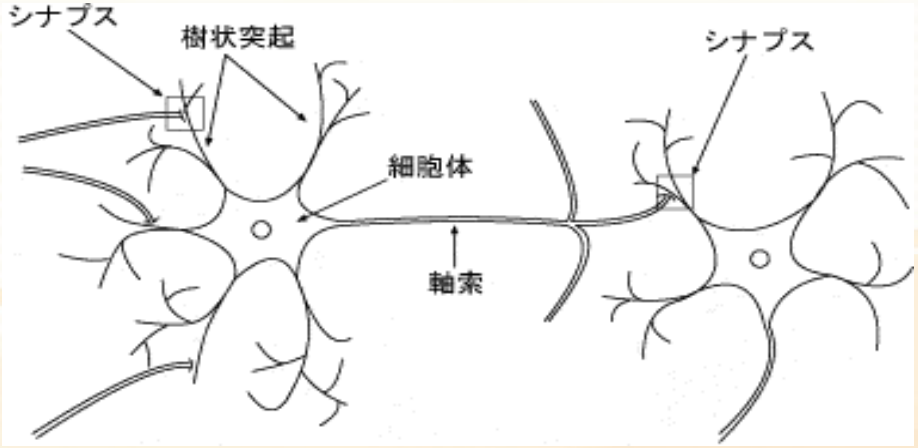
神経細胞の基本要素

細胞体：ニューロンの本体で、ここから樹状突起や軸索が伸びている。

樹状突起：細胞体から複雑に枝分かれする突起。神経伝達の入力部。

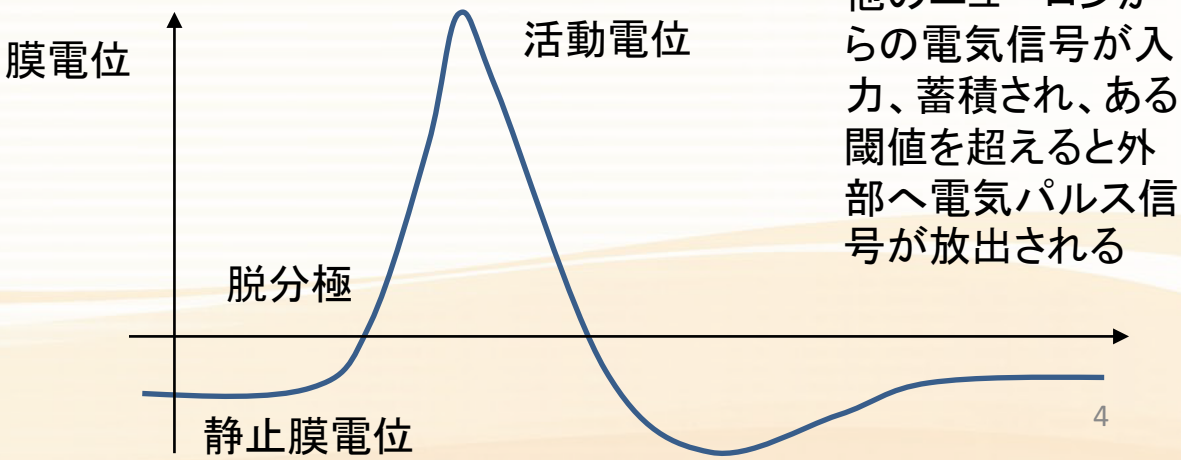
軸索：次の細胞に電気的に信号を伝達する神経線維。
細胞体から1本伸びた繊維は、次の細胞近傍で枝分かれする。

シナプス：神経接合部であり、ニューロンからニューロンへ電気的
信号を化学物質の信号に変換して、情報を伝達する部分。



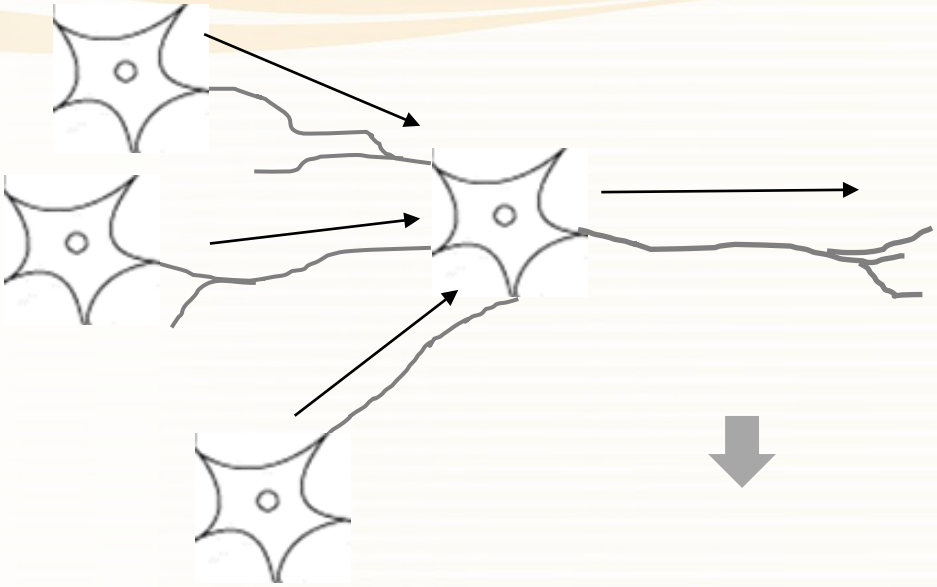
神経細胞の活動原理

①神経細胞は、イオンチャンネル・イオン交換によって一定の静止膜電位に保たれている。信号を受け取ると、膜電位がわずかに上昇する（脱分極）。
②これが一定の電位（閾値）を超えると、カリウムチャンネルが閉じ、ナトリウムチャンネルが開く。
③ナトリウムイオンが細胞に
入ってくることによって、内側電位が上昇する。
④電位を元に戻そうとカリウムチャンネルが開き、カリウムイオンが外へ、ナトリウムチャンネルが閉じる。
⑤電位が静止膜電位に戻る。

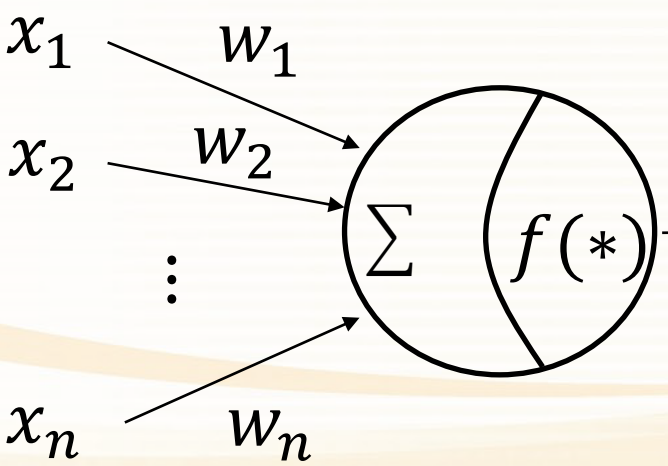


他のニューロンからの電気信号が入力、蓄積され、ある閾値を超えると外部へ電気パルス信号が放出される

ニューラルネットワーク



他のニューロンから樹状突起、シナプスを介して受け取った電気信号の総和が膜電位の上昇となり、活動電位となり電気信号を放出する



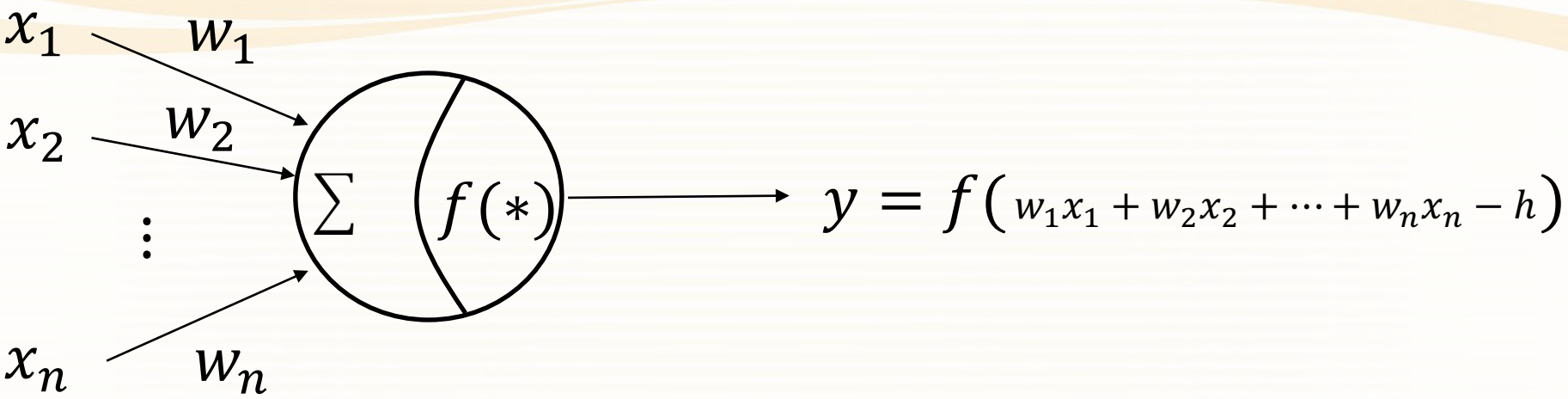
他のニューロンからの入力信号　x_i
ニューロン間のシナプス結合　w_i
ニューロンからの出力信号　y
膜電位の閾値　h

$$y = f\left(w_1x_1 + w_2x_2 + \cdots + w_nx_n - h\right)$$

活動電位の関数f、例えば

$$f(x) = \begin{cases} 0 & (x \leq 0) \\ 1 & (x > 0) \end{cases}$$

ニューラルネットワーク



学習データ： 入力 $\boldsymbol{x}^{(i)}$、出力 $t^{(i)}$ $(i = 1,2,\cdots,N)$ が与えられているときに、
ニューラルネットワークが学習データの入出力関係（$\boldsymbol{x}^{(i)}$から$t^{(i)}$が出力）となるように
シナプス重みパラメータ $\boldsymbol{w}$ を調整する（最適化する）。
これによって、所望する関数を学習することになる。

ただし、

$\boldsymbol{x} = (x_1, x_2, \cdots, x_n)$, $\boldsymbol{w} = (w_1, w_2, \cdots, w_n)$

ニューラルネットワークと論理回路

（命題A） 国語試験が90点以上
（命題B） 算数試験が90点以上
（命題Z） 進級

「国語試験が90点以上」かつ「算数試験が90点以上」ならば、「進級」する。
それ以外ならば、「進級」しない。

これを論理式で記述する。

「命題Aが真（正しい）」かつ「命題Bが真」ならば、「命題Zが真」

「命題Aが偽（真でない）」かつ「命題Bが真」ならば、「命題Zが偽」

「命題Aが真」かつ「命題Bが偽」ならば、「命題Zが偽」

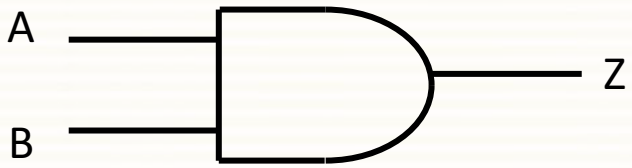
「命題Aが偽」かつ「命題Bが偽」ならば、「命題Zが偽」

ニューラルネットワークと論理回路

「真」を1、「偽」をOとすると、上記の論理式を以下のような表としてまとめることができる。

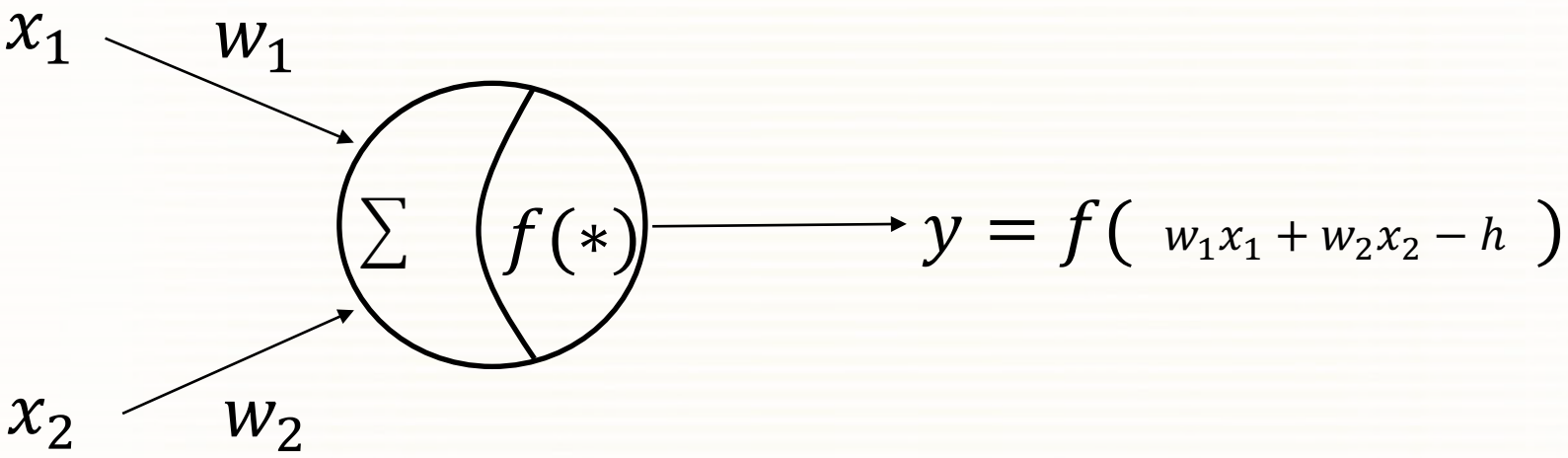
命題A	命題B	命題Z
1	1	1
0	1	0
1	0	0
0	0	0

このような関係を論理積（ANDゲート）と呼ぶ。



ニューラルネットワークと論理回路

論理積（ANDゲート）をニューラルネットで表すことを考える。



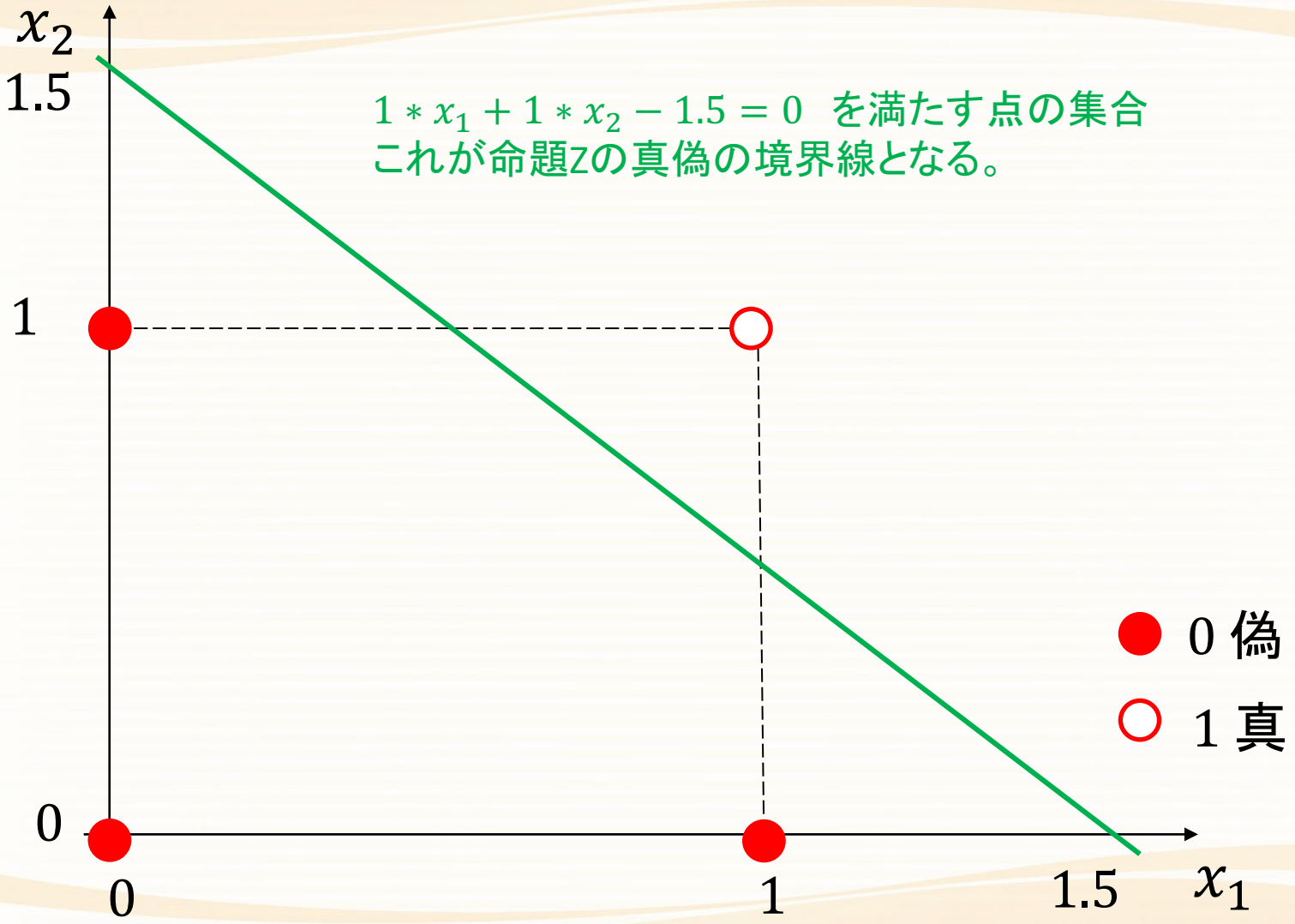
$$w_1 = 1, w_2 = 1, h = 1.5$$

$$f(x) = \begin{cases} 0 & (x \leq 0) \\ 1 & (x > 0) \end{cases}$$

x_1	x_2	$w_1x_1 + w_2x_2 - h$	$f(*)$
1	1	0.5	1
0	1	-0.5	0
1	0	-0.5	0
0	0	-1.5	0

とすると、右のような論理積を
実現することができる。

ニューラルネットワークと論理回路



ニューラルネットワークと論理回路

（命題A） 国語試験が90点以上
（命題B） 算数試験が90点以上
（命題Z） 進級

「国語試験が90点以上」または「算数試験が90点以上」ならば、「進級」する。
それ以外ならば、「進級」しない。

これを論理式で記述する。

「命題Aが真（正しい）」または「命題Bが真」ならば、「命題Zが真」

「命題Aが偽（真でない）」または「命題Bが真」ならば、「命題Zが真」

「命題Aが真」または「命題Bが偽」ならば、「命題Zが真」

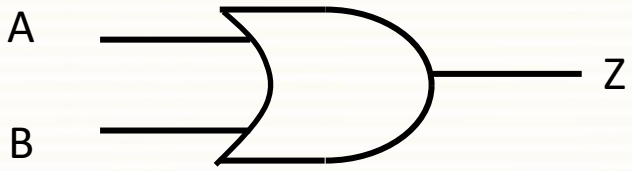
「命題Aが偽」または「命題Bが偽」ならば、「命題Zが偽」

ニューラルネットワークと論理回路

「真」を1、「偽」を0とすると、上記の論理式を以下のような表としてまとめることができる。

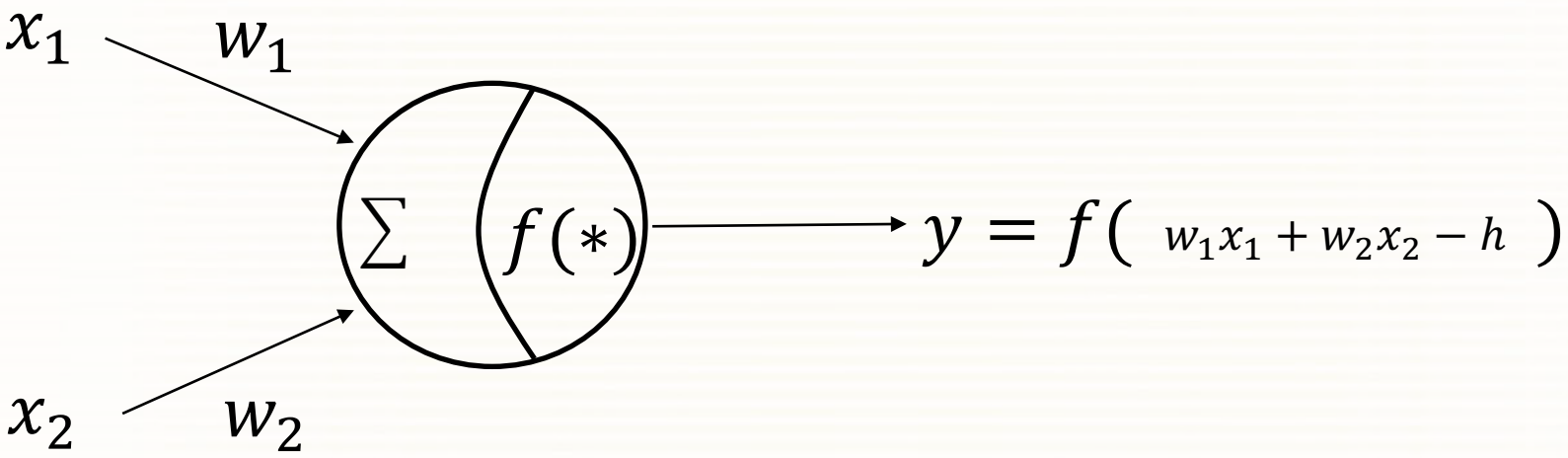
命題A	命題B	命題Z
1	1	1
0	1	1
1	0	1
0	0	0

このような関係を論理和（ORゲート）と呼ぶ。



ニューラルネットワークと論理回路

論理和（ORゲート）をニューラルネットで表すことを考える。



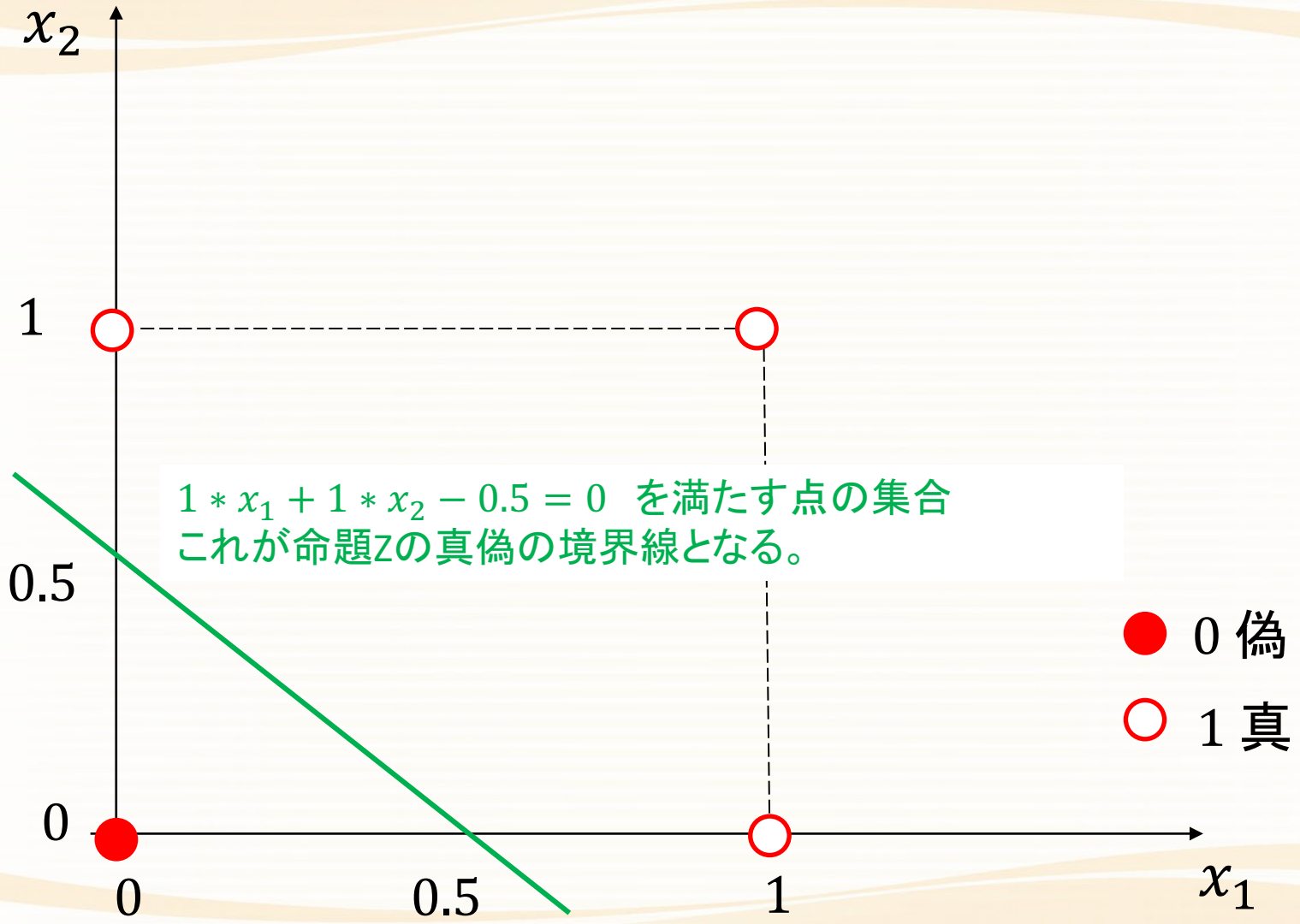
$$w_1 = 1, w_2 = 1, h = 0.5$$

$$f(x) = \begin{cases} 0 & (x \leq 0) \\ 1 & (x > 0) \end{cases}$$

x_1	x_2	$w_1x_1 + w_2x_2 - h$	$f(*)$
1	1	1.5	1
0	1	0.5	1
1	0	0.5	1
0	0	-0.5	0

とすると、右のような論理和を
実現することができる。

ニューラルネットワークと論理回路



ニューラルネットワークと論理回路

（命題A） 国語試験が90点以上
（命題B） 算数試験が90点以上
（命題Z） 落第

「国語試験が90点以上」かつ「算数試験が90点以上」ならば、「落第」しない。
それ以外ならば、「落第」する。

これを論理式で記述する。

「命題Aが真（正しい）」かつ「命題Bが真」ならば、「命題Zが偽」

「命題Aが偽（真でない）」かつ「命題Bが真」ならば、「命題Zが真」

「命題Aが真」かつ「命題Bが偽」ならば、「命題Zが真」

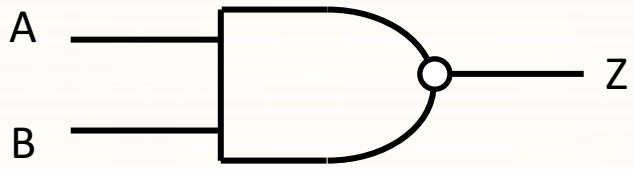
「命題Aが偽」かつ「命題Bが偽」ならば、「命題Zが真」

ニューラルネットワークと論理回路

「真」を1、「偽」を0とすると、上記の論理式を以下のような表としてまとめることができる。

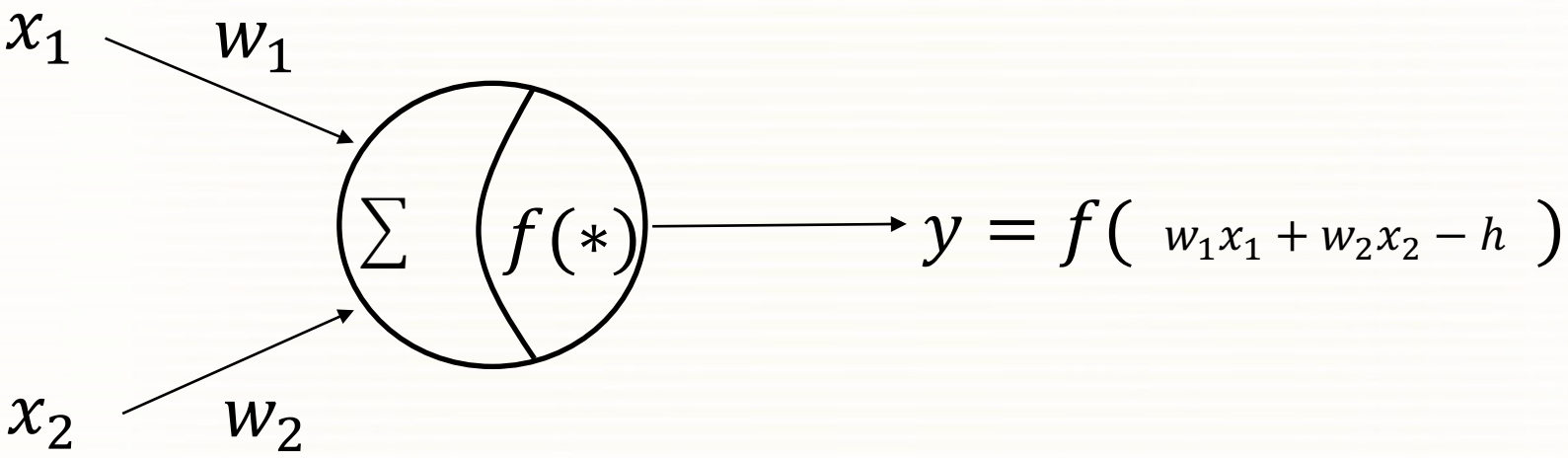
命題A	命題B	命題Z
1	1	0
0	1	1
1	0	1
0	0	1

このような関係を否定論理積（NANDゲート Not AND）と呼ぶ。



ニューラルネットワークと論理回路

否定論理積（NANDゲート）をニューラルネットで表すことを考える。



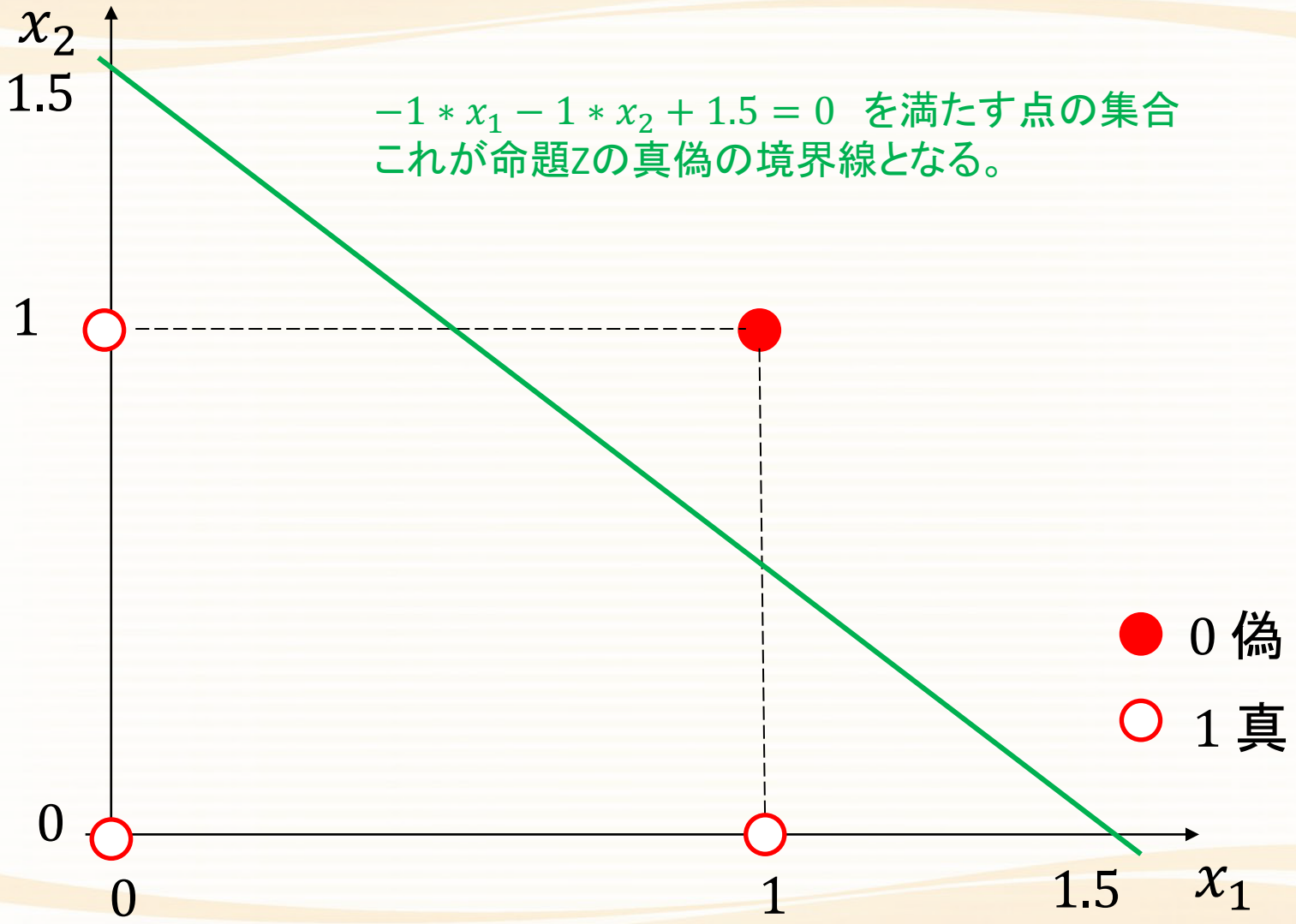
$$w_1 = -1, w_2 = -1, h = -1.5$$

$$f(x) = \begin{cases} 0 & (x \leq 0) \\ 1 & (x > 0) \end{cases}$$

x_1	x_2	$w_1x_1 + w_2x_2 - h$	$f(*)$
1	1	-0.5	0
0	1	0.5	1
1	0	0.5	1
0	0	1.5	1

とすると、右のような否定論理積を
実現することができる。

ニューラルネットワークと論理回路



ニューラルネットワークと論理回路

（命題A） 国語試験が90点以上
（命題B） 算数試験が90点以上
（命題Z） クラスそのまま

「国語試験が90点以上」かつ「算数試験が90点以上」ならば、上のクラスへ移動。
「国語試験が90点未満」かつ「算数試験が90点以上」ならば、クラスそのまま。
「国語試験が90点以上」かつ「算数試験が90点未満」ならば、クラスそのまま。
「国語試験が90点未満」かつ「算数試験が90点未満」ならば、下のクラスへ移動。

これを論理式で記述する。

「命題Aが真（正しい）」かつ「命題Bが真」ならば、「命題Zが偽」

「命題Aが偽（真でない）」かつ「命題Bが真」ならば、「命題Zが真」

「命題Aが真」かつ「命題Bが偽」ならば、「命題Zが真」

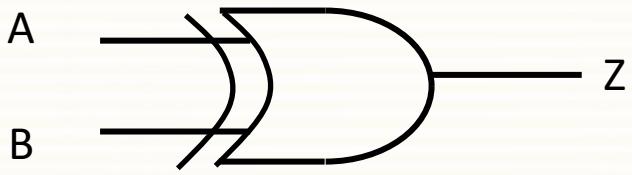
「命題Aが偽」かつ「命題Bが偽」ならば、「命題Zが偽」

ニューラルネットワークと論理回路

「真」を1、「偽」を0とすると、上記の論理式を以下のような表としてまとめることができる。

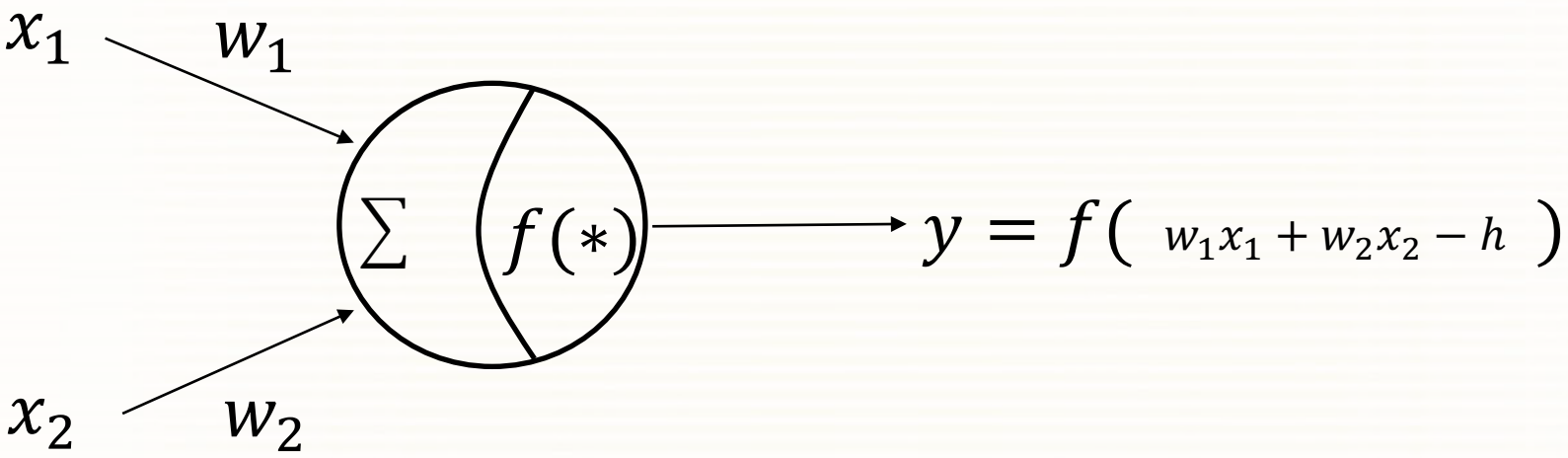
命題A	命題B	命題Z
1	1	0
0	1	1
1	0	1
0	0	0

このような関係を排他的論理和（XORゲート eXclusive OR）と呼ぶ。



ニューラルネットワークと論理回路

排他的論理和（XORゲート）をニューラルネットで表すことを考える。

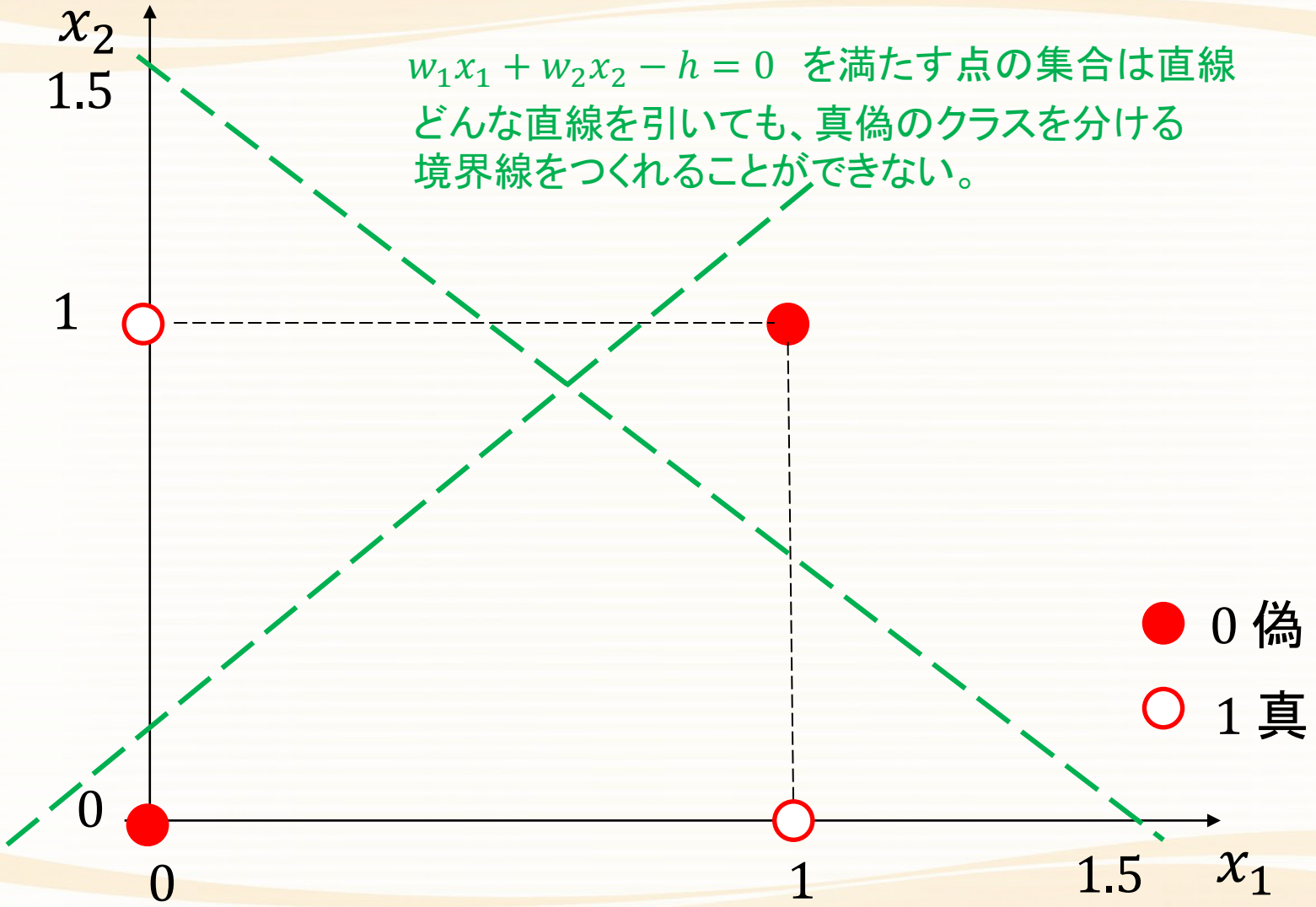


$$f(x) = \begin{cases} 0 & (x \leq 0) \\ 1 & (x > 0) \end{cases}$$

x_1	x_2	$w_1x_1 + w_2x_2 - h$	$f(*)$
1	1		0
0	1		1
1	0		1
0	0		0

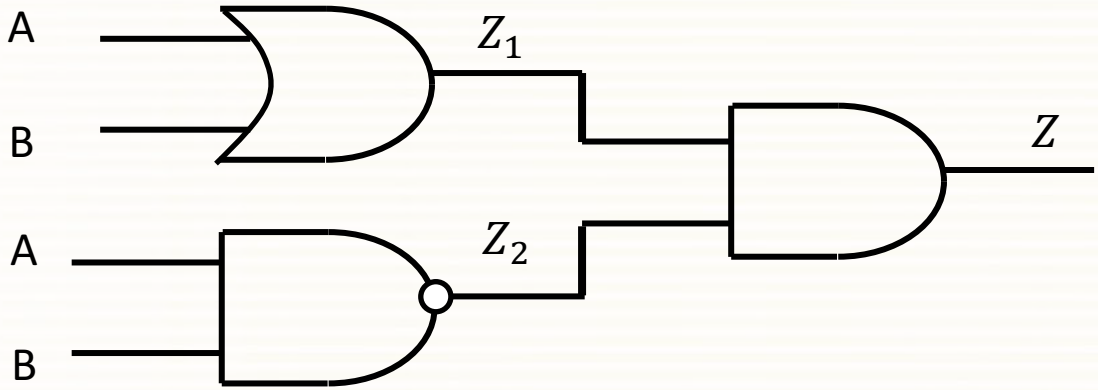
w_1, w_2, h をどのような値に設定しても排他的論理和を満たすような関数fを作ることができない

ニューラルネットワークと論理回路



ニューラルネットワークと論理回路

排他的論理和を作成できないことを見た。
しかし、複数の論理回路を組み合わせると否定論理和を
実現することができることを見てみよう。



A	B	Z_1	Z_2
1	1	1	0
0	1	1	1
1	0	1	1
0	0	0	1

Z_1	Z_2	Z
1	0	0
1	1	1
1	1	1
0	1	0

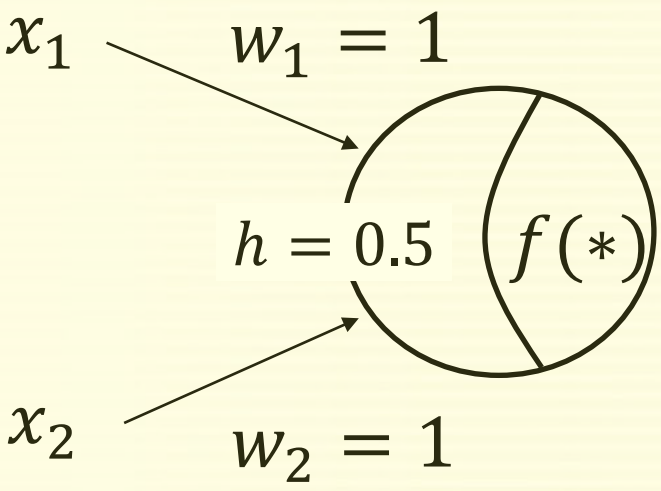
論理和（OR）と否定論理積(NAND)を並置し、その結果を論理積(AND)に入力するような階層構造を設計すると排他的論理和（XOR）を作ることができる

ニューラルネットワークと論理回路

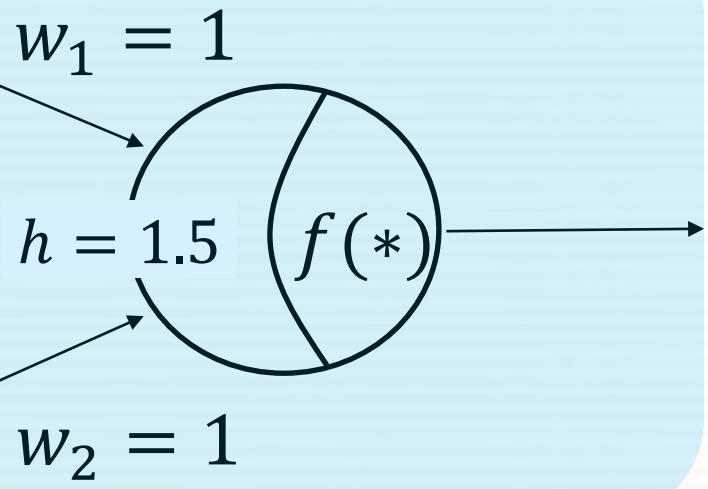
最も簡素な各モジュールを

単純パーセプトロン

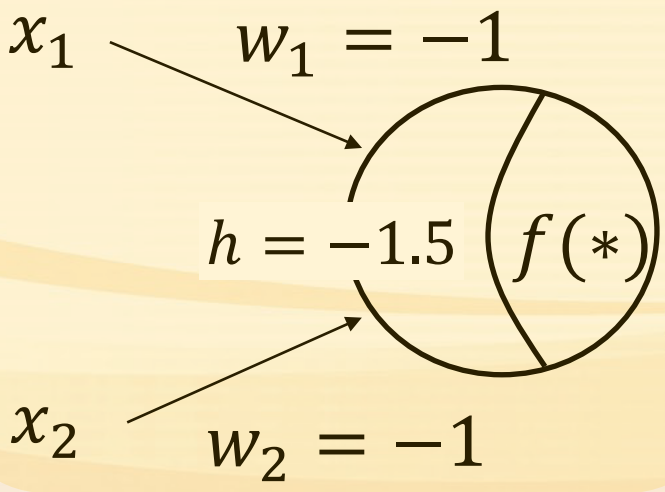
論理和（OR）



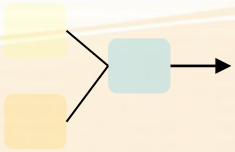
論理積（AND）



否定論理積(NAND)

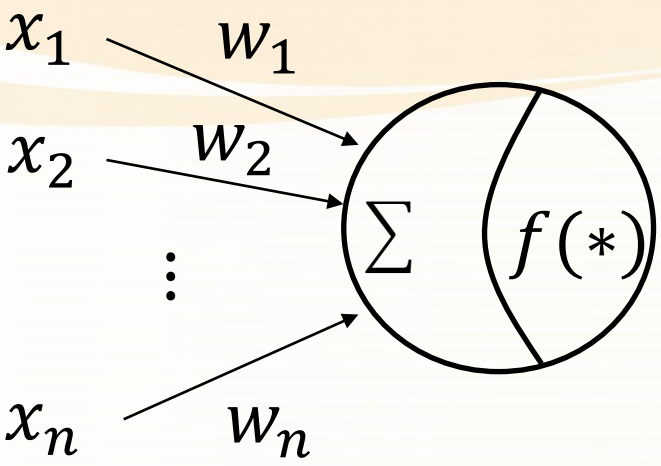


単パーセプトロンを階層的に配置する



多重パーセプトロン

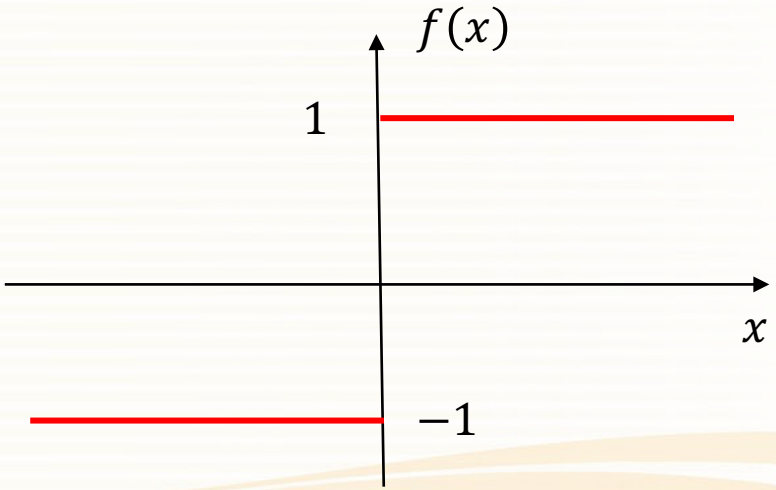
シナプス可塑性とHebb則



$$y = f\left(w_1x_1 + w_2x_2 + \cdots + w_nx_n\right)$$

活動電位の関数f、例えば

$$f(x) = \begin{cases} -1 & (x \leq 0) \\ 1 & (x > 0) \end{cases}$$



学習データ： 入力 $\boldsymbol{x}^{(i)}$、出力 $t^{(i)}$ $(i = 1,2,\cdots,N)$ が与えられているときに、
ニューラルネットワークが学習データの入出力関係（$\boldsymbol{x}^{(i)}$から$t^{(i)}$が出力）となるように
シナプス重みパラメータ $\boldsymbol{w}$ を調整する手順を考えてみる

シナプス可塑性とHebb則

学習データ $t^{(i)}$ とニューラルネットワークの出力
$f(w_1 {x_1}^{(i)} + w_2 {x_2}^{(i)} + \cdots + w_n {x_n}^{(i)})$ の誤差を考える。

1. $w_1 {x_1}^{(i)} + w_2 {x_2}^{(i)} + \cdots + w_n {x_n}^{(i)} \leq 0$ かつ $t^{(i)} = -1$ なら
   学習データとニューラルネットの出力が一致（出力は正解）

2. $w_1 {x_1}^{(i)} + w_2 {x_2}^{(i)} + \cdots + w_n {x_n}^{(i)} \leq 0$ かつ $t^{(i)} = 1$ なら
   学習データとニューラルネットの出力が不一致（出力は不正解）

3. $w_1 {x_1}^{(i)} + w_2 {x_2}^{(i)} + \cdots + w_n {x_n}^{(i)} > 0$ かつ $t^{(i)} = -1$ なら
   学習データとニューラルネットの出力が不一致（出力は不正解）

4. $w_1 {x_1}^{(i)} + w_2 {x_2}^{(i)} + \cdots + w_n {x_n}^{(i)} > 0$ かつ $t^{(i)} = 1$ なら
   学習データとニューラルネットの出力が一致（出力は正解）

シナプス可塑性とHebb則

以上の場合をふまえると、

学習データとニューラルネットの出力が一致する場合：

$$\left(w_1 {x_1}^{(i)} + w_2 {x_2}^{(i)} + \cdots + w_n {x_n}^{(i)}\right) t^{(i)} \geq 0$$

学習データとニューラルネットの出力が一致しない場合：

$$\left(w_1 {x_1}^{(i)} + w_2 {x_2}^{(i)} + \cdots + w_n {x_n}^{(i)}\right) t^{(i)} < 0$$

という関係が成り立つ。

したがって、

$$\Phi = \sum_i \left(w_1 {x_1}^{(i)} + w_2 {x_2}^{(i)} + \cdots + w_n {x_n}^{(i)}\right) t^{(i)}$$

を目的関数として、目的関数が大きくなるように重みパラメータ w_j を修正する。

※最適化（参考）

シナプス可塑性とHebb則

$$w_j \rightarrow w_j + \epsilon \frac{\partial \Phi}{\partial w_j}$$

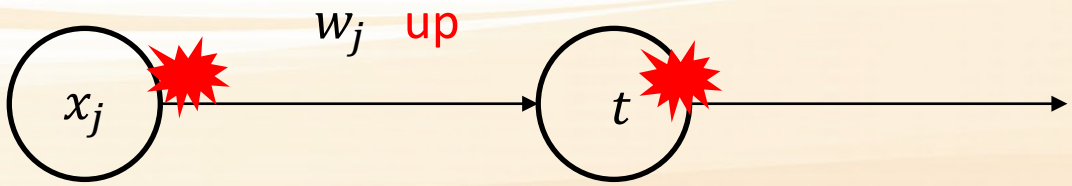
$$= w_j + \epsilon \sum_i x_j^{(i)} t^{(i)}$$

ϵ : 正の小さい定数

入力ニューロンx_jと出力ニューロンtがともに活性している場合、または不活性の場合は、重み w_j を増強する。

入力ニューロンx_jと出力ニューロンtの一方のみが活性している場合、重み w_j を抑制する。
ことを意味する（Hebb則）。

「2つのニューロンが同時に発火するときに、その間のシナプス結合は強まる」というシナプスの可塑性に一致することは興味深い



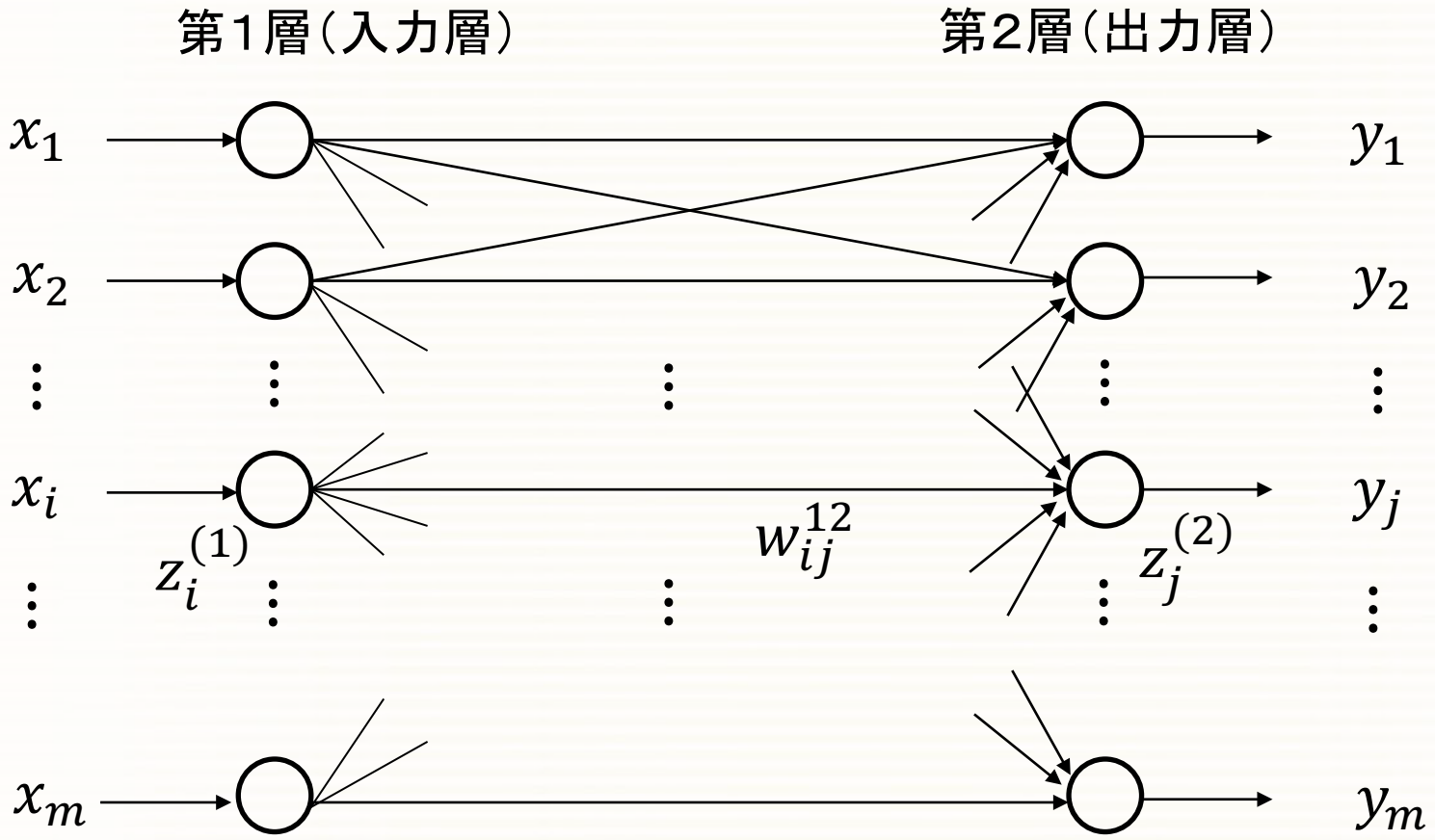
ニューラルネットワークの学習

ニューラルネットワークの学習は、
学習データの入力・出力関係をニューラルネットワークが表現するように、
　・シナプスの重み結合
　・活性化関数のパラメータ
を最適化することである。

最適化計算の方法・手順について、
・単純パーセプトロン
・３層ニューラルネットワーク
において説明する。

単純パーセプトロンの学習

単純パーセプトロン



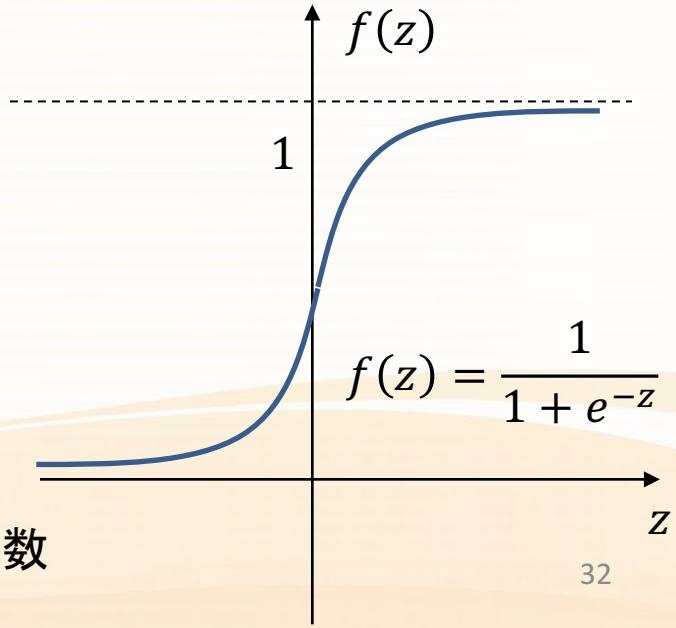
単純パーセプトロンの学習

第1層（入力層）における入力・内部状態・出力について

入力 $\boldsymbol{x}^{(0)} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}$ ➡ 内部状態 $\boldsymbol{z}^{(1)} = \begin{bmatrix} x_1 - h_1 \\ x_2 - h_2 \\ \vdots \\ x_m - h_m \end{bmatrix}$

h：バイアスパラメータ

➡ 出力 $\boldsymbol{x}^{(1)} = \begin{bmatrix} f(x_1 - h_1) \\ f(x_2 - h_2) \\ \vdots \\ f(x_m - h_m) \end{bmatrix}$



$f(z)$: 活性化関数であり、
右のようなシグモイド関数
を使用する

単純パーセプトロンの学習

第2層（出力層）における入力・内部状態・出力について

入力

$$\begin{bmatrix} \tilde{x}_1^{(2)} \\ \tilde{x}_2^{(2)} \\ \vdots \\ \tilde{x}_n^{(2)} \end{bmatrix} = \begin{bmatrix} w_{11}^{12}\, x_1^{(1)} + w_{21}^{12}\, x_2^{(1)} + \cdots + w_{m1}^{12}\, x_m^{(1)} \\ w_{12}^{12}\, x_1^{(1)} + w_{22}^{12}\, x_2^{(1)} + \cdots + w_{m2}^{12}\, x_m^{(1)} \\ \vdots \\ w_{1n}^{12}\, x_1^{(1)} + w_{2n}^{12}\, x_2^{(1)} + \cdots + w_{mn}^{12}\, x_m^{(1)} \end{bmatrix}$$

$$= \begin{bmatrix} w_{11}^{12} & w_{21}^{12} & \cdots & w_{m1}^{12} \\ w_{12}^{12} & w_{22}^{12} & \cdots & w_{m2}^{12} \\ & \vdots & & \\ w_{1n}^{12} & w_{2n}^{12} & \cdots & w_{mn}^{12} \end{bmatrix} \begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \\ \vdots \\ x_m^{(1)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{w}_1^{12\,T} \\ \boldsymbol{w}_2^{12\,T} \\ \vdots \\ \boldsymbol{w}_n^{12\,T} \end{bmatrix} \boldsymbol{x}^{(1)}$$

$$= \boldsymbol{W}^{\boldsymbol{12}}\, \boldsymbol{x}^{(1)}$$

単純パーセプトロンの学習

第2層（出力層）における入力・内部状態・出力について

内部状態

$$\boldsymbol{z}^{(2)} = \begin{bmatrix} z_1^{(2)} \\ z_2^{(2)} \\ \vdots \\ z_n^{(2)} \end{bmatrix} = \begin{bmatrix} \tilde{x}_1^{(2)} - h_1^{(2)} \\ \tilde{x}_2^{(2)} - h_2^{(2)} \\ \vdots \\ \tilde{x}_n^{(2)} - h_n^{(2)} \end{bmatrix} = \boldsymbol{W^{12}}\, \boldsymbol{x}^{(1)} - \boldsymbol{h}^{(2)}$$

出力

$$\boldsymbol{x}^{(2)} = \begin{bmatrix} f(z_1^{(2)}) \\ f(z_2^{(2)}) \\ \vdots \\ f(z_n^{(2)}) \end{bmatrix} = \boldsymbol{y}$$

単純パーセプトロンの学習

学習データ：入力 $\boldsymbol{u}$ → 出力 $\boldsymbol{t}$ となるようにパラメータを調整する

学習データの出力 $\boldsymbol{t}$ とニューラルネットワークの出力 $\boldsymbol{y}$ の誤差ベクトル

$$\boldsymbol{e} = \boldsymbol{y} - \boldsymbol{t}$$

評価関数を誤差の大きさとする

$$\boldsymbol{\varphi} = \frac{1}{2}\,\boldsymbol{e}^T\,\boldsymbol{e}$$

評価関数が小さくなるように最急降下法を用いてパラメータを修正する

単純パーセプトロンの学習

※1　出力層のバイアスパラメータ　$\boldsymbol{h}^{(2)}$　について

$$\frac{\partial \varphi}{\partial \boldsymbol{h}^{(2)}} = \boxed{\frac{\partial \varphi}{\partial \boldsymbol{e}}} \boxed{\frac{\partial \boldsymbol{e}}{\partial \boldsymbol{y}}} \boxed{\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{z}^{(2)}}} \boxed{\frac{\partial \boldsymbol{z}^{(2)}}{\partial \boldsymbol{h}^{(2)}}}$$

(1)　(2)　(3)　(4)

(1)　$\boldsymbol{e}^T$

(2)　I　(単位行列)

(3)
$$\begin{bmatrix} \partial f(z_1^{(2)})/\partial z_1^{(2)} & & \boldsymbol{O} \\ & \partial f(z_2^{(2)})/\partial z_2^{(2)} & \\ & & \ddots \\ \boldsymbol{O} & & \partial f(z_n^{(2)})/\partial z_n^{(2)} \end{bmatrix}$$

シグモイド関数の微分

$$\frac{\partial f(z)}{\partial z} = \frac{e^{-z}}{(1+e^{-z})^2}$$

(4)　$-I$

単純パーセプトロンの学習

※1　ちなみに、(1)～(4)の項を掛け合わせると

$$\frac{\partial \varphi}{\partial \boldsymbol{h}^{(2)}} = \left[-(y_1 - t_1)\frac{\partial f(z_1^{(2)})}{\partial z_1^{(2)}} \quad \cdots \quad -(y_n - t_n)\frac{\partial f(z_n^{(2)})}{\partial z_n^{(2)}} \right]$$

学習データごとに上記勾配を計算し、それらを足し合わせて
パラメータ　$\boldsymbol{h}^{(2)}$ の変更量 $\Delta \boldsymbol{h}^{(2)}$を計算する。

$$\Delta \boldsymbol{h}^{(2)} = \sum_{\boldsymbol{i}} \left(\frac{\partial \varphi_i}{\partial \boldsymbol{h}^{(2)}} \right)^{\boldsymbol{T}}$$

$$\boldsymbol{h}^{(2)} \leftarrow \boldsymbol{h}^{(2)} - \alpha\, \Delta \boldsymbol{h}^{(2)} \qquad (\alpha > 0)$$

単純パーセプトロンの学習

※２ シナプス重み結合のパラメータ $\boldsymbol{w}_k^{12}$ について

$$\frac{\partial \varphi}{\partial \boldsymbol{w}_k^{12}} = \boxed{\frac{\partial \varphi}{\partial \boldsymbol{e}}} \boxed{\frac{\partial \boldsymbol{e}}{\partial \boldsymbol{y}}} \boxed{\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{z}^{(2)}}} \boxed{\frac{\partial \boldsymbol{z}^{(2)}}{\partial \boldsymbol{w}_k^{12}}}$$

(1) (2) (3) (4)

(1)～(3)は既に計算済み

(4) $\begin{bmatrix} \mathbf{0} \\ \boldsymbol{x}^{(1)^T} \\ \mathbf{0} \end{bmatrix} < k$ (k行目以外はすべてゼロ)

単純パーセプトロンの学習

※２　ちなみに、(1)～(4)の項を掛け合わせると

$$\frac{\partial \varphi}{\partial \boldsymbol{w}_k^{12}} = (y_k - t_k)\frac{\partial f(z_k^{(2)})}{\partial z_k^{(2)}}\boldsymbol{x}^{(1)^T}$$

学習データごとに上記勾配を計算し、それらを足し合わせて
パラメータ $\boldsymbol{w}_k^{12}$ の変更量 $\Delta\boldsymbol{w}_k^{12}$ を計算する。

$$\Delta\boldsymbol{w}_k^{12} = \sum_{\boldsymbol{i}}\left(\frac{\partial \varphi_i}{\partial \boldsymbol{w}_k^{12}}\right)^{\boldsymbol{T}}$$

$$\boldsymbol{w}_k^{12} \leftarrow \boldsymbol{w}_k^{12} - \alpha\Delta\boldsymbol{w}_k^{12} \qquad (\alpha > 0)$$

単純パーセプトロンの学習

※３　入力層のバイアスパラメータ　$\boldsymbol{h}^{(1)}$　について

$$\frac{\partial \varphi}{\partial \boldsymbol{h}^{(1)}} = \boxed{\frac{\partial \varphi}{\partial \boldsymbol{e}}}\ \boxed{\frac{\partial \boldsymbol{e}}{\partial \boldsymbol{y}}}\ \boxed{\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{z}^{(2)}}}\ \boxed{\frac{\partial \boldsymbol{z}^{(2)}}{\partial \boldsymbol{x}^{(1)}}}\ \boxed{\frac{\partial \boldsymbol{x}^{(1)}}{\partial \boldsymbol{z}^{(1)}}}\ \boxed{\frac{\partial \boldsymbol{z}^{(1)}}{\partial \boldsymbol{h}^{(1)}}}$$

(1)　(2)　(3)　(4)　(5)　(6)

(1)～(3)は既に計算済み

(4)　$\boldsymbol{W^{12}}$

(5)
$$\begin{bmatrix} \partial f(z_1^{(1)})/\partial z_1^{(1)} & & \boldsymbol{O} \\ & \partial f(z_2^{(1)})/\partial z_2^{(1)} & \\ & & \ddots \\ \boldsymbol{O} & & \partial f(z_m^{(1)})/\partial z_m^{(1)} \end{bmatrix}$$

(6)　$-I$

単純パーセプトロンの学習

※３　(1)～(6)の項を掛け合わせると勾配 $\frac{\partial \varphi}{\partial \boldsymbol{h}^{(1)}}$ を計算できる。

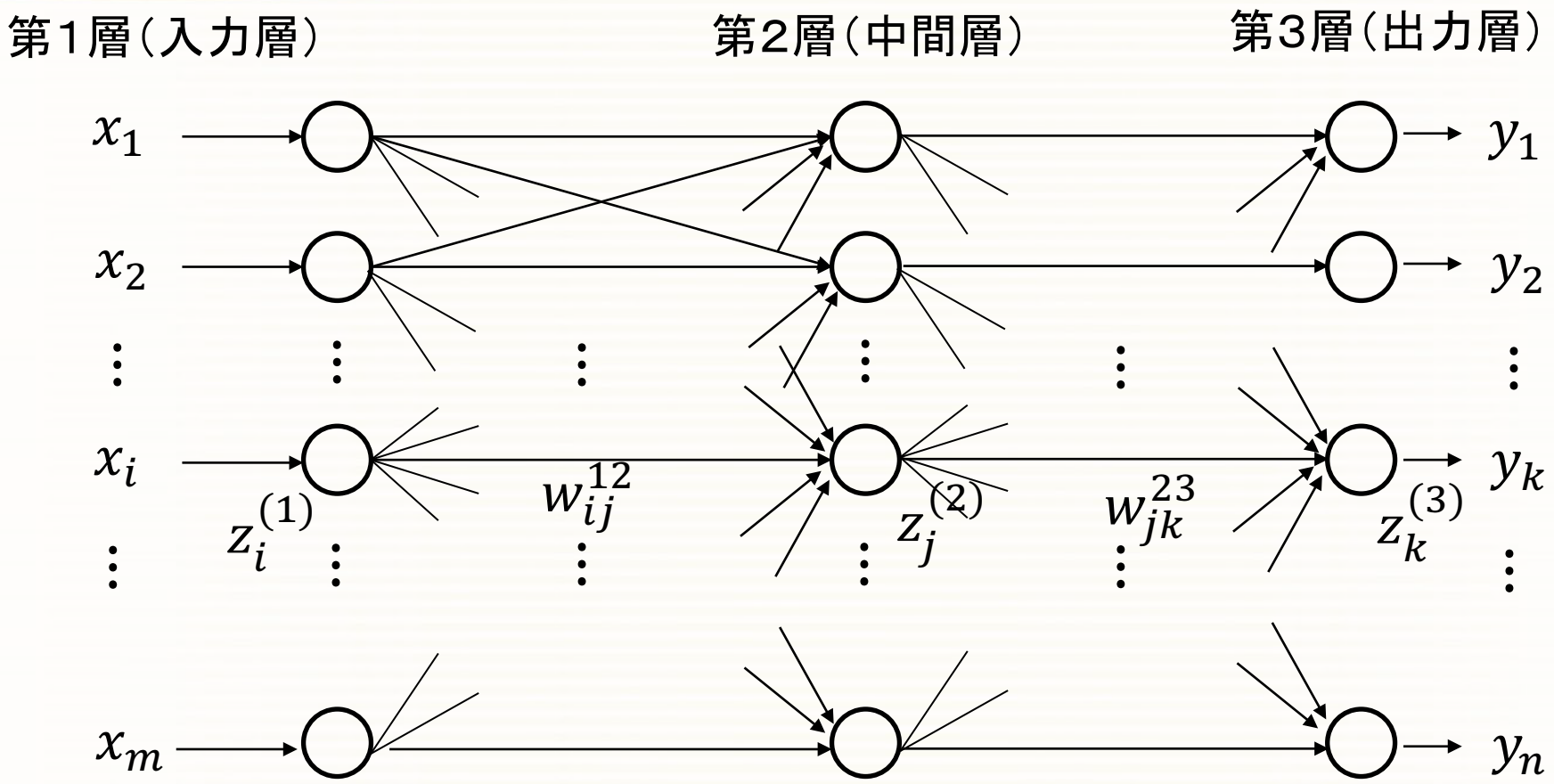
学習データごとに上記勾配を足し合わせて
パラメータ $\boldsymbol{h}^{(1)}$ の変更量 $\Delta \boldsymbol{h}^{(1)}$ を計算する。

$$\Delta \boldsymbol{h}^{(1)} = \sum_{\boldsymbol{i}} \left(\frac{\partial \varphi_i}{\partial \boldsymbol{h}^{(1)}} \right)^{\boldsymbol{T}}$$

$$\boldsymbol{h}^{(1)} \leftarrow \boldsymbol{h}^{(1)} - \alpha \, \Delta \boldsymbol{h}^{(1)} \qquad (\alpha > 0)$$

以上のように※１、※２、※３と出力層側のパラメータから修正する。
このような後ろ向きの手順に従って、パラメータを更新する方法を
誤差逆伝搬法（バックプロパゲーション）と呼ぶ。

3層ニューラルネットワークの学習



単純パーセプトロンの場合と同様に、ニューラルネットワークのパラメータを最適化する

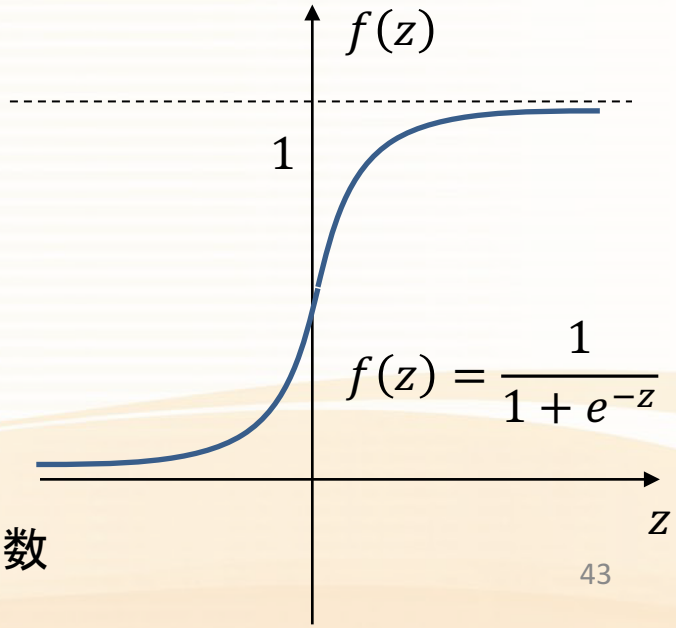
3層ニューラルネットワークの学習

第1層（入力層）における入力・内部状態・出力について
（単純パーセプトロンの場合と同じ）

入力 $\boldsymbol{x}^{(0)} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}$ ➡ 内部状態 $\boldsymbol{z}^{(1)} = \begin{bmatrix} x_1 - h_1 \\ x_2 - h_2 \\ \vdots \\ x_m - h_m \end{bmatrix}$

h：バイアスパラメータ

➡ 出力 $\boldsymbol{x}^{(1)} = \begin{bmatrix} f(x_1 - h_1) \\ f(x_2 - h_2) \\ \vdots \\ f(x_m - h_m) \end{bmatrix}$



$f(z)$: 活性化関数であり、
右のようなシグモイド関数
を使用する

3層ニューラルネットワークの学習

第2層（中間層）における入力・内部状態・出力について
（単純パーセプトロンの場合と同じ）

入力

$$\begin{bmatrix} \tilde{x}_1^{(2)} \\ \tilde{x}_2^{(2)} \\ \vdots \\ \tilde{x}_l^{(2)} \end{bmatrix} = \begin{bmatrix} w_{11}^{12}\, x_1^{(1)} + w_{21}^{12}\, x_2^{(1)} + \cdots + w_{m1}^{12}\, x_m^{(1)} \\ w_{12}^{12}\, x_1^{(1)} + w_{22}^{12}\, x_2^{(1)} + \cdots + w_{m2}^{12}\, x_m^{(1)} \\ \vdots \\ w_{1l}^{12}\, x_1^{(1)} + w_{2l}^{12}\, x_2^{(1)} + \cdots + w_{ml}^{12}\, x_m^{(1)} \end{bmatrix}$$

$$= \begin{bmatrix} w_{11}^{12} & w_{21}^{12} & \cdots & w_{m1}^{12} \\ w_{12}^{12} & w_{22}^{12} & \cdots & w_{m2}^{12} \\ & \vdots & & \\ w_{1l}^{12} & w_{2l}^{12} & \cdots & w_{ml}^{12} \end{bmatrix} \begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \\ \vdots \\ x_m^{(1)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{w}_1^{12\,T} \\ \boldsymbol{w}_2^{12\,T} \\ \vdots \\ \boldsymbol{w}_l^{12\,T} \end{bmatrix} \boldsymbol{x}^{(1)}$$

$$= \boldsymbol{W}^{\boldsymbol{12}}\, \boldsymbol{x}^{(1)}$$

3層ニューラルネットワークの学習

第2層（中間層）における入力・内部状態・出力について
（単純パーセプトロンの場合と同じ）

内部状態 $\boldsymbol{z}^{(2)} = \begin{bmatrix} z_1^{(2)} \\ z_2^{(2)} \\ \vdots \\ z_l^{(2)} \end{bmatrix} = \begin{bmatrix} \tilde{x}_1^{(2)} - h_1^{(2)} \\ \tilde{x}_2^{(2)} - h_2^{(2)} \\ \vdots \\ \tilde{x}_l^{(2)} - h_l^{(2)} \end{bmatrix} = \boldsymbol{W^{12}}\, \boldsymbol{x}^{(1)} - \boldsymbol{h}^{(2)}$

出力 $\boldsymbol{x}^{(2)} = \begin{bmatrix} f(z_1^{(2)}) \\ f(z_2^{(2)}) \\ \vdots \\ f(z_l^{(2)}) \end{bmatrix}$

3層ニューラルネットワークの学習

第3層（出力層）における入力・内部状態・出力について
（単純パーセプトロンの場合と同じ）

入力

$$\begin{bmatrix} \tilde{x}_1^{(3)} \\ \tilde{x}_2^{(3)} \\ \vdots \\ \tilde{x}_n^{(3)} \end{bmatrix} = \begin{bmatrix} w_{11}^{23}\, x_1^{(2)} + w_{21}^{23}\, x_2^{(2)} + \cdots + w_{l1}^{23}\, x_l^{(2)} \\ w_{12}^{23}\, x_1^{(2)} + w_{22}^{23}\, x_2^{(2)} + \cdots + w_{l2}^{23}\, x_l^{(2)} \\ \vdots \\ w_{1n}^{23}\, x_1^{(2)} + w_{2n}^{23}\, x_2^{(2)} + \cdots + w_{ln}^{23}\, x_l^{(2)} \end{bmatrix}$$

$$= \begin{bmatrix} w_{11}^{23} & w_{21}^{23} & \cdots & w_{l1}^{23} \\ w_{12}^{23} & w_{22}^{23} & \cdots & w_{l2}^{23} \\ & \vdots & & \\ w_{1n}^{23} & w_{2n}^{23} & \cdots & w_{ln}^{23} \end{bmatrix} \begin{bmatrix} x_1^{(2)} \\ x_2^{(2)} \\ \vdots \\ x_l^{(2)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{w}_1^{23\,T} \\ \boldsymbol{w}_2^{23\,T} \\ \vdots \\ \boldsymbol{w}_n^{23\,T} \end{bmatrix} \boldsymbol{x}^{(2)}$$

$$= \boldsymbol{W^{23}}\, \boldsymbol{x}^{(2)}$$

3層ニューラルネットワークの学習

第3層（出力層）における入力・内部状態・出力について
（単純パーセプトロンの場合と同じ）

内部状態 $\boldsymbol{z}^{(3)} = \begin{bmatrix} z_1^{(3)} \\ z_2^{(3)} \\ \vdots \\ z_n^{(3)} \end{bmatrix} = \begin{bmatrix} \tilde{x}_1^{(3)} - h_1^{(3)} \\ \tilde{x}_2^{(3)} - h_2^{(3)} \\ \vdots \\ \tilde{x}_n^{(3)} - h_n^{(3)} \end{bmatrix} = \boldsymbol{W^{23}}\ \boldsymbol{x}^{(2)} - \boldsymbol{h}^{(3)}$

出力 $\boldsymbol{x}^{(3)} = \begin{bmatrix} f(z_1^{(3)}) \\ f(z_2^{(3)}) \\ \vdots \\ f(z_n^{(3)}) \end{bmatrix} = \boldsymbol{y}$

3層ニューラルネットワークの学習

学習データ：入力 $\boldsymbol{u}$ → 出力 $\boldsymbol{t}$ となるようにパラメータを調整する
（単純パーセプトロンと同じ）

学習データの出力 $\boldsymbol{t}$ とニューラルネットワークの出力 $\boldsymbol{y}$ の誤差ベクトル

$$\boldsymbol{e} = \boldsymbol{y} - \boldsymbol{t}$$

評価関数を誤差の大きさとする

$$\boldsymbol{\varphi} = \frac{1}{2}\,\boldsymbol{e}^T\,\boldsymbol{e}$$

評価関数が小さくなるように最急降下法を用いてパラメータを修正する

3層ニューラルネットワークの学習

※1　出力層のバイアスパラメータ　$\boldsymbol{h}^{(3)}$　について

$$\frac{\partial \varphi}{\partial \boldsymbol{h}^{(3)}} = \boxed{\frac{\partial \varphi}{\partial \boldsymbol{e}}} \boxed{\frac{\partial \boldsymbol{e}}{\partial \boldsymbol{y}}} \boxed{\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{z}^{(3)}}} \boxed{\frac{\partial \boldsymbol{z}^{(3)}}{\partial \boldsymbol{h}^{(3)}}}$$

(1)　(2)　(3)　(4)

(1)　$\boldsymbol{e}^T$

(2)　I　(単位行列)

(3)
$$\begin{bmatrix} \partial f(z_1^{(3)})/\partial z_1^{(3)} & & \boldsymbol{O} \\ & \partial f(z_2^{(3)})/\partial z_2^{(3)} & \\ & & \ddots \\ \boldsymbol{O} & & \partial f(z_n^{(3)})/\partial z_n^{(3)} \end{bmatrix}$$

シグモイド関数の微分

$$\frac{\partial f(z)}{\partial z} = \frac{e^{-z}}{(1+e^{-z})^2}$$

(4)　$-I$

3層ニューラルネットワークの学習

※1　ちなみに、(1)～(4)の項を掛け合わせると

$$\frac{\partial \varphi}{\partial \boldsymbol{h}^{(3)}} = \left[-(y_1 - t_1)\frac{\partial f(z_1^{(3)})}{\partial z_1^{(3)}} \quad \cdots \quad -(y_n - t_n)\frac{\partial f(z_n^{(3)})}{\partial z_n^{(3)}} \right]$$

学習データごとに上記勾配を計算し、それらを足し合わせて
パラメータ　$\boldsymbol{h}^{(3)}$ の変更量 $\Delta \boldsymbol{h}^{(3)}$を計算する。

$$\Delta \boldsymbol{h}^{(3)} = \sum_{\boldsymbol{i}} \left(\frac{\partial \varphi_i}{\partial \boldsymbol{h}^{(3)}} \right)^{\boldsymbol{T}}$$

$$\boldsymbol{h}^{(3)} \leftarrow \boldsymbol{h}^{(3)} - \alpha\, \Delta \boldsymbol{h}^{(3)} \qquad (\alpha > 0)$$

3層ニューラルネットワークの学習

※２ シナプス重み結合のパラメータ $\boldsymbol{w}_k^{23}$ について

$$\frac{\partial \varphi}{\partial \boldsymbol{w}_k^{23}} = \boxed{\frac{\partial \varphi}{\partial \boldsymbol{e}}} \boxed{\frac{\partial \boldsymbol{e}}{\partial \boldsymbol{y}}} \boxed{\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{z}^{(3)}}} \boxed{\frac{\partial \boldsymbol{z}^{(3)}}{\partial \boldsymbol{w}_k^{23}}}$$

(1) (2) (3) (4)

(1)～(3)は既に計算済み

(4) $\begin{bmatrix} \mathbf{0} \\ \boldsymbol{x}^{(2)^T} \\ \mathbf{0} \end{bmatrix} < k$ (k行目以外はすべてゼロ)

3層ニューラルネットワークの学習

※２　ちなみに、(1)～(4)の項を掛け合わせると

$$\frac{\partial \varphi}{\partial \boldsymbol{w}_k^{23}} = (y_k - t_k)\frac{\partial f(z_k^{(3)})}{\partial z_k^{(3)}}\boldsymbol{x}^{(2)^T}$$

学習データごとに上記勾配を計算し、それらを足し合わせて
パラメータ $\boldsymbol{w}_k^{23}$ の変更量 $\Delta\boldsymbol{w}_k^{23}$ を計算する。

$$\Delta\boldsymbol{w}_k^{23} = \sum_{\boldsymbol{i}}\left(\frac{\partial \varphi_i}{\partial \boldsymbol{w}_k^{23}}\right)^{\boldsymbol{T}}$$

$$\boldsymbol{w}_k^{23} \leftarrow \boldsymbol{w}_k^{23} - \alpha\Delta\boldsymbol{w}_k^{23} \qquad (\alpha > 0)$$

3層ニューラルネットワークの学習

※３　中間層のバイアスパラメータ　$\boldsymbol{h}^{(2)}$　について

$$\frac{\partial \varphi}{\partial \boldsymbol{h}^{(2)}} = \boxed{\frac{\partial \varphi}{\partial \boldsymbol{e}}}\ \boxed{\frac{\partial \boldsymbol{e}}{\partial \boldsymbol{y}}}\ \boxed{\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{z}^{(3)}}}\ \boxed{\frac{\partial \boldsymbol{z}^{(3)}}{\partial \boldsymbol{x}^{(2)}}}\ \boxed{\frac{\partial \boldsymbol{x}^{(2)}}{\partial \boldsymbol{z}^{(2)}}}\ \boxed{\frac{\partial \boldsymbol{z}^{(2)}}{\partial \boldsymbol{h}^{(2)}}}$$

(1)　(2)　(3)　(4)　(5)　(6)

(1)～(3)は既に計算済み

(4)　$\boldsymbol{W^{23}}$

(5)
$$\begin{bmatrix} \partial f(z_1^{(2)})/\partial z_1^{(2)} & & \boldsymbol{O} \\ & \partial f(z_2^{(2)})/\partial z_2^{(2)} & \\ & & \ddots \\ \boldsymbol{O} & & \partial f(z_l^{(2)})/\partial z_l^{(2)} \end{bmatrix}$$

(6)　$-I$

3層ニューラルネットワークの学習

※３　(1)～(6)の項を掛け合わせると勾配 $\frac{\partial \varphi}{\partial \boldsymbol{h}^{(2)}}$ を計算できる。

学習データごとに上記勾配を足し合わせて
パラメータ $\boldsymbol{h}^{(2)}$ の変更量 $\Delta \boldsymbol{h}^{(2)}$ を計算する。

$$\Delta \boldsymbol{h}^{(2)} = \sum_{\boldsymbol{i}} \left(\frac{\partial \varphi_i}{\partial \boldsymbol{h}^{(2)}} \right)^{\boldsymbol{T}}$$

$$\boldsymbol{h}^{(2)} \leftarrow \boldsymbol{h}^{(2)} - \alpha \, \Delta \boldsymbol{h}^{(2)} \qquad (\alpha > 0)$$

3層ニューラルネットワークの学習

※4　シナプス重み結合のパラメータ $\boldsymbol{w}_k^{12}$　について

$$\frac{\partial \varphi}{\partial \boldsymbol{w}_k^{12}} = \underbrace{\frac{\partial \varphi}{\partial \boldsymbol{e}}}_{(1)} \underbrace{\frac{\partial \boldsymbol{e}}{\partial \boldsymbol{y}}}_{(2)} \underbrace{\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{z}^{(3)}}}_{(3)} \underbrace{\frac{\partial \boldsymbol{z}^{(3)}}{\partial \boldsymbol{x}^{(2)}}}_{(4)} \underbrace{\frac{\partial \boldsymbol{x}^{(2)}}{\partial \boldsymbol{z}^{(2)}}}_{(5)} \underbrace{\frac{\partial \boldsymbol{z}^{(2)}}{\partial \boldsymbol{w}_k^{12}}}_{(6)}$$

(1)～(5)は既に計算済み

(6)　$\begin{bmatrix} \mathbf{0} \\ \boldsymbol{x}^{(1)^T} \\ \mathbf{0} \end{bmatrix} < k$　　(k行目以外はすべてゼロ)

3層ニューラルネットワークの学習

※4　(1)～(6)の項を掛け合わせると勾配 $\dfrac{\partial \varphi}{\partial \boldsymbol{w}_k^{12}}$ を計算できる。

学習データごとに上記勾配を足し合わせて
パラメータ $\boldsymbol{w}_k^{12}$ の変更量 $\Delta \boldsymbol{w}_k^{12}$ を計算する。

$$\Delta \boldsymbol{w}_k^{12} = \sum_{\boldsymbol{i}} \left(\frac{\partial \varphi_i}{\partial \boldsymbol{w}_k^{12}} \right)^{\boldsymbol{T}}$$

$$\boldsymbol{w}_k^{12} \leftarrow \boldsymbol{w}_k^{12} - \alpha \, \Delta \boldsymbol{w}_k^{12} \qquad (\alpha > 0)$$

3層ニューラルネットワークの学習

※５　入力層のバイアスパラメータ $\boldsymbol{h}^{(1)}$ について

$$\frac{\partial \varphi}{\partial \boldsymbol{h}^{(1)}} = \frac{\partial \varphi}{\partial \boldsymbol{e}} \frac{\partial \boldsymbol{e}}{\partial \boldsymbol{y}} \frac{\partial \boldsymbol{y}}{\partial \boldsymbol{z}^{(3)}} \frac{\partial \boldsymbol{z}^{(3)}}{\partial \boldsymbol{x}^{(2)}} \frac{\partial \boldsymbol{x}^{(2)}}{\partial \boldsymbol{z}^{(2)}} \frac{\partial \boldsymbol{z}^{(2)}}{\partial \boldsymbol{x}^{(1)}} \frac{\partial \boldsymbol{x}^{(1)}}{\partial \boldsymbol{z}^{(1)}} \frac{\partial \boldsymbol{z}^{(1)}}{\partial \boldsymbol{h}^{(1)}}$$

(1) (2) (3) (4) (5) (6) (7) (8)

(1)～(5)は既に計算済み

(6)　$\boldsymbol{W^{12}}$

(7)
$$\begin{bmatrix} \partial f(z_1^{(1)})/\partial z_1^{(1)} & & \boldsymbol{O} \\ & \partial f(z_2^{(1)})/\partial z_2^{(1)} & \\ & & \ddots \\ \boldsymbol{O} & & \partial f(z_l^{(1)})/\partial z_m^{(1)} \end{bmatrix}$$

(8)　$-I$

3層ニューラルネットワークの学習

※５　(1)～(8)の項を掛け合わせると勾配 $\frac{\partial \varphi}{\partial \boldsymbol{h}^{(1)}}$ を計算できる。

学習データごとに上記勾配を足し合わせて
パラメータ $\boldsymbol{h}^{(1)}$ の変更量 $\Delta \boldsymbol{h}^{(1)}$ を計算する。

$$\Delta \boldsymbol{h}^{(1)} = \sum_{\boldsymbol{i}} \left(\frac{\partial \varphi_i}{\partial \boldsymbol{h}^{(1)}} \right)^{\boldsymbol{T}}$$

$$\boldsymbol{h}^{(1)} \leftarrow \boldsymbol{h}^{(1)} - \alpha \, \Delta \boldsymbol{h}^{(1)} \qquad (\alpha > 0)$$

単純パーセプトロンの場合と同様に、※１、※２、※３、※４、※５と
出力層側のパラメータから修正する。このような後ろ向きの手順に従って、パラメータを更新・学習することができる

データサイエンス応用コース
（データ生成過程のモデル化）

高野　渉
大阪大学

パーティクルフィルタ

データやそのデータを生み出す状態のメカニズムを数学モデルとして規定したうえで，観測されるデータから状態を推定する.

パーティクルフィルタは，多くの状態をサンプリングして，観測値に基づいて状態を評価する手順を繰り返しながら，状態を推定する方法.

１．状態の分布から複数の状態（粒子）を生成
２．観測値から各状態を評価
３．評価に基づいて状態の分布を修正
４．1に戻る

パーティクルフィルタ

データやそのデータを生み出す状態のメカニズムを数学モデルとして規定したうえで，観測されるデータから状態を推定する.

時刻　k

状態　$\boldsymbol{x}_k \in R^n$　　観測値　$\boldsymbol{y}_k \in R^p$

平均0白色雑音　$\boldsymbol{w}_k \in R^m, \boldsymbol{v}_k \in R^r$

状態方程式（状態の移り変わり・ダイナミクスを表現）

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}_k(\boldsymbol{x}_k, \boldsymbol{w}_k)$$

観測方程式（状態と観測の関係を表現）

$$\boldsymbol{y}_k = \boldsymbol{h}_k(\boldsymbol{x}_k, \boldsymbol{v}_k)$$

パーティクルフィルタ

観測値の集合 $D_k = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_k\}$
初期状態確率 $p(\boldsymbol{x}_1|\boldsymbol{y}_1) = p(\boldsymbol{x}_1)$
状態方程式 $\boldsymbol{f}_k(\boldsymbol{x}_k, \boldsymbol{w}_k)$
観測方程式 $\boldsymbol{h}_k(\boldsymbol{x}_k, \boldsymbol{v}_k)$
は与えられているものとする.

仮に, $D_{k-1} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_{k-1}\}$の条件のもとで, $p(\boldsymbol{x}_{k-1}|\boldsymbol{D}_{k-1})$がわかっているとすると, 確率のチェーンルールと周辺化より, $p(\boldsymbol{x}_k|\boldsymbol{D}_{k-1})$は以下の通りに計算することができる.

$$p(\boldsymbol{x}_k|D_{k-1}) = \int p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) p(\boldsymbol{x}_{k-1}|D_{k-1})\, dx_{k-1} \quad \cdots ①$$

パーティクルフィルタ

観測値の集合 $D_k = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_k\}$
初期状態確率 $p(\boldsymbol{x}_1|\boldsymbol{y}_1) = p(\boldsymbol{x}_1)$
状態方程式 $\boldsymbol{f}_k(\boldsymbol{x}_k, \boldsymbol{w}_k)$
観測方程式 $\boldsymbol{h}_k(\boldsymbol{x}_k, \boldsymbol{v}_k)$
は与えられているものとする.

仮に, $D_{k-1} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_{k-1}\}$の条件のもとで, $p(\boldsymbol{x}_{k-1}|\boldsymbol{D}_{k-1})$がわかっているとすると, 確率のチェーンルールと周辺化より, $p(\boldsymbol{x}_k|\boldsymbol{D}_{k-1})$は以下の通りに計算することができる.

$$p(\boldsymbol{x}_k|D_{k-1}) = \int p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) p(\boldsymbol{x}_{k-1}|D_{k-1})\, dx_{k-1} \quad \cdots ①$$

パーティクルフィルタ

$p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$ は状態方程式に依存しており，

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) = \int p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{w}_{k-1})\, p(\boldsymbol{w}_{k-1}|\boldsymbol{x}_{k-1}) d\boldsymbol{w}_{k-1}$$

となる．雑音 $\boldsymbol{w}_{k-1}$ と状態$\boldsymbol{x}_{k-1}$ は独立であると仮定する．

$$p(\boldsymbol{w}_{k-1}|\boldsymbol{x}_{k-1}) = p(\boldsymbol{w}_{k-1})$$

$p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{w}_{k-1})$について，状態$\boldsymbol{x}_{k-1}$と雑音 $\boldsymbol{w}_{k-1}$から状態方程式によって決まる状態$\boldsymbol{x}_k$のみが起こりうる事象とみると，上式は以下のように書き換えられる．

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) = \int \delta\big(\boldsymbol{x}_k - \boldsymbol{f}_{k-1}(\boldsymbol{x}_{k-1}, \boldsymbol{w}_{k-1})\big)\, p(\boldsymbol{w}_{k-1}) d\boldsymbol{w}_{k-1} \quad \cdots ②$$

δ はディラックデルタ関数である．

パーティクルフィルタ

多くの $(\boldsymbol{x}_{k-1}, \boldsymbol{w}_{k-1})$を生成して，数値的に$p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$を計算することができる（サンプリング）.
$p(\boldsymbol{x}_{k-1}|\boldsymbol{D}_{k-1})$がわかっているものとするので，式①より$p(\boldsymbol{x}_k|D_{k-1})$が計算できることになる.

時刻$k-1$までの観測データから時刻kの状態の分布を推定する処理なので，これは状態を予測することに相当する

パーティクルフィルタ

次に，時刻kの観測値 $\boldsymbol{y}_k$ が与えられたときの状態の分布 $p(\boldsymbol{x}_k|D_k)$ を推定することを考える.

$$p(\boldsymbol{x}_k|D_k) = \frac{p(\boldsymbol{x}_k, \boldsymbol{y}_k|D_{k-1})}{p(\boldsymbol{y}_k\ |D_{k-1})}$$

ベイズルールより

$$= \frac{p(\boldsymbol{y}_k|\boldsymbol{x}_k, D_{k-1})p(\boldsymbol{x}_k|D_{k-1})}{p(\boldsymbol{y}_k\ |D_{k-1})}$$

チェーンルールより

$$= \frac{p(\boldsymbol{y}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|D_{k-1})}{p(\boldsymbol{y}_k\ |D_{k-1})} \quad \cdots ③$$

観測方程式から$\boldsymbol{y}_k$は $\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_{k-1}$に依存しない

式③において，$p(\boldsymbol{x}_k|D_{k-1})$ は式①によって計算済みである.

パーティクルフィルタ

次に，時刻kの観測値 $\boldsymbol{y}_k$ が与えられたときの状態の分布 $p(\boldsymbol{x}_k|D_k)$ を推定することを考える.

$$p(\boldsymbol{x}_k|D_k) = \frac{p(\boldsymbol{x}_k, \boldsymbol{y}_k|D_{k-1})}{p(\boldsymbol{y}_k\ |D_{k-1})}$$

ベイズルールより

$$= \frac{p(\boldsymbol{y}_k|\boldsymbol{x}_k, D_{k-1})p(\boldsymbol{x}_k|D_{k-1})}{p(\boldsymbol{y}_k\ |D_{k-1})}$$

チェーンルールより

$$= \frac{p(\boldsymbol{y}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|D_{k-1})}{p(\boldsymbol{y}_k\ |D_{k-1})} \quad \cdots ③$$

観測方程式から$\boldsymbol{y}_k$は $\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_{k-1}$に依存しない

式③において，$p(\boldsymbol{x}_k|D_{k-1})$ は式①によって計算済みである.

パーティクルフィルタ

式③の $p(\boldsymbol{y}_k|\boldsymbol{x}_k)$ に関して,

$$p(\boldsymbol{y}_k|\boldsymbol{x}_k) = \int p(\boldsymbol{y}_k|\boldsymbol{x}_k, \boldsymbol{v}_k)p(\boldsymbol{v}_k|\boldsymbol{x}_k)d\boldsymbol{v}_k \qquad \text{周辺化より}$$

雑音 $\boldsymbol{v}_k$ と状態$\boldsymbol{x}_k$ は独立であると仮定し,

状態$\boldsymbol{x}_k$と雑音 $\boldsymbol{v}_k$から観測方程式によって決まる状態$\boldsymbol{y}_k$のみが起こりうる事象とみると, 上式は以下のように書き換えられる.

$$p(\boldsymbol{y}_k|\boldsymbol{x}_k) = \int \delta\big(\boldsymbol{y}_k - h_k(\boldsymbol{x}_k, \boldsymbol{v}_k)\big)p(\boldsymbol{v}_k)d\boldsymbol{v}_k \qquad \cdots④$$

これもサンプリング手法を用いて数値的に求めることができる.

パーティクルフィルタ

式③の$p(\boldsymbol{y}_k\,|D_{k-1})$に関して，

$$p(\boldsymbol{y}_k\,|D_{k-1}) = \int p(\boldsymbol{y}_k|\boldsymbol{x}_k)\,p(\boldsymbol{x}_k|D_{k-1})d\boldsymbol{x}_k \quad \cdots⑤ \quad \text{周辺化より}$$

$p(\boldsymbol{y}_k|\boldsymbol{x}_k)$は式④により計算済み，$p(\boldsymbol{x}_k|D_{k-1})$は式①より計算済みであるため，$p(\boldsymbol{y}_k\,|D_{k-1})$も計算することができる.

したがって，式③ の $p(\boldsymbol{x}_k|D_k)$は式①④⑤を用いて計算できることになる.

時刻kの観測データを用いて時刻kの状態の分布を予測から修正する処理であり，最新の観測データを使って状態を更新することに相当する.

パーティクルフィルタ

時刻$k-1$までの観測値から時刻kの状態の分布

$$p(\boldsymbol{x}_k \mid D_{k-1})$$

を計算する予測 (Prediction)と

時刻 k の観測値を用いて時刻kの状態の分布

$$p(\boldsymbol{x}_k \mid D_k)$$

を計算する更新 (Update)

とを繰り返しながら状態の分布を逐次的に推定する.

パーティクルフィルタ

パーティクルフィルタをどのように実装するのか？

Step1 状態の分布$p(\boldsymbol{x}_{k-1}|D_{k-1})$に基づいて，
$\left\{\boldsymbol{x}_{k-1}{}^{(1)}, \boldsymbol{x}_{k-1}{}^{(2)}, \cdots, \boldsymbol{x}_{k-1}{}^{(N)}\right\}$ をサンプリングして生成する．

Step2 各状態$\boldsymbol{x}_{k-1}{}^{(i)}$を状態方程式に代入して，
$\left\{\boldsymbol{x}_{k}{}^{(1)}, \boldsymbol{x}_{k}{}^{(2)}, \cdots, \boldsymbol{x}_{k}{}^{(N)}\right\}$ を計算する

Step3 各状態$\boldsymbol{x}_{k}{}^{(i)}$を観測値$\boldsymbol{y}_k$を用いて，
評価値 $p\left(\boldsymbol{y}_k|\boldsymbol{x}_{k}{}^{(i)}\right)$ を計算する

Step4 評価値（尤もらしさ） $p\left(\boldsymbol{y}_k|\boldsymbol{x}_{k}{}^{(i)}\right)$ を用いて，各状態$\boldsymbol{x}_{k}{}^{(i)}$の分布

$$q^{(i)} = p\left(\boldsymbol{y}_k|\boldsymbol{x}_{k}{}^{(i)}\right) \Big/ \sum_{i=1}^{N} p\left(\boldsymbol{y}_k|\boldsymbol{x}_{k}{}^{(i)}\right)$$

を計算する．これを分布$p(\boldsymbol{x}_k|D_k)$としてstep1に戻る．

カルマンフィルタ

パーティクルフィルタの特別な場合として，状態方程式，観測方程式が線形方程式で記述され，状態と観測のノイズが正規分布に従うと仮定する.

状態　$\boldsymbol{x}_k \in R^n$　　観測値　$\boldsymbol{y}_k \in R^p$

$\boldsymbol{w}_k \sim N(\boldsymbol{0}, \boldsymbol{\Sigma}_w) \in R^n, \boldsymbol{v}_k \sim N(\boldsymbol{0}, \boldsymbol{\Sigma}_v) \in R^r$

状態方程式

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k \boldsymbol{x}_k + \boldsymbol{w}_k$$

観測方程式（状態と観測の関係を表現）

$$\boldsymbol{y}_k = \boldsymbol{C}_k \boldsymbol{x}_k + \boldsymbol{v}_k$$

カルマンフィルタ

観測データ $D_{k-1} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_{k-1}\}$ のもとで状態$\boldsymbol{x}_k$の分布は以下のように与えられる（パーティクルフィルタの復習）

$$p(\boldsymbol{x}_k|D_{k-1}) = \int p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) p(\boldsymbol{x}_{k-1}|D_{k-1})\, dx_{k-1}$$

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) = \int p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{w}_{k-1})\, p(\boldsymbol{w}_{k-1}|\boldsymbol{x}_{k-1}) d\boldsymbol{w}_{k-1}$$

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) = \int \delta(\boldsymbol{x}_k - \boldsymbol{A}_k\, \boldsymbol{x}_k - \boldsymbol{w}_k)\, p(\boldsymbol{w}_{k-1}) d\boldsymbol{w}_{k-1}$$

ノイズ$\boldsymbol{w}_k$は平均ゼロ，分散・共分散行列$\boldsymbol{\Sigma}_w$の正規分布に従うことから

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) = \frac{1}{\sqrt{(2\pi)^n |\boldsymbol{\Sigma}_w|}} \times \exp\left\{-\frac{1}{2}(\boldsymbol{x}_k - \boldsymbol{A}_{k-1}\, \boldsymbol{x}_{k-1} - \boldsymbol{w}_{k-1})^T \boldsymbol{\Sigma}_w{}^{-1} (\boldsymbol{x}_k - \boldsymbol{A}_{k-1}\, \boldsymbol{x}_{k-1} - \boldsymbol{w}_{k-1})\right\}$$

カルマンフィルタ

$p(\boldsymbol{x}_{k-1}|D_{k-1})$ が平均ベクトル$\boldsymbol{\mu}_{k-1|k-1}$, 分散共分散行列$\boldsymbol{\Sigma}_{k-1|k-1}$ の正規分布であるとすると,

$$p(\boldsymbol{x}_k|D_{k-1}) = \int \left[\frac{1}{\sqrt{(2\pi)^n|\boldsymbol{\Sigma}_w|}} \times \frac{1}{\sqrt{(2\pi)^n\left|\boldsymbol{\Sigma}_{k-1|k-1}\right|}} \right.$$
$$\times \exp\left\{-\frac{1}{2}(\boldsymbol{x}_k - \boldsymbol{A}_{k-1}\,\boldsymbol{x}_{k-1} - \boldsymbol{w}_k)^T {\boldsymbol{\Sigma}_w}^{-1}(\boldsymbol{x}_k - \boldsymbol{A}_{k-1}\,\boldsymbol{x}_{k-1} - \boldsymbol{w}_k)\right\}$$
$$\left. \times \exp\left\{-\frac{1}{2}\left(\boldsymbol{x}_{k-1} - \boldsymbol{\mu}_{k-1|k-1}\right)^T {\boldsymbol{\Sigma}_{k-1|k-1}}^{-1}\left(\boldsymbol{x}_{k-1} - \boldsymbol{\mu}_{k-1|k-1}\right)\right\}\right] d\boldsymbol{x}_{k-1}$$

exp の指数部は以下となる

$$-\frac{1}{2}\left[{\boldsymbol{x}_k}^T {\boldsymbol{\Sigma}_w}^{-1}\boldsymbol{x}_k - 2{\boldsymbol{x}_{k-1}}^T {\boldsymbol{A}_{k-1}}^T {\boldsymbol{\Sigma}_w}^{-1}\boldsymbol{x}_k + {\boldsymbol{x}_{k-1}}^T {\boldsymbol{A}_{k-1}}^T {\boldsymbol{\Sigma}_w}^{-1}\boldsymbol{A}_{k-1}\,\boldsymbol{x}_{k-1}\right.$$
$$+ {\boldsymbol{x}_{k-1}}^T {\boldsymbol{\Sigma}_{k-1|k-1}}^{-1}\boldsymbol{x}_{k-1} - 2{\boldsymbol{x}_{k-1}}^T {\boldsymbol{\Sigma}_{k-1|k-1}}^{-1}\boldsymbol{\mu}_{k-1|k-1}$$
$$\left. + {\boldsymbol{\mu}_{k-1|k-1}}^T {\boldsymbol{\Sigma}_{k-1|k-1}}^{-1}\boldsymbol{\mu}_{k-1|k-1}\right]$$

カルマンフィルタ

$\boldsymbol{x}_{k-1}$に関して以下のように整理すると,

$$-\frac{1}{2}\left[(\boldsymbol{x}_{k-1}-\boldsymbol{\alpha})^T\left({\boldsymbol{A}_{k-1}}^T{\boldsymbol{\Sigma}_w}^{-1}\boldsymbol{A}_{k-1}+{\boldsymbol{\Sigma}_{k-1|k-1}}^{-1}\right)(\boldsymbol{x}_{k-1}-\boldsymbol{\alpha})+\beta\right]$$

$\boldsymbol{x}_{k-1}$の1次の項を比較することによって$\boldsymbol{\alpha}$が求まる.

$$\left({\boldsymbol{A}_{k-1}}^T{\boldsymbol{\Sigma}_w}^{-1}\boldsymbol{A}_{k-1}+{\boldsymbol{\Sigma}_{k-1|k-1}}^{-1}\right)\boldsymbol{\alpha}={\boldsymbol{A}_{k-1}}^T{\boldsymbol{\Sigma}_w}^{-1}\boldsymbol{x}_k+{\boldsymbol{\Sigma}_{k-1|k-1}}^{-1}\boldsymbol{\mu}_{k-1|k-1}$$

$$\therefore \boldsymbol{\alpha}=\left({\boldsymbol{A}_{k-1}}^T{\boldsymbol{\Sigma}_w}^{-1}\boldsymbol{A}_{k-1}+{\boldsymbol{\Sigma}_{k-1|k-1}}^{-1}\right)^{-1}\left({\boldsymbol{A}_{k-1}}^T{\boldsymbol{\Sigma}_w}^{-1}\boldsymbol{x}_k+{\boldsymbol{\Sigma}_{k-1|k-1}}^{-1}\boldsymbol{\mu}_{k-1|k-1}\right)$$

$\boldsymbol{x}_{k-1}$の0次の項を比較することによってβが求まる.

$$\boldsymbol{\alpha}^T\left({\boldsymbol{A}_{k-1}}^T{\boldsymbol{\Sigma}_w}^{-1}\boldsymbol{A}_{k-1}+{\boldsymbol{\Sigma}_{k-1|k-1}}^{-1}\right)\boldsymbol{\alpha}+\beta$$

$$={\boldsymbol{x}_k}^T{\boldsymbol{\Sigma}_w}^{-1}\boldsymbol{x}_k+{\boldsymbol{\mu}_{k-1|k-1}}^T{\boldsymbol{\Sigma}_{k-1|k-1}}^{-1}\boldsymbol{\mu}_{k-1|k-1}$$

カルマンフィルタ

$$\beta = \boldsymbol{x}_k{}^T \boldsymbol{\Sigma}_w{}^{-1} \boldsymbol{x}_k + \boldsymbol{\mu}_{k-1|k-1}{}^T \boldsymbol{\Sigma}_{k-1|k-1}{}^{-1} \boldsymbol{\mu}_{k-1|k-1}$$

$$- \left(\boldsymbol{x}_k{}^T \boldsymbol{\Sigma}_w{}^{-1} \boldsymbol{A}_{k-1} + \boldsymbol{\mu}_{k-1|k-1}{}^T \boldsymbol{\Sigma}_{k-1|k-1}{}^{-1} \right)$$

$$\times \left(\boldsymbol{A}_{k-1}{}^T \boldsymbol{\Sigma}_w{}^{-1} \boldsymbol{A}_{k-1} + \boldsymbol{\Sigma}_{k-1|k-1}{}^{-1} \right)^{-1} \left(\boldsymbol{A}_{k-1}{}^T \boldsymbol{\Sigma}_w{}^{-1} \boldsymbol{x}_k + \boldsymbol{\Sigma}_{k-1|k-1}{}^{-1} \boldsymbol{\mu}_{k-1|k-1} \right)$$

$p(\boldsymbol{x}_k|D_{k-1})$ は,

$$p(\boldsymbol{x}_k|D_{k-1}) = \frac{1}{Z_1}$$

$$\times \int \exp\left[-\frac{1}{2} (\boldsymbol{x}_{k-1} - \boldsymbol{\alpha})^T \left(\boldsymbol{A}_{k-1}{}^T \boldsymbol{\Sigma}_w{}^{-1} \boldsymbol{A}_{k-1} + \boldsymbol{\Sigma}_{k-1|k-1}{}^{-1} \right) (\boldsymbol{x}_{k-1} - \boldsymbol{\alpha}) + \beta \right] d\boldsymbol{x}_{k-1}$$

$$= \frac{1}{Z_2} \exp\left(-\frac{1}{2} \beta \right)$$

Z_1, Z_2は $p(\boldsymbol{x}_k|D_{k-1})$が正規分布となるための定数部

カルマンフィルタ

β を$\boldsymbol{x}_k$に関して平方完成する

$$\beta = (\boldsymbol{x}_k - \boldsymbol{\gamma})^T \left\{ {\boldsymbol{\Sigma}_w}^{-1} - {\boldsymbol{\Sigma}_w}^{-1} \boldsymbol{A}_{k-1} \left({\boldsymbol{A}_{k-1}}^T {\boldsymbol{\Sigma}_w}^{-1} \boldsymbol{A}_{k-1} + {\boldsymbol{\Sigma}_{k-1|k-1}}^{-1} \right)^{-1} {\boldsymbol{A}_{k-1}}^T {\boldsymbol{\Sigma}_w}^{-1} \right\} \times (\boldsymbol{x}_k - \boldsymbol{\gamma}) + \varepsilon$$

公式 $\left(\widetilde{\boldsymbol{A}} + \widetilde{\boldsymbol{B}}\widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{D}}\right)^{-1} = \widetilde{\boldsymbol{A}}^{-1} - \widetilde{\boldsymbol{A}}^{-1}\widetilde{\boldsymbol{B}}\left(\widetilde{\boldsymbol{D}}\widetilde{\boldsymbol{A}}^{-1}\widetilde{\boldsymbol{B}} + \widetilde{\boldsymbol{C}}^{-1}\right)^{-1}\widetilde{\boldsymbol{D}}\widetilde{\boldsymbol{A}}^{-1}$において

$\widetilde{\boldsymbol{A}} = \boldsymbol{\Sigma}_w, \widetilde{\boldsymbol{B}} = \boldsymbol{A}_{k-1}, \widetilde{\boldsymbol{C}} = \boldsymbol{\Sigma}_{k-1|k-1}, \widetilde{\boldsymbol{D}} = {\boldsymbol{A}_{k-1}}^T$ とすると

$$ {\boldsymbol{\Sigma}_w}^{-1} - {\boldsymbol{\Sigma}_w}^{-1} \boldsymbol{A}_{k-1} \left({\boldsymbol{A}_{k-1}}^T {\boldsymbol{\Sigma}_w}^{-1} \boldsymbol{A}_{k-1} + {\boldsymbol{\Sigma}_{k-1|k-1}}^{-1} \right)^{-1} {\boldsymbol{A}_{k-1}}^T {\boldsymbol{\Sigma}_w}^{-1} = \left(\boldsymbol{\Sigma}_w + \boldsymbol{A}_{k-1} \boldsymbol{\Sigma}_{k-1|k-1} {\boldsymbol{A}_{k-1}}^T \right)^{-1} $$

が得られる. したがって, β は

$$\beta = (\boldsymbol{x}_k - \boldsymbol{\gamma})^T \left(\boldsymbol{\Sigma}_w + \boldsymbol{A}_{k-1} \boldsymbol{\Sigma}_{k-1|k-1} {\boldsymbol{A}_{k-1}}^T \right)^{-1} (\boldsymbol{x}_k - \boldsymbol{\gamma}) + \varepsilon$$

と表せる.

カルマンフィルタ

β において$\boldsymbol{x}_k$ の1次の項を比較すると

$$\left(\boldsymbol{\Sigma}_w + \boldsymbol{A}_{k-1}\boldsymbol{\Sigma}_{k-1|k-1}{\boldsymbol{A}_{k-1}}^T\right)^{-1}\boldsymbol{\gamma}$$

$$= {\boldsymbol{\Sigma}_w}^{-1}\boldsymbol{A}_{k-1}\left({\boldsymbol{A}_{k-1}}^T{\boldsymbol{\Sigma}_w}^{-1}\boldsymbol{A}_{k-1} + {\boldsymbol{\Sigma}_{k-1|k-1}}^{-1}\right)^{-1}{\boldsymbol{\Sigma}_{k-1|k-1}}^{-1}\boldsymbol{\mu}_{k-1|k-1}$$

$$\boldsymbol{\gamma} = \left(\boldsymbol{\Sigma}_w + \boldsymbol{A}_{k-1}\boldsymbol{\Sigma}_{k-1|k-1}{\boldsymbol{A}_{k-1}}^T\right){\boldsymbol{\Sigma}_w}^{-1}\boldsymbol{A}_{k-1}$$

$$\times\left({\boldsymbol{A}_{k-1}}^T{\boldsymbol{\Sigma}_w}^{-1}\boldsymbol{A}_{k-1} + {\boldsymbol{\Sigma}_{k-1|k-1}}^{-1}\right)^{-1}{\boldsymbol{\Sigma}_{k-1|k-1}}^{-1}\boldsymbol{\mu}_{k-1|k-1}$$

$$= \left(\boldsymbol{I} + \boldsymbol{A}_{k-1}\boldsymbol{\Sigma}_{k-1|k-1}{\boldsymbol{A}_{k-1}}^T{\boldsymbol{\Sigma}_w}^{-1}\right)\boldsymbol{A}_{k-1}$$

$$\times\left(\boldsymbol{\Sigma}_{k-1|k-1}{\boldsymbol{A}_{k-1}}^T{\boldsymbol{\Sigma}_w}^{-1}\boldsymbol{A}_{k-1} + \boldsymbol{I}\right)^{-1}\boldsymbol{\mu}_{k-1|k-1}$$

$$= \boldsymbol{A}_{k-1}\left({\boldsymbol{A}_{k-1}}^{-1} + \boldsymbol{\Sigma}_{k-1|k-1}{\boldsymbol{A}_{k-1}}^T{\boldsymbol{\Sigma}_w}^{-1}\right)\boldsymbol{A}_{k-1}$$

$$\times\left(\boldsymbol{\Sigma}_{k-1|k-1}{\boldsymbol{A}_{k-1}}^T{\boldsymbol{\Sigma}_w}^{-1}\boldsymbol{A}_{k-1} + \boldsymbol{I}\right)^{-1}\boldsymbol{\mu}_{k-1|k-1}$$

カルマンフィルタ

$$= \boldsymbol{A}_{k-1}\left(\boldsymbol{I} + \boldsymbol{\Sigma}_{k-1|k-1}{\boldsymbol{A}_{k-1}}^{T}{\boldsymbol{\Sigma}_{w}}^{-1}\boldsymbol{A}_{k-1}\right)$$

$$\times \left(\boldsymbol{\Sigma}_{k-1|k-1}{\boldsymbol{A}_{k-1}}^{T}{\boldsymbol{\Sigma}_{w}}^{-1}\boldsymbol{A}_{k-1} + \boldsymbol{I}\right)^{-1}\boldsymbol{\mu}_{k-1|k-1}$$

$$= \boldsymbol{A}_{k-1}\,\boldsymbol{\mu}_{k-1|k-1}$$

したがって，$p(\boldsymbol{x}_k|D_{k-1})$ は

$$p(\boldsymbol{x}_k|D_{k-1}) = \frac{1}{Z_2}\exp\left\{-\frac{1}{2}\left(\boldsymbol{x}_k - \boldsymbol{A}_{k-1}\,\boldsymbol{\mu}_{k-1|k-1}\right)^{T}\right.$$

$$\times \left(\boldsymbol{\Sigma}_{w} + \boldsymbol{A}_{k-1}\boldsymbol{\Sigma}_{k-1|k-1}{\boldsymbol{A}_{k-1}}^{T}\right)^{-1}$$

$$\left.\times \left(\boldsymbol{x}_k - \boldsymbol{A}_{k-1}\,\boldsymbol{\mu}_{k-1|k-1}\right)\right\}$$

となり，平均ベクトル $\boldsymbol{\mu}_{k|k-1}$，分散共分散行列$\boldsymbol{\Sigma}_{k|k-1}$

$$\boldsymbol{\mu}_{k|k-1} = \boldsymbol{A}_{k-1}\,\boldsymbol{\mu}_{k-1|k-1}$$

$$\boldsymbol{\Sigma}_{k|k-1} = \boldsymbol{\Sigma}_{w} + \boldsymbol{A}_{k-1}\boldsymbol{\Sigma}_{k-1|k-1}{\boldsymbol{A}_{k-1}}^{T}$$

の正規分布となる.

カルマンフィルタ

次に，$p(\boldsymbol{x}_k|D_k)$ について考える（観測値$\boldsymbol{y}_k$を利用して状態$\boldsymbol{x}_k$を更新する）

$$p(\boldsymbol{x}_k|D_k) = \frac{p(\boldsymbol{y}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|D_{k-1})}{p(\boldsymbol{y}_k|D_{k-1})}$$

$$p(\boldsymbol{y}_k|\boldsymbol{x}_k) = \int p(\boldsymbol{y}_k|\boldsymbol{x}_k, \boldsymbol{v}_k)p(\boldsymbol{v}_k|\boldsymbol{x}_k)d\boldsymbol{v}_k$$

$$= \int \delta(\boldsymbol{y}_k - \boldsymbol{C}_k\,\boldsymbol{x}_k - \boldsymbol{v}_k)p(\boldsymbol{v}_k)d\boldsymbol{v}_k$$

であった.

ノイズ$\boldsymbol{v}_k$は平均ゼロ，分散・共分散行列$\boldsymbol{\Sigma}_v$の正規分布に従うことから

$$p(\boldsymbol{y}_k|\boldsymbol{x}_k) = \frac{1}{\sqrt{(2\pi)^p|\boldsymbol{\Sigma}_v|}}$$

$$\times \exp\left\{-\frac{1}{2}(\boldsymbol{y}_k - \boldsymbol{C}_k\,\boldsymbol{x}_k - \boldsymbol{v}_k)^T{\boldsymbol{\Sigma}_v}^{-1}(\boldsymbol{y}_k - \boldsymbol{C}_k\,\boldsymbol{x}_k - \boldsymbol{v}_k)\right\}$$

カルマンフィルタ

$p(\boldsymbol{x}_k|D_{k-1})$は平均 $\boldsymbol{\mu}_{k|k-1}$，分散共分散行列$\boldsymbol{\Sigma}_{k|k-1}$の正規分布であるので

$$p(\boldsymbol{x}_k|D_k) \propto \exp\left\{-\frac{1}{2}(\boldsymbol{y}_k - \boldsymbol{C}_k\,\boldsymbol{x}_k - \boldsymbol{v}_k)^T {\boldsymbol{\Sigma}_v}^{-1}(\boldsymbol{y}_k - \boldsymbol{C}_k\,\boldsymbol{x}_k - \boldsymbol{v}_k)\right.$$
$$\left. -\frac{1}{2}\left(\boldsymbol{x}_k - \boldsymbol{\mu}_{k|k-1}\right)^T {\boldsymbol{\Sigma}_{k|k-1}}^{-1}\left(\boldsymbol{x}_k - \boldsymbol{\mu}_{k|k-1}\right)\right\}$$

exp の指数部は以下となる

$$-\frac{1}{2}\left[{\boldsymbol{y}_k}^T {\boldsymbol{\Sigma}_v}^{-1}\boldsymbol{y}_k - 2{\boldsymbol{x}_k}^T {\boldsymbol{C}_k}^T {\boldsymbol{\Sigma}_v}^{-1}\boldsymbol{y}_k + {\boldsymbol{x}_k}^T {\boldsymbol{C}_k}^T {\boldsymbol{\Sigma}_v}^{-1}\boldsymbol{C}_k\,\boldsymbol{x}_k\right.$$
$$\left. +{\boldsymbol{x}_k}^T {\boldsymbol{\Sigma}_{k|k-1}}^{-1}\boldsymbol{x}_k - 2{\boldsymbol{x}_k}^T {\boldsymbol{\Sigma}_{k|k-1}}^{-1}\,\boldsymbol{\mu}_{k|k-1} + \,{\boldsymbol{\mu}_{k|k-1}}^T {\boldsymbol{\Sigma}_{k|k-1}}^{-1}\,\boldsymbol{\mu}_{k|k-1}\right]$$

カルマンフィルタ

これを$\boldsymbol{x}_k$に関して以下のように平方完成する.

$$-\frac{1}{2}\left[(\boldsymbol{x}_k-\boldsymbol{\alpha})^T\left({\boldsymbol{\Sigma}_{k|k-1}}^{-1}+{\boldsymbol{C}_k}^T{\boldsymbol{\Sigma}_v}^{-1}\boldsymbol{C}_k\right)(\boldsymbol{x}_k-\boldsymbol{\alpha})+\beta\right]$$

$\boldsymbol{x}_k$の1次の項を比較すると

$$\left({\boldsymbol{\Sigma}_{k|k-1}}^{-1}+{\boldsymbol{C}_k}^T{\boldsymbol{\Sigma}_v}^{-1}\boldsymbol{C}_k\right)\boldsymbol{\alpha}={\boldsymbol{C}_k}^T{\boldsymbol{\Sigma}_v}^{-1}\boldsymbol{y}_k+{\boldsymbol{\Sigma}_{k|k-1}}^{-1}\boldsymbol{\mu}_{k|k-1}$$

$$\boldsymbol{\alpha}=\left({\boldsymbol{\Sigma}_{k|k-1}}^{-1}+{\boldsymbol{C}_k}^T{\boldsymbol{\Sigma}_v}^{-1}\boldsymbol{C}_k\right)^{-1}\left({\boldsymbol{C}_k}^T{\boldsymbol{\Sigma}_v}^{-1}\boldsymbol{y}_k+{\boldsymbol{\Sigma}_{k|k-1}}^{-1}\boldsymbol{\mu}_{k|k-1}\right)$$

公式$\left(\widetilde{\boldsymbol{A}}+\widetilde{\boldsymbol{B}}\widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{D}}\right)^{-1}=\widetilde{\boldsymbol{A}}^{-1}-\widetilde{\boldsymbol{A}}^{-1}\widetilde{\boldsymbol{B}}\left(\widetilde{\boldsymbol{D}}\widetilde{\boldsymbol{A}}^{-1}\widetilde{\boldsymbol{B}}+\widetilde{\boldsymbol{C}}^{-1}\right)^{-1}\widetilde{\boldsymbol{D}}\widetilde{\boldsymbol{A}}^{-1}$において

$\widetilde{\boldsymbol{A}}={\boldsymbol{\Sigma}_{k|k-1}}^{-1},\ \widetilde{\boldsymbol{B}}={\boldsymbol{C}_k}^T,\ \widetilde{\boldsymbol{C}}={\boldsymbol{\Sigma}_v}^{-1},\ \widetilde{\boldsymbol{D}}=\boldsymbol{C}_k$とすると

$$\boldsymbol{\alpha}=\left(\boldsymbol{\Sigma}_{k|k-1}-\boldsymbol{\Sigma}_{k|k-1}{\boldsymbol{C}_k}^T\left(\boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}{\boldsymbol{C}_k}^T+\boldsymbol{\Sigma}_v\right)^{-1}\boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}\right)$$
$$\times\left({\boldsymbol{C}_k}^T{\boldsymbol{\Sigma}_v}^{-1}\boldsymbol{y}_k+{\boldsymbol{\Sigma}_{k|k-1}}^{-1}\boldsymbol{\mu}_{k|k-1}\right)$$

カルマンフィルタ

$$\boldsymbol{\alpha} = \boldsymbol{\mu}_{k|k-1} - \boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T\left(\boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T + \boldsymbol{\Sigma}_v\right)^{-1}\boldsymbol{C}_k\,\boldsymbol{\mu}_{k|k-1}$$

$$+\left(\boldsymbol{\Sigma}_{k|k-1} - \boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T\left(\boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T + \boldsymbol{\Sigma}_v\right)^{-1}\boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}\right)\boldsymbol{C}_k{}^T\boldsymbol{\Sigma}_v{}^{-1}\boldsymbol{y}_k$$

$$= \boldsymbol{\mu}_{k|k-1} - \boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T\left(\boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T + \boldsymbol{\Sigma}_v\right)^{-1}\boldsymbol{C}_k\,\boldsymbol{\mu}_{k|k-1}$$

$$+\boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T\left(\boldsymbol{I} - \underline{\underline{\left(\boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T + \boldsymbol{\Sigma}_v\right)^{-1}\boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T}}\right)\boldsymbol{\Sigma}_v{}^{-1}\boldsymbol{y}_k$$

上式の二重線部分に対して,

公式 $\left(\widetilde{\boldsymbol{A}} + \widetilde{\boldsymbol{B}}\widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{D}}\right)^{-1} = \widetilde{\boldsymbol{A}}^{-1} - \widetilde{\boldsymbol{A}}^{-1}\widetilde{\boldsymbol{B}}\left(\widetilde{\boldsymbol{D}}\widetilde{\boldsymbol{A}}^{-1}\widetilde{\boldsymbol{B}} + \widetilde{\boldsymbol{C}}^{-1}\right)^{-1}\widetilde{\boldsymbol{D}}\widetilde{\boldsymbol{A}}^{-1}$を適用する

$\widetilde{\boldsymbol{A}} = \boldsymbol{I},\ \widetilde{\boldsymbol{B}} = \boldsymbol{I},\ \widetilde{\boldsymbol{C}} = \boldsymbol{\Sigma}_v{}^{-1},\ \widetilde{\boldsymbol{D}} = \boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T$とすると

$$\boldsymbol{\alpha} = \boldsymbol{\mu}_{k|k-1} - \boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T\left(\boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T + \boldsymbol{\Sigma}_v\right)^{-1}\boldsymbol{C}_k\,\boldsymbol{\mu}_{k|k-1}$$

$$+\boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T\left(\boldsymbol{I} + \boldsymbol{\Sigma}_v{}^{-1}\boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T\right)^{-1}\boldsymbol{\Sigma}_v{}^{-1}\boldsymbol{y}_k$$

カルマンフィルタ

$$\boldsymbol{\alpha} = \boldsymbol{\mu}_{k|k-1} - \boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T\left(\boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T + \boldsymbol{\Sigma}_v\right)^{-1}\boldsymbol{C}_k\,\boldsymbol{\mu}_{k|k-1} + \boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T\left(\boldsymbol{\Sigma}_v + \boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T\right)^{-1}\boldsymbol{y}_k$$

$\boldsymbol{K} = \boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T\left(\boldsymbol{\Sigma}_v + \boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T\right)^{-1}$とおくと

$\boldsymbol{\alpha} = \boldsymbol{\mu}_{k|k-1} + \boldsymbol{K}\left(\boldsymbol{y}_k - \boldsymbol{C}_k\,\boldsymbol{\mu}_{k|k-1}\right)$ となる.

すなわち, $p(\boldsymbol{x}_k|D_k)$の平均ベクトル $\boldsymbol{\mu}_{k|k}$ は

$$\boldsymbol{\mu}_{k|k} = \boldsymbol{\mu}_{k|k-1} + \boldsymbol{K}\left(\boldsymbol{y}_k - \boldsymbol{C}_k\,\boldsymbol{\mu}_{k|k-1}\right)$$

また, 分散共分散行列$\boldsymbol{\Sigma}_{k|k}$ は

$$\begin{aligned}\boldsymbol{\Sigma}_{k|k} &= \left(\boldsymbol{\Sigma}_{k|k-1}{}^{-1} + \boldsymbol{C}_k{}^T\boldsymbol{\Sigma}_v{}^{-1}\boldsymbol{C}_k\right)^{-1}\\ &= \boldsymbol{\Sigma}_{k|k-1} - \boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T\left(\boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}\boldsymbol{C}_k{}^T + \boldsymbol{\Sigma}_v\right)^{-1}\boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}\\ &= \boldsymbol{\Sigma}_{k|k-1} - \boldsymbol{K}\,\boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}\end{aligned}$$

カルマンフィルタ

【まとめ】

予測：状態 $\boldsymbol{x}_k$ の平均・分散共分散行列は

$$\boldsymbol{\mu}_{k|k-1} = \boldsymbol{A}_{k-1}\,\boldsymbol{\mu}_{k-1|k-1}$$

$$\boldsymbol{\Sigma}_{k|k-1} = \boldsymbol{\Sigma}_w + \boldsymbol{A}_{k-1}\boldsymbol{\Sigma}_{k-1|k-1}{\boldsymbol{A}_{k-1}}^T$$

更新：状態 $\boldsymbol{x}_k$ の平均・分散共分散行列は

$$\boldsymbol{K} = \boldsymbol{\Sigma}_{k|k-1}{\boldsymbol{C}_k}^T\left(\boldsymbol{\Sigma}_v + \boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}{\boldsymbol{C}_k}^T\right)^{-1}$$

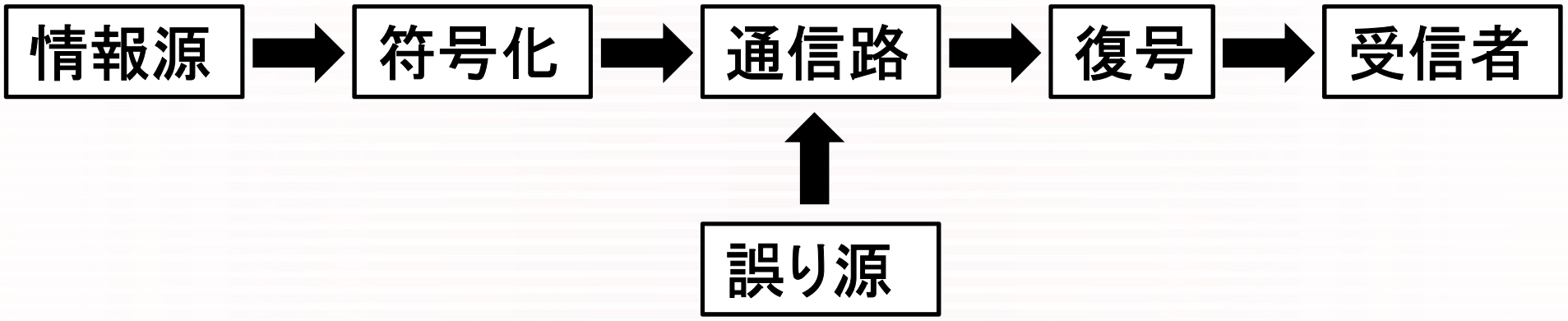
$$\boldsymbol{\mu}_{k|k} = \boldsymbol{\mu}_{k|k-1} + \boldsymbol{K}\left(\boldsymbol{y}_k - \boldsymbol{C}_k\,\boldsymbol{\mu}_{k|k-1}\right)$$

$$\boldsymbol{\Sigma}_{k|k} = \boldsymbol{\Sigma}_{k|k-1} - \boldsymbol{K}\,\boldsymbol{C}_k\boldsymbol{\Sigma}_{k|k-1}$$

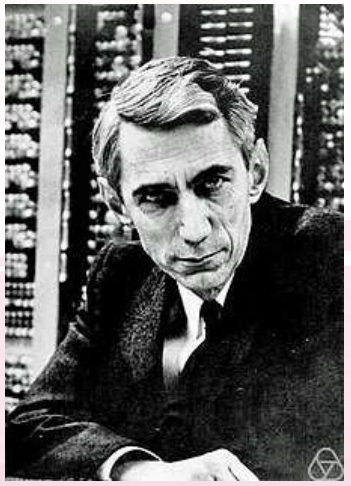
上記の予測と更新を繰り返して，状態の分布を観測値から推定する計算をカルマンフィルタと呼ぶ．

情報源符号化(データ圧縮)

通信における情報伝達のモデル

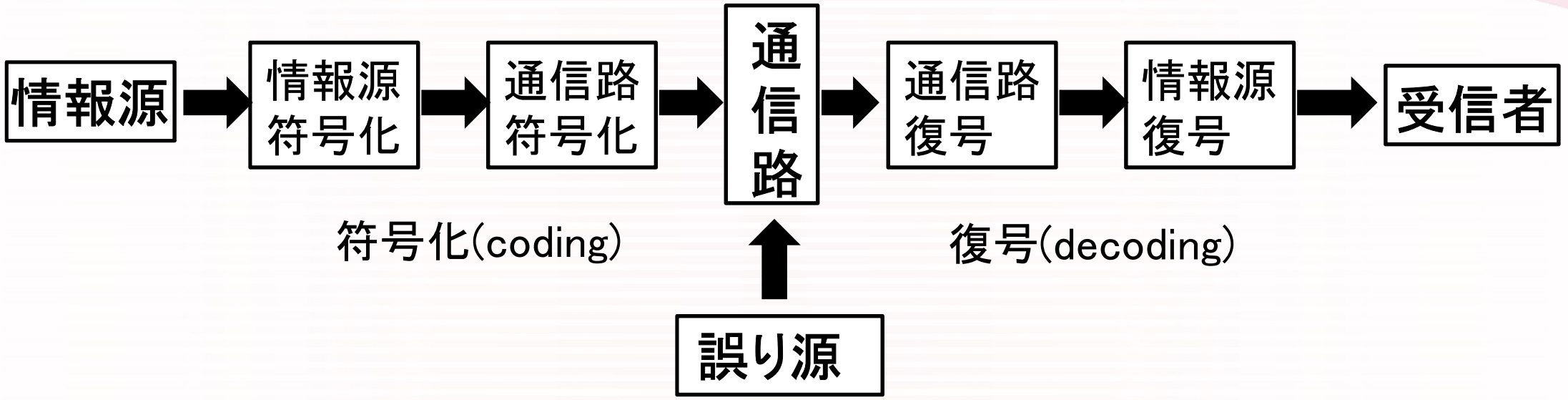


クロード・シャノン
「通信の数学的理論」1948年



Wikipedia

通信における情報伝達のモデル



情報源符号化：伝送効率向上のための符号化
（できるだけ少ない記号で）

通信路符号化：信頼性確保のための符号化
（通信路の誤りの影響を抑える冗長性確保）

情報の定量化

通報(事象)の生起確率が p のとき、その情報量 $f(p)$ を定義する

1. 確率が小さい事象が起こったという通報のほうが
   確率が大きい事象の通報より情報量が大きい。
   すなわち、$f(p)$ は p の減少関数である。

2. 独立事象E_1, E_2の生起確率をp_1, p_2とすると、E_1, E_2 が
   同時に起こる確率は $p_1 p_2$ であるから、
   $$f(p_1 p_2) = f(p_1) + f(p_2)$$

3. $f(p)$ は p の連続関数である。近い確率の事象の情報量は近い。

   $\Rightarrow f(p)$ は $-\log p$ の正の定数倍

情報の定量化

関数方程式 $f(p_1 p_2) = f(p_1) + f(p_2)$ を解く。

$p_1 = 2^{-q_1}$, $p_2 = 2^{-q_2}$とおくと、q_1, q_2は0以上の実数値をとる。
そこで、

$$g(x) = f(2^{-x})$$

とおけば、

$$g(q_1 + q_2) = g(q_1) + g(q_2)$$

を得る。

情報の定量化

定義: 事象 E の生起確率が p であるとき、
その情報量を $-\log_2 p$ と定める。(単位: ビット)

定義: M個の独立な通報 (事象) $a_1, \ldots, a_M$があり、各通報が送られる確率(事象の生起確率)が$p_1, \ldots, p_M$であるとする $(p_1 + \cdots + p_M = 1)$。このとき、
1通報(事象)あたりの平均情報量 (エントロピー) $H(A)$ を

$$H(A) = -\sum_{i=1}^{M} p_i \log_2 p_i$$

と定める。(= 独立生起情報源のエントロピー)

情報の定量化

例題: 4つの通報a_1, a_2, a_3, a_4をそれぞれ確率0.6, 0.2, 0.1, 0.1で発生する独立生起情報源がある。この情報源から発生する通報のエントロピーを求めよ。

解: 定義に従って、

$$-(0.6\log_2 0.6 + 0.2\log_2 0.2 + 0.1\log_2 0.1 + 0.1\log_2 0.1)$$

= 1.57 ビット。

エントロピーの性質

(1) エントロピーは0以上である。

証明:情報量が0以上であることから従う。

(2) すべての事象の生起確率が等しいとき、
エントロピーは最大になる。

証明:$\log_e x \leq x - 1 \quad (x > 0)$ を用いる
(等号成立は$x = 1$のとき)。

エントロピーの性質

(2) すべての事象の生起確率が等しいとき、
エントロピーは最大になる。

情報源符号化

符号化:　情報源記号系列からなる通報を
通信路記号系列に1対1に割り当てる

以下簡単のため、0と1のみからなる二元符号を扱う
(多元符号に一般化可能)

通報	符号I	符号II	符号III	符号IV
a_1	0	0	1	00
a_2	1	01	01	01
a_3	01	011	001	10
a_4	10	0111	0001	11

符号語: 割り当てられた通信路記号系列

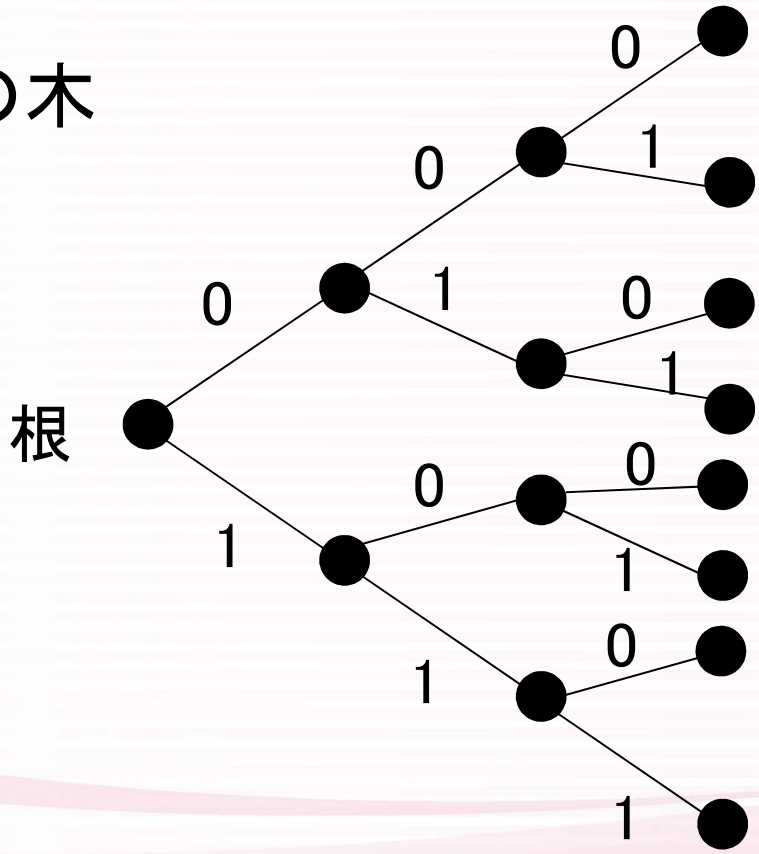
情報源符号化

通報	符号I	符号II	符号III	符号IV
a_1	0	0	1	00
a_2	1	01	01	01
a_3	01	011	001	10
a_4	10	0111	0001	11
一意復号可能?	×	○	○	○
瞬時復号可能?	×	×	○	○

木

木 (tree): 連結で、サイクルを持たない無向グラフ

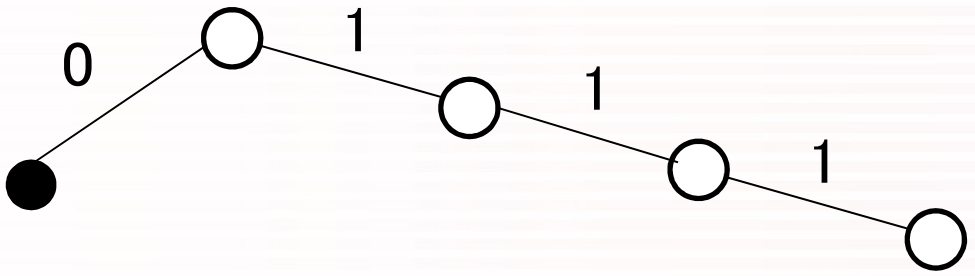
符号の木



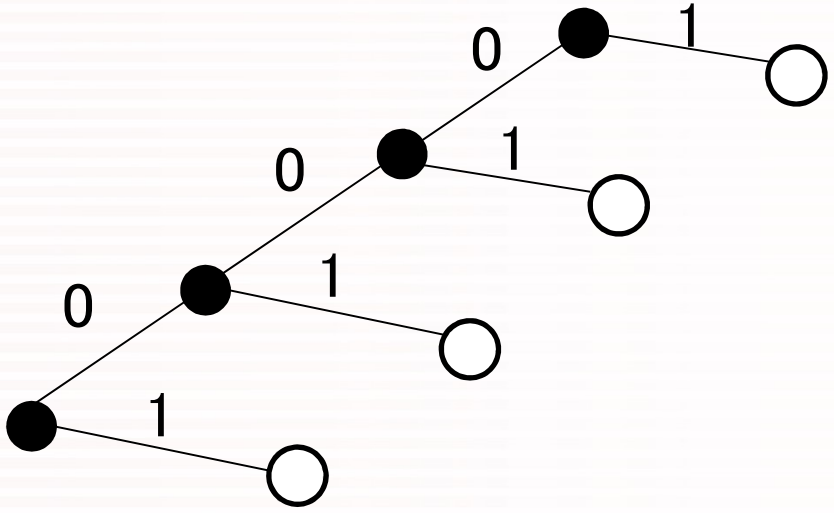
葉：根以外の頂点のうち、
接続する辺が1本しかないもの

符号の木

符号IIの木



符号IIIの木



符号が瞬時復号可能であるための必要十分条件は、
符号の木において、
すべての符号語○が葉に対応づけられること。

クラフトの不等式

符号語長が $l_1, \ldots, l_M$ のM個の符号語からなる、
瞬時復号可能な2元符号が構成できるための必要十分条件は

$$\sum_{i=1}^{M} 2^{-l_j} \leq 1$$

が成り立つことである。

(証明の方針) 符号の木を $\max(l_1, \ldots, l_m)$ 次まで伸ばして、
葉の個数を数える。
(注)「瞬時復号可能」を「一意復号可能」に変えても
成り立つことが知られている(マクミランの不等式)。

平均符号長

通報 $a_1, \ldots, a_M$ の生起確率をそれぞれ $p_1, \ldots, p_M$、
符号長をそれぞれ $l_1, \ldots, l_M$とすると、

1通報あたりの平均符号長Lは

$$L = \sum_{i=1}^{M} p_i l_i$$

で与えられる。

平均符号長

例題: 4つの通報a_1, a_2, a_3, a_4をそれぞれ確率0.6, 0.2, 0.1, 0.1で発生する独立生起情報源がある。この情報源を符号IIIで符号化したときの平均符号長Lを求めよ。

解: 定義に従って、
$L = 1 \times 0.6 + 2 \times 0.2 + 3 \times 0.1 + 4 \times 0.1 = 1.7$ ビット。

エントロピーが$H = 1.57$ビットだったことを思い出そう。

情報源符号化定理

一般に、瞬時復号可能な2元符号に対して

$$H \leq L$$

が成り立つ(注1)。また、

$$H \leq L \leq H + 1$$

をみたすような符号化が存在する(注2)。

(注1) 平均符号長 L はエントロピー H より小さくなれない。

(注2) 通報a_iに対応する符号語の長さl_iを、 $-\log_2 p_i$ 以上の最小の整数にとればよい(確率の高い通報の符号長を短く)

ハフマン符号

与えられた独立生起情報源からの通報を符号化するとき、
平均符号長を最小にする符号を最短符号(compact code)という。

最短符号の構成法として、ハフマン符号が知られ、
JPEGやZIPなどの圧縮フォーマットで用いられている。

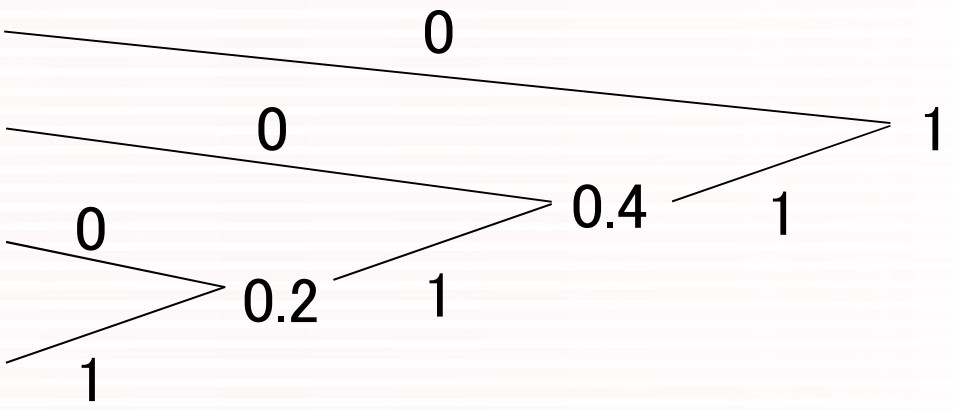
構成法：

1. *M*個の通報に対応する葉を作る。生起確率の最も小さい葉を2つ選び、枝で結んで頂点を作る。それぞれの枝に0, 1を割り当てる。
2. この頂点を新しい葉とみなし、もとの2つの葉の生起確率の和を新たな葉の生起確率とする。
3. 葉が1枚になるまで繰り返し、そこからもとの葉に至るまでの枝に割り当てられた数字を順に読む

ハフマン符号

例題: 4つの通報a_1, a_2, a_3, a_4をそれぞれ確率0.6, 0.2, 0.1, 0.1で発生する独立生起情報源がある。この通報を2元ハフマン符号で符号化せよ。

符号語	通報	確率
0	a_1	0.6
10	a_2	0.2
110	a_3	0.1
111	a_4	0.1



この符号の平均符号長は$1 \times 0.6 + 2 \times 0.2 + 3 \times 0.1 + 3 \times 0.1$ = 1.6ビットである。

データ構造（リスト、配列、木構造）

アルゴリズムとデータ構造

アルゴリズム

計算機を用いて何らかの問題を解く際の手続き(解き方)

プログラム

その解き方を計算機上で実際に実行可能な命令列として
表現したもの

データ構造

計算機を用いて計算や処理を効率的に行うのに適した
データの保持方法

コンピュータ(PC)のハードウェア構成

CPU (中央演算処理装置, Central Processing Unit)

命令を高速で処理する
クロック周波数：1秒間に何回命令処理ができるか
(GHz = 1秒間に10億回)

メインメモリ

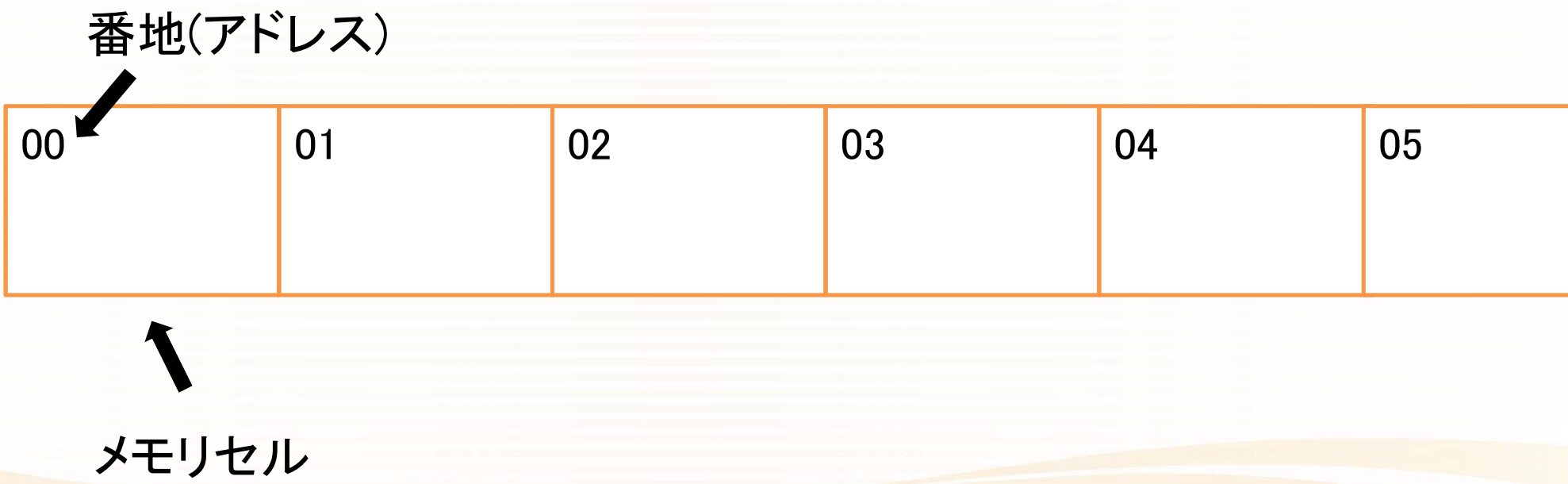
データの書き込みと読み込みに特化。CPUとはバスで通信。
電源を切ると内容が失われる(Dynamic RAM)

外部記憶装置・周辺機器(ディスプレイ・キーボードなど)

コントローラを介してバスに接続

メモリの構造の模式図

コンピュータで実行されるプログラムや、プログラムが
扱うデータを一時的に保持



データ構造とクエリ処理

・要素 x をデータ構造に挿入する

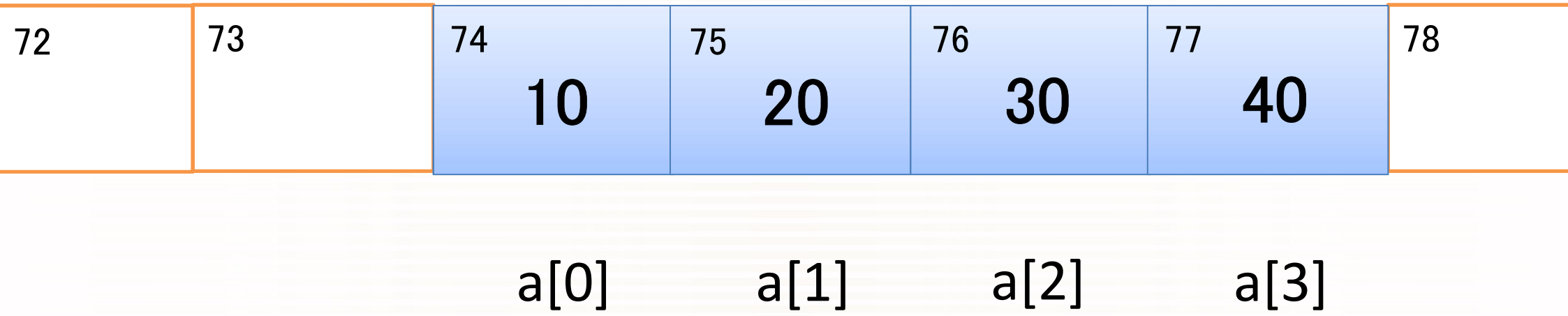
・要素 x をデータ構造から削除する

・要素 x がデータ構造に含まれるかどうかを判定する

⇒ 使用するデータ構造によって計算時間に差が生じる

配列(array)

連続するメモリ領域を必要なだけ確保することで、
順番を保持してメモリに記録できる



C++では std::vector、Pythonではlist

配列(array)の計算量

配列のサイズをNとする。

i 番目の要素へのアクセス : $O(1)$

要素 x を最後尾に挿入 : $O(1)$

要素 x を特定の要素 y の直後に挿入 : $O(N)$

要素 x を削除 : $O(N)$

要素 x を検索 : $O(N)$

ビッグ・オー記法

$O(1)$: Nによらない
定数

$O(N)$: Nの
1次式(以下)

ビッグ･オー記法

アルゴリズムの計算量を、入力サイズnの関数として表すとき、その漸近的上界を示す。

関数$f(x), g(x)$に対し、十分大きなnについて

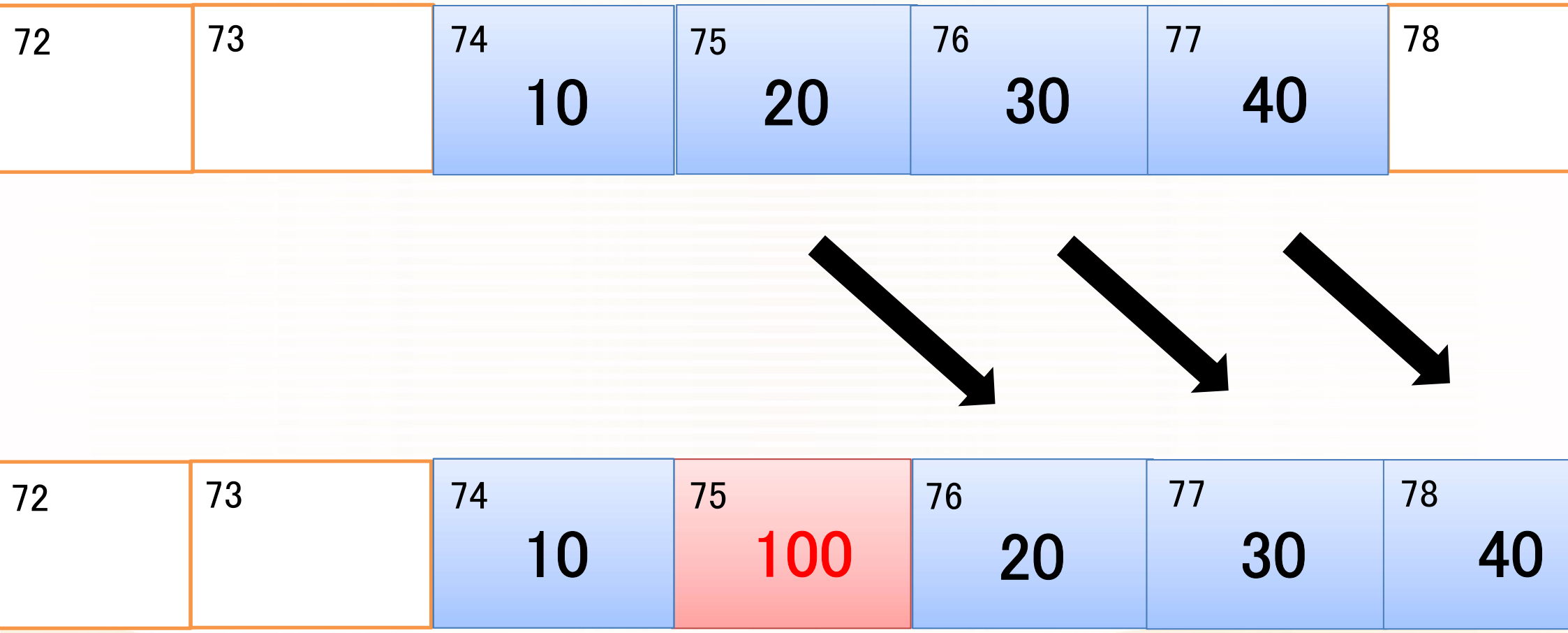
$$f(n) \leq c\, g(n)$$

となる定数cが存在するならば、

$$f(n) = O\big(g(n)\big)$$

と表し、$f(n)$のオーダーは$g(n)$であるという。

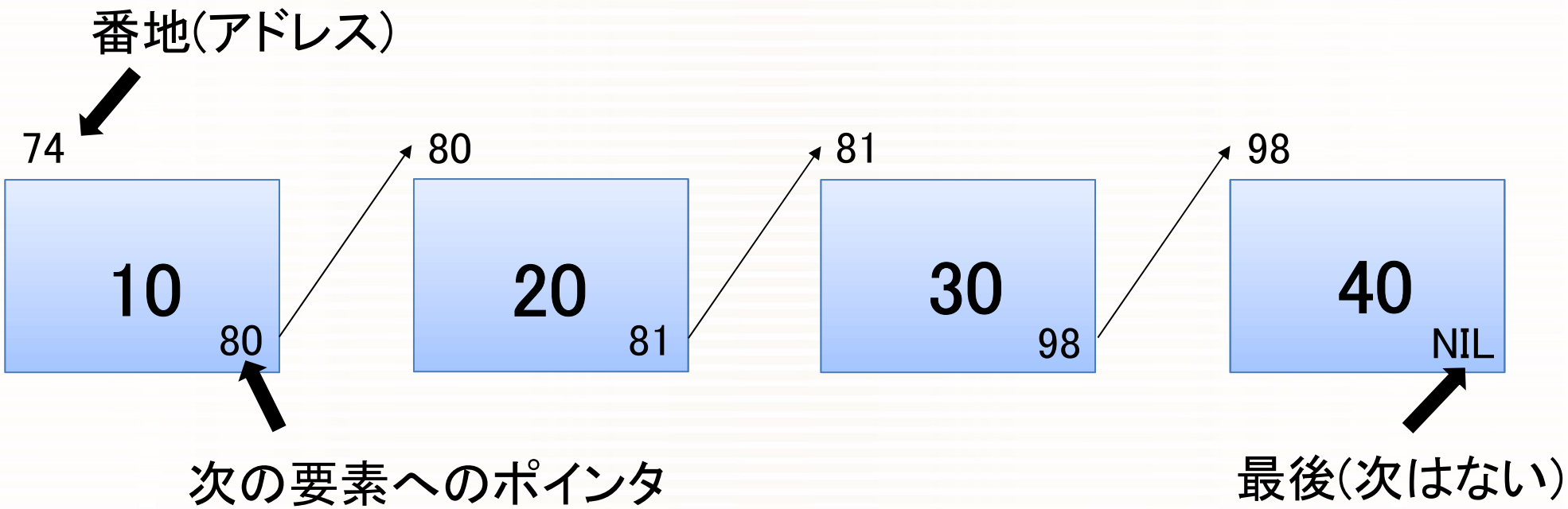
配列における要素の挿入



連結リスト(linked list)

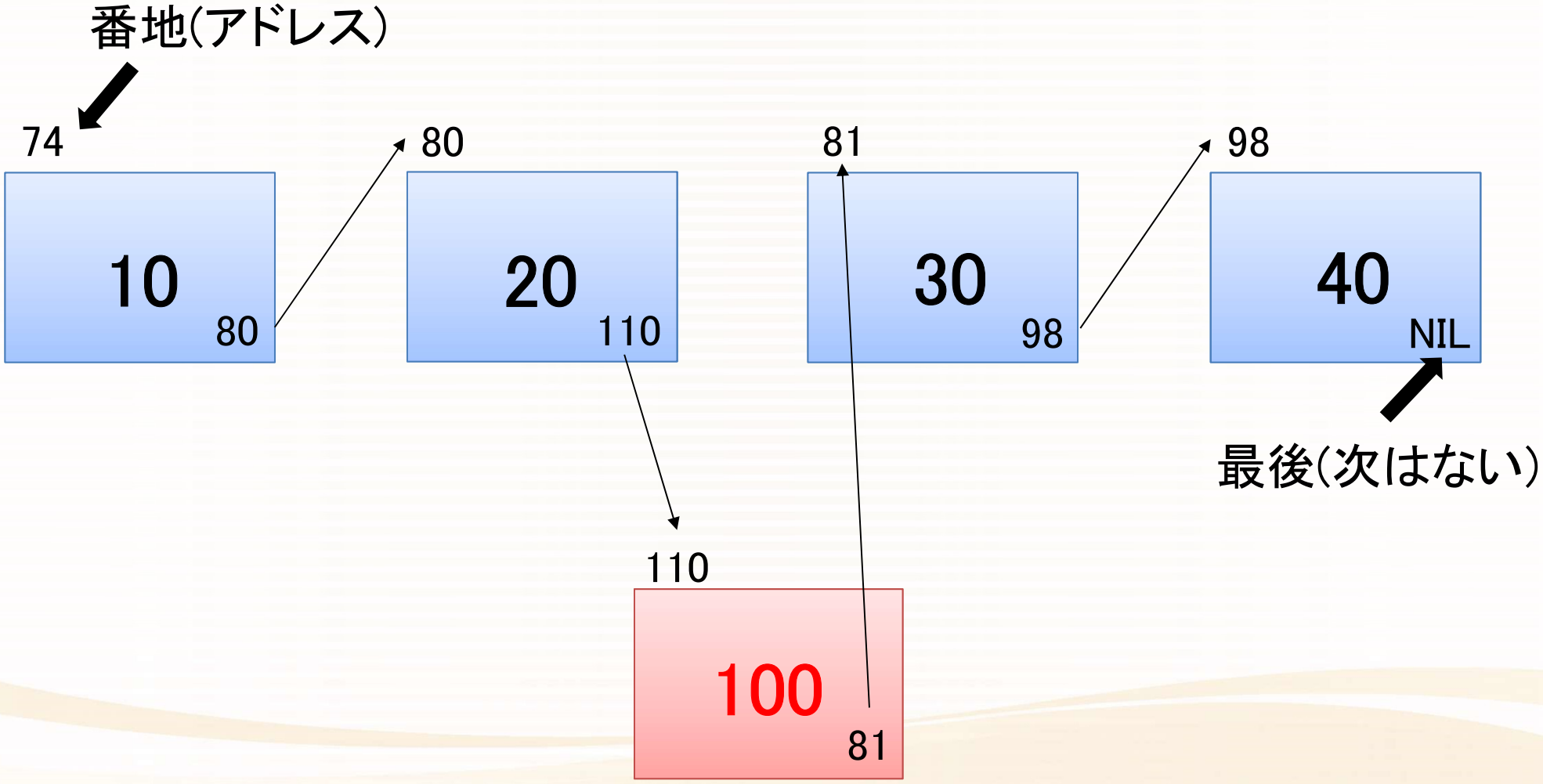
配列の弱点である挿入・削除クエリに強い

C++では
std::list



図：単方向連結リストの場合。双方向にもできる
(次の要素と前の要素へのポインタをもつ)

連結リストにおける要素の挿入



連結リスト(linked list)の計算量

配列のサイズをNとする。

i 番目の要素へのアクセス: $O(N)$

要素 x を最後尾に挿入: $O(1)$

要素 x を特定の要素 y の直後に挿入: $O(1)$

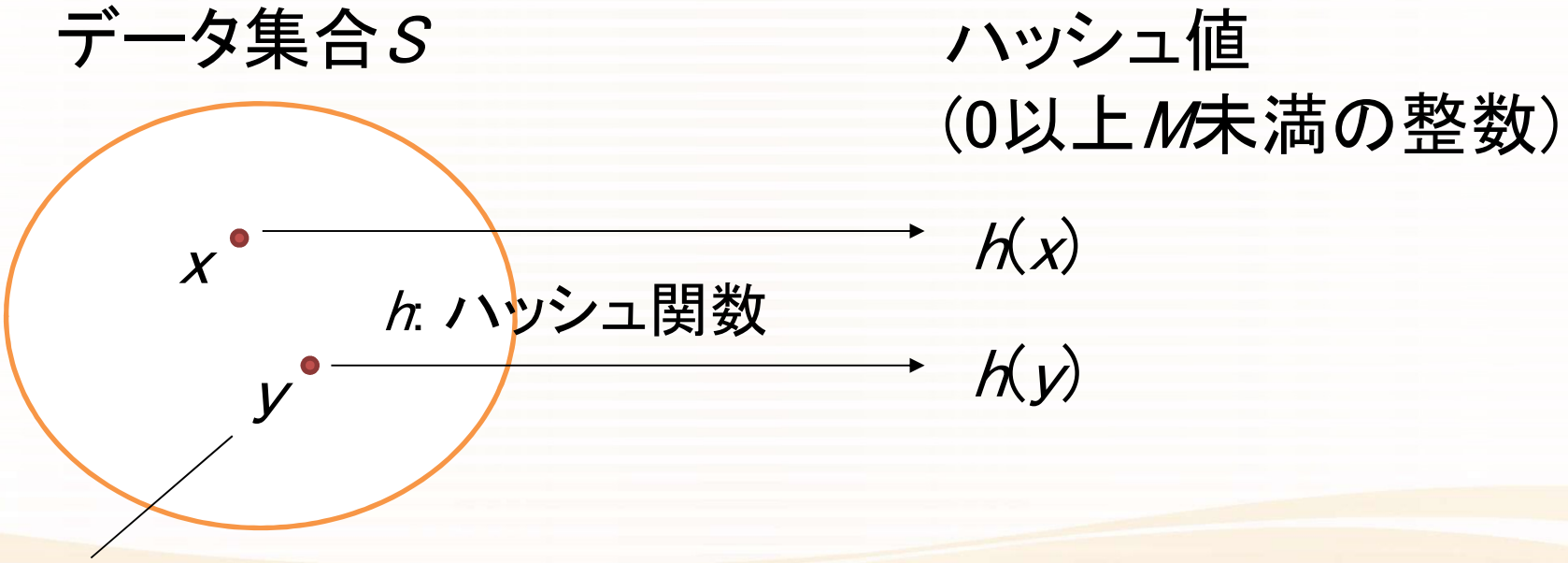
要素 x を削除: $O(1)$

要素 x を検索: $O(N)$

ハッシュテーブル

要素の検索が$O(1)$の計算量で
できるが、各要素間の順序に
関する情報をもたない

C++では std::unordered_set、
Pythonではset



ハッシュテーブル

配列 T を用意して、

要素 x の挿入: $T[h(x)] \leftarrow$ true

要素 x の削除: $T[h(x)] \leftarrow$ false

要素 x の検索: $T[h(x)]$ が true かどうか

いずれも計算量は平均的に $O(1)$。

ただし、ハッシュ値が等しい $h(x) = h(y)$ ものがある場合は、
それらで連結リストを作り、$T[h(x)]$ にその連結リストの
先頭を指すポインタを入れる。

スタックとキュー

配列や連結リストを用いて実現可能

クエリ
push(x): 要素 x をデータ構造に挿入する
pop(): データ構造から要素を1つ取り出す
isEmpty(): データ構造が空かどうかを調べる

C++ では std::stack, std::queue

スタックとキュー

pop() でデータ構造から要素を1つ取り出す際に、

スタックの場合は、last-in first-out (LIFO)、すなわち
最後に push された要素を取り出す。

例: Webブラウザの訪問履歴 (戻るボタンがpop)、
テキストエディタのUndo

配列での実現：左側が閉じているイメージ、
行き止まりのトンネルに要素を突っ込むイメージ

スタックとキュー

pop() でデータ構造から要素を1つ取り出す際に、

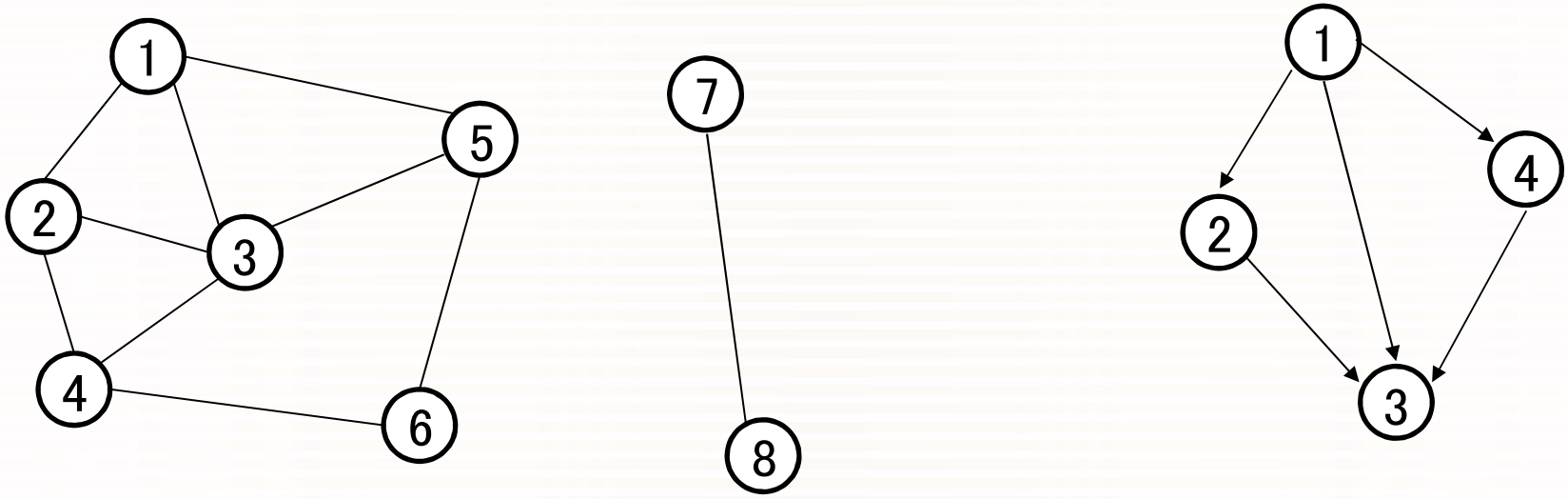
キューの場合は、first-in first-out (FIFO) 、すなわち
最初に push された要素を取り出す。

例: 航空券予約のキャンセル待ち処理、印刷機の
ジョブスケジューリング

配列での実現：両端が開いているイメージ。
右からenqueueして、左からdequeueする。

グラフと木

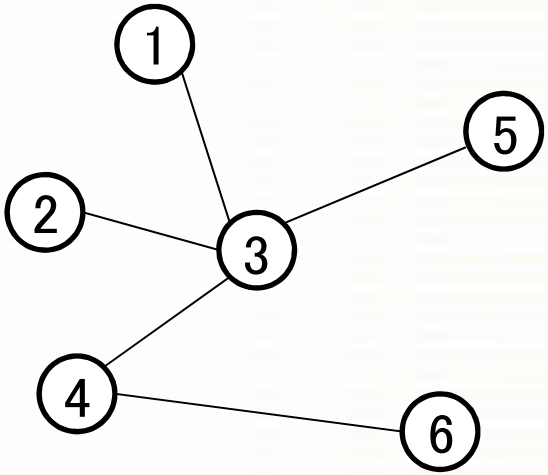
無向グラフ：頂点の集合 V と辺の集合 E の組
有向グラフ：グラフの各辺に向きがあるもの

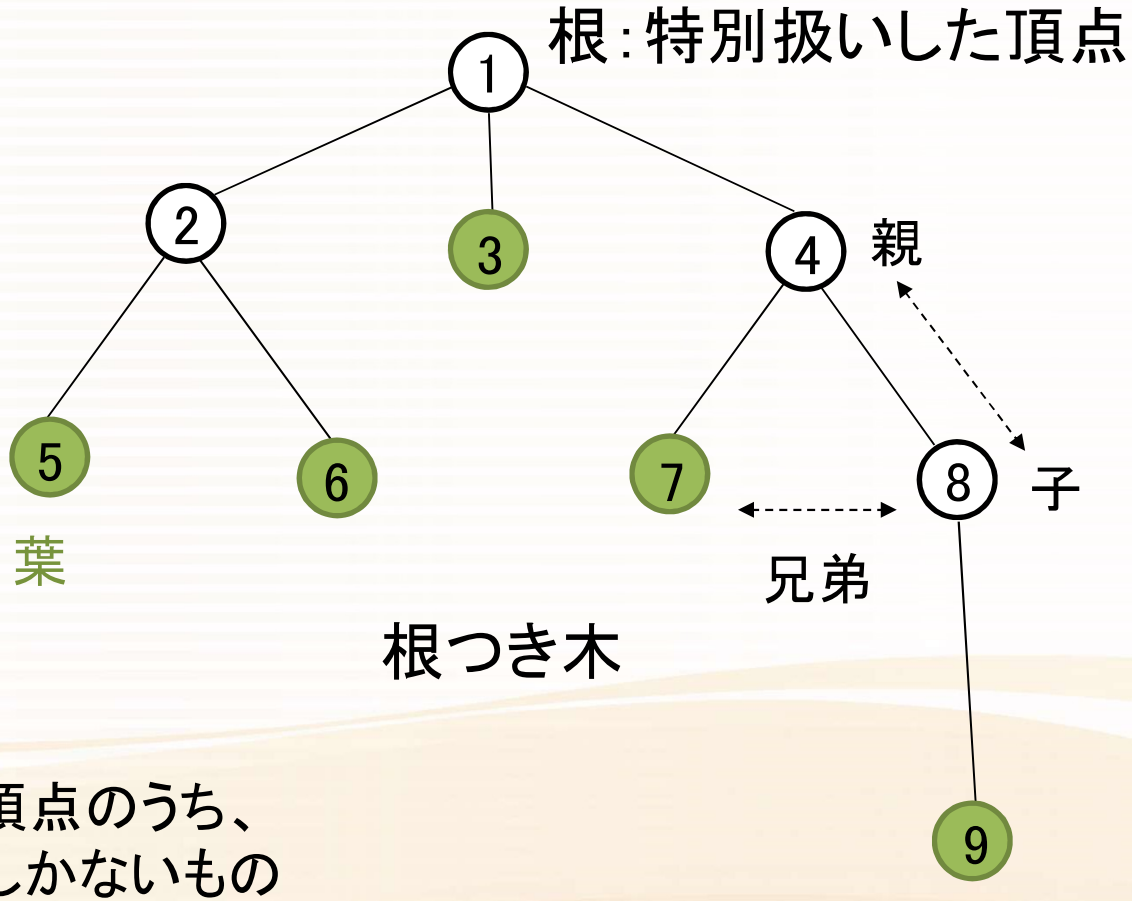


無向グラフの例: ソーシャルネットワーク、交通ネットワーク
有向グラフの例: タスクの依存関係、ゲームの局面遷移

グラフと木

木 (tree): 連結で、サイクルを持たない無向グラフ



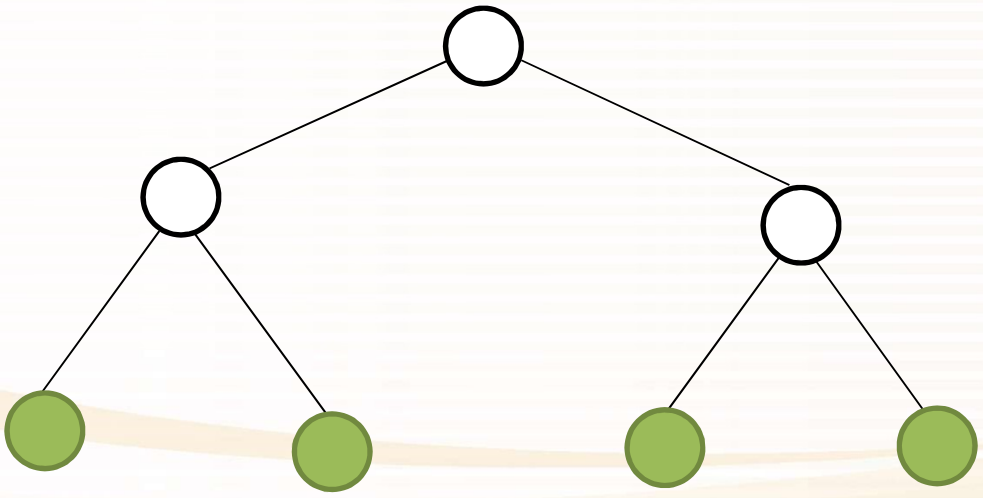


葉：根つき木において、根以外の頂点のうち、
その頂点に接続している辺が1本しかないもの

二分木 (binary tree)

順序木: 根つき木において、各頂点の子頂点の順序を考慮したもの。兄弟間で兄と弟の区別がつく

二分木: 順序木において、すべての頂点が高々 2 個の子頂点をもつもの。



葉

深さ: 根と頂点を結ぶパスの長さ
高さ: 深さの最大値

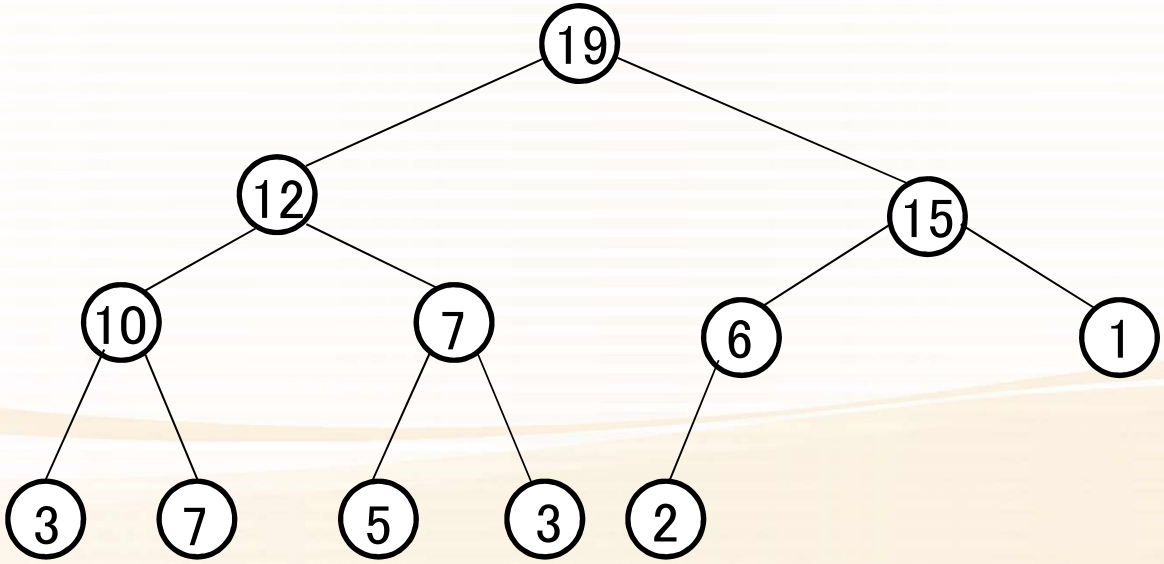
完全二分木: すべての葉の深さが等しい (左例ではすべて2)

完全二分木では、頂点数がNのとき深さは $O(\log N)$

二分ヒープ

各頂点 v がキーとよばれる値 key[v] をもつ二分木で、
以下の条件をみたすものを(二分)ヒープという。

・頂点 v の親頂点 p に対して、key[p] ≥ key[v]
・木の高さを h とすると、木の深さ h−1 以下の部分は
　完全二分木であり、木の深さ h の部分は左詰め

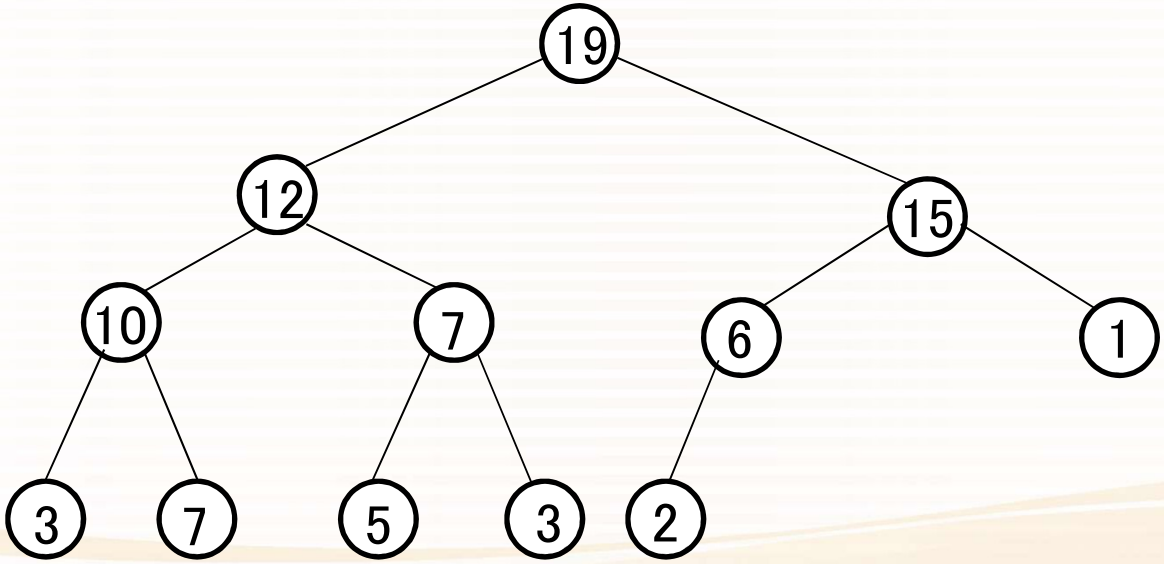


ヒープのクエリ処理

値 x を挿入する: 挿入した後、形を整える。$O(\log N)$

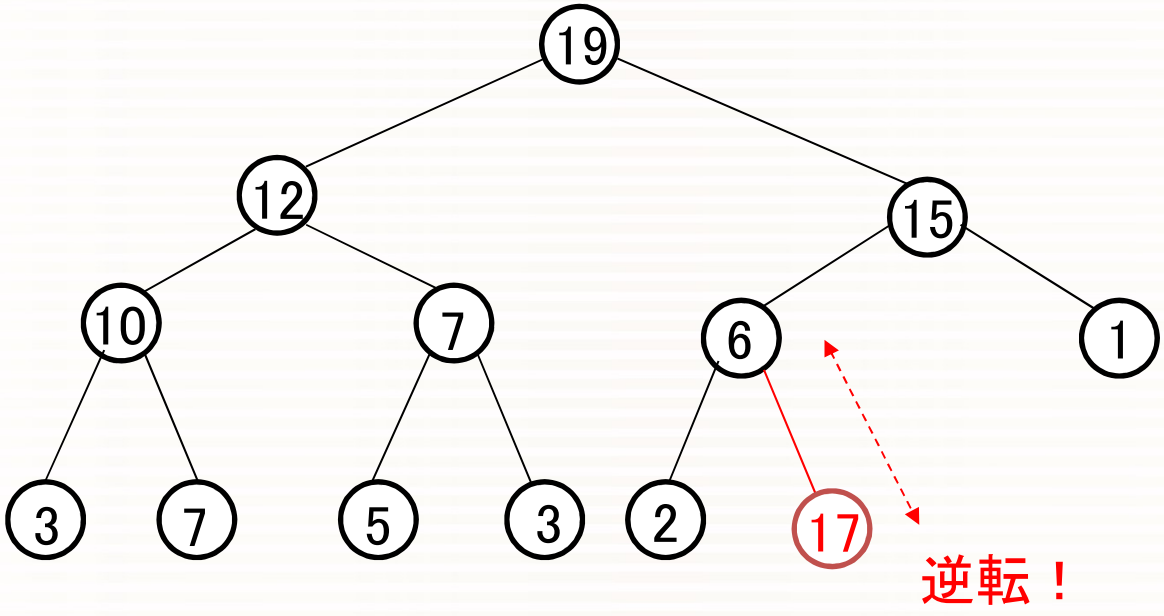
最大値を取得する: 根の値を取得する。$O(1)$

最大値を削除する: 根を削除後、形を整える。$O(\log N)$



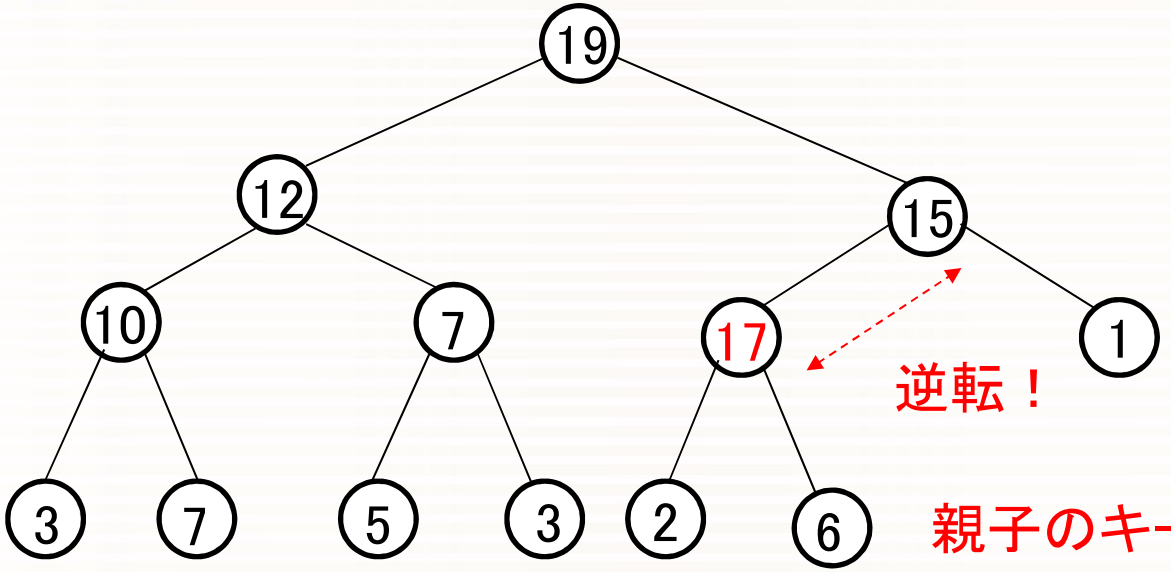
配列による実現: 19, 12, 15, 10, 7, 6, 1, 3, 7, 5, 3, 2

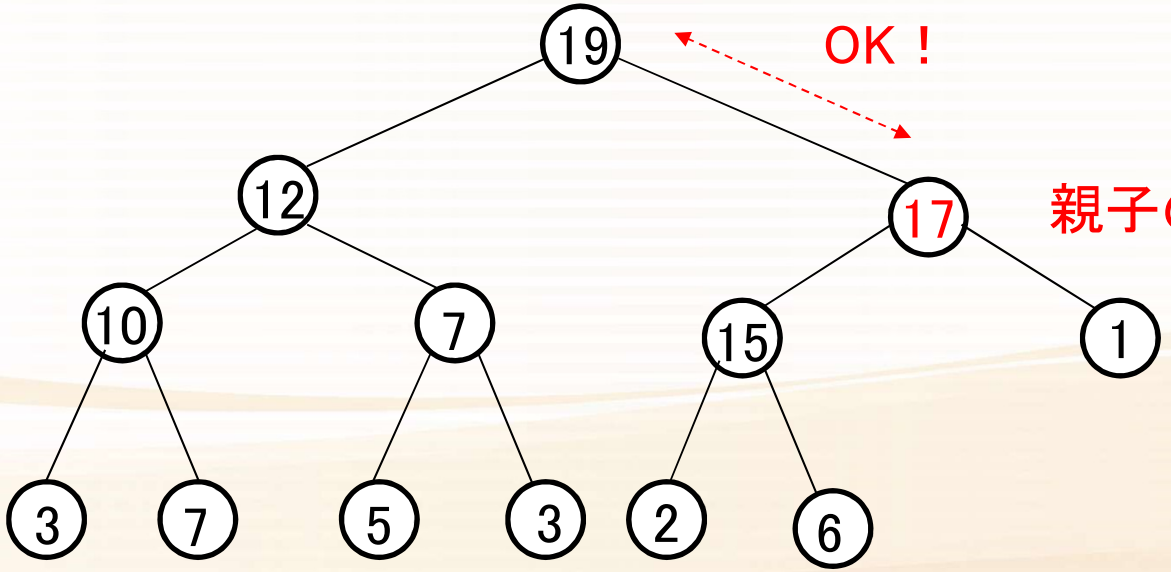
ヒープに値を挿入する



ヒープの最後尾に挿入

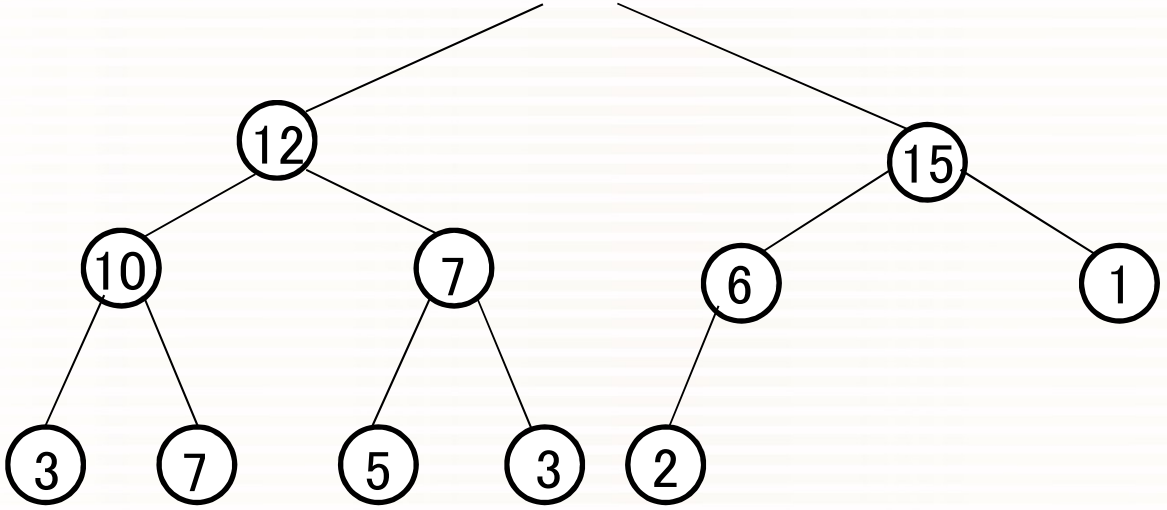
ヒープに値を挿入する

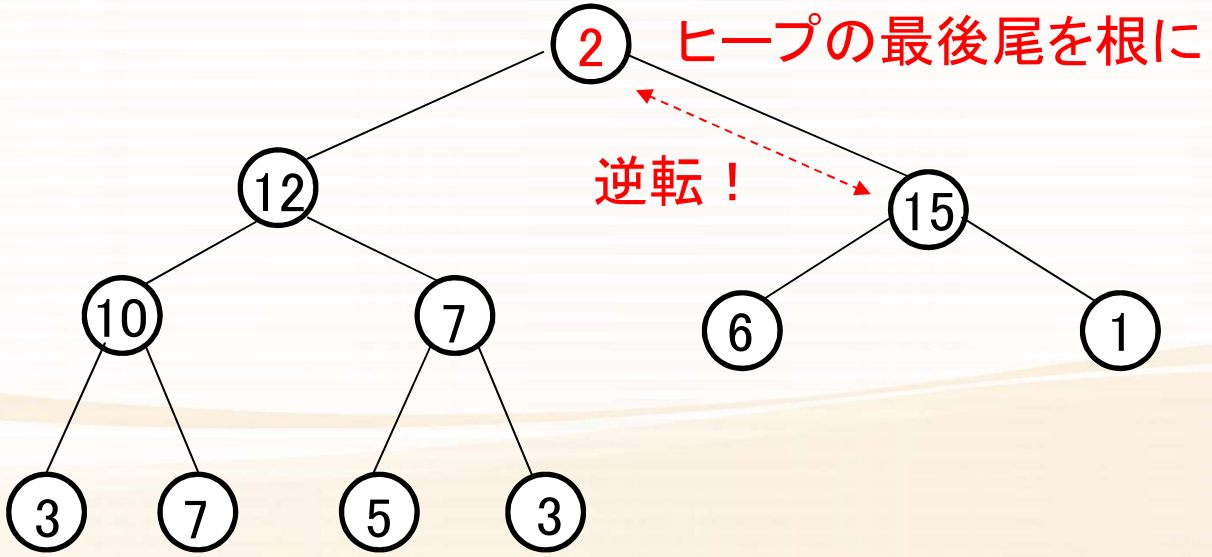




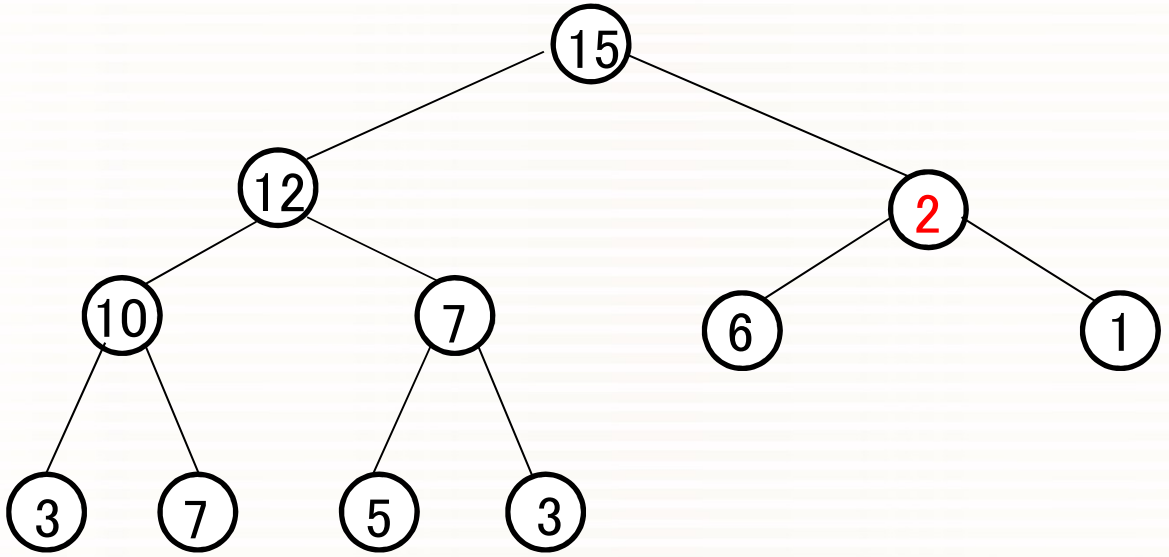
計算量は
木の高さ $O(\log N)$

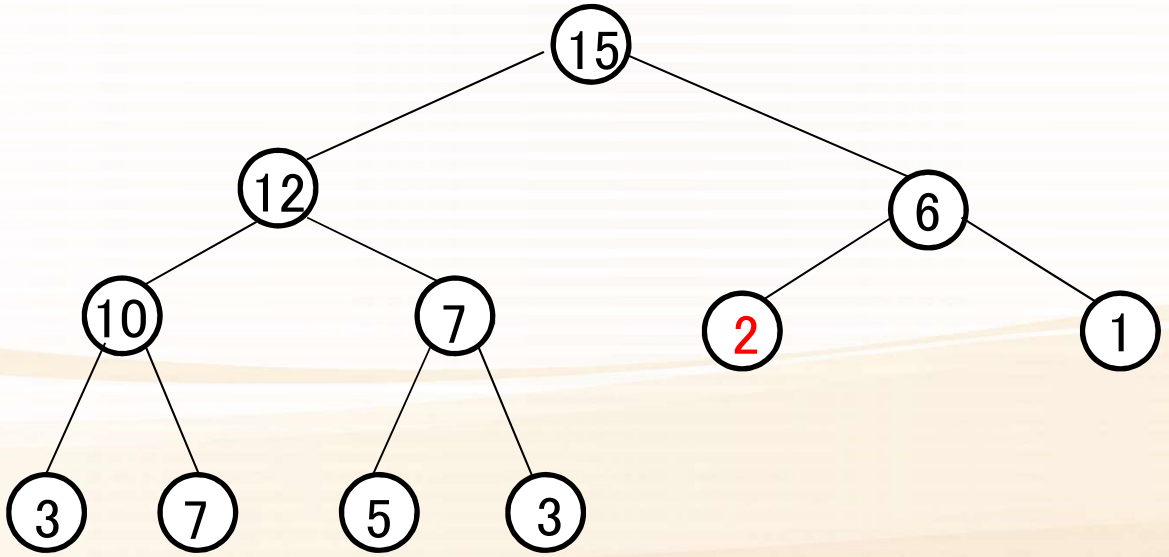
ヒープから最大値を削除する





ヒープから最大値を削除する



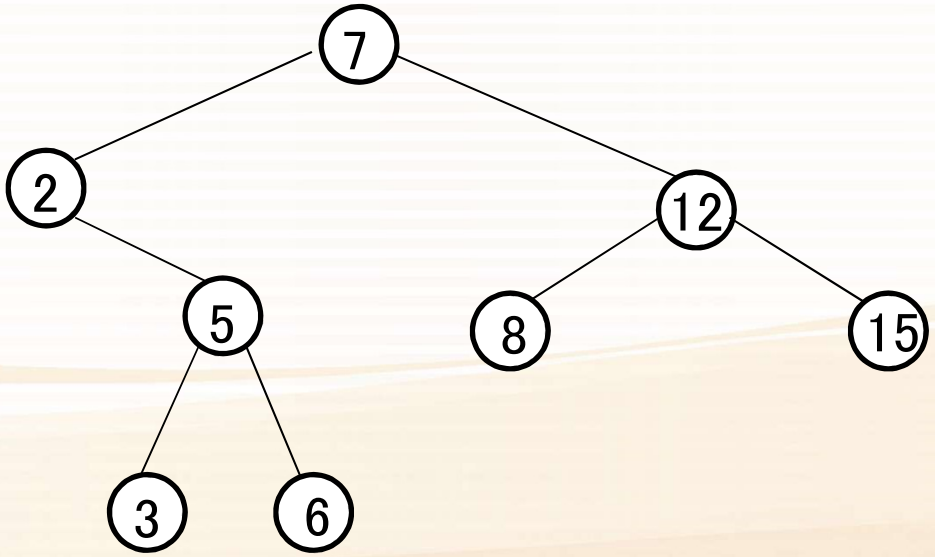


計算量は
木の高さ $O(\log N)$

二分探索木

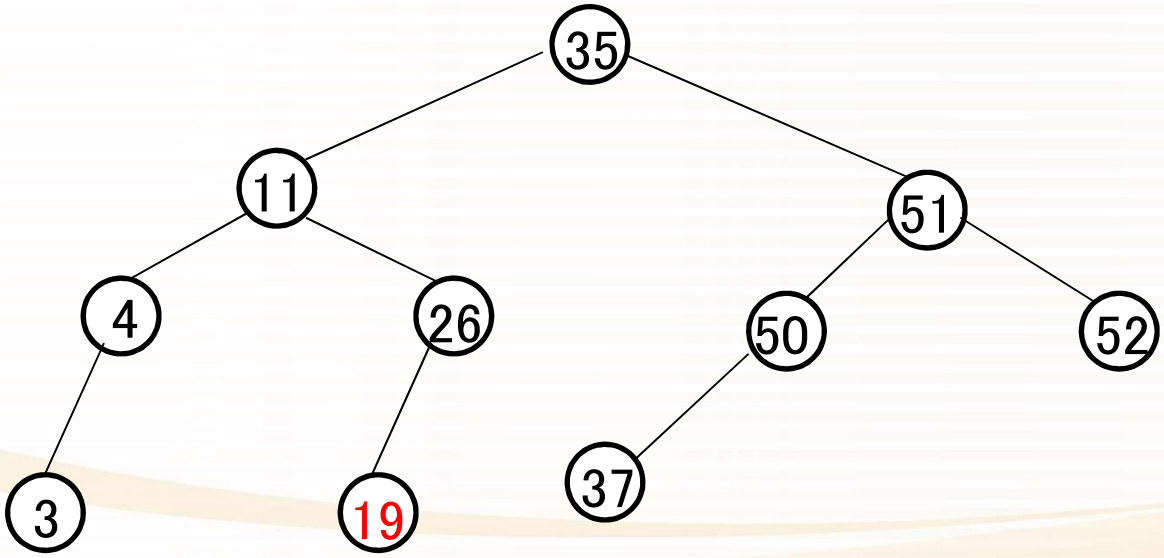
各頂点 v がキーとよばれる値 key[v] をもつ二分木で、
以下の条件をみたすものを二分探索木という。

任意の頂点 v に対し、
・v の左部分木に含まれるすべての頂点 v’ に対して
key[v] ≥ key[v’] が成立し、
・v の右部分木に含まれるすべての頂点 v’ に対して
key[v] ≤ key[v’] が成立する。



二分探索木のクエリ処理

クエリ
- 要素 x をデータ構造に挿入する
- 要素 x をデータ構造から削除する
- 要素 x がデータ構造に含まれるかどうかを判定する



二分探索木

3, 4, 11, 19, 26, 35, 37, 50, 51, 52

（ソート済み配列に対する）
二分探索アルゴリズム