名古屋工業大学 先進セラミックス研究センター　井田 隆　　2023 年 1 月 20 日更新

１４．人工ニューロン
Artificial Neuron

人工ニューロン（人工神経細胞） artificial neuron とは，生物のニューロン（神経細胞）の働きを数学的に単純化してなぞらえるもので，人工神経網 artificial neural network を構成する基本単位です。

生物のニューロンは，外的な刺激や，別のニューロンからシナプス・樹状突起を介して興奮性の刺激や抑制性の刺激を受けて，足し合わせた興奮性の刺激の強さが**閾値** threshold を超えた時に**発火**して，軸索を通じて末端のシナプスにパルス性の信号を出力するものと考えます。

１４−１　人工ニューロンの基本構造

Basic Structure of Artificial Neuron

典型的な人工ニューロンの仕組みは，「複数入力にそれぞれ別の重みを付けて足し合わせた和の値」を，「**伝達函数** transfer function または**活性化函数** activation function と呼ばれる 1 変数の**非線形函数** non-linear function （補足 14.1.A）」に渡し，その函数から戻ってくる値を出力するものです。

人工ニューロンに，**入力** input を $m+1$ 本の信号線でつなぎ，それぞれの**入力信号** signal を x_0 から x_m，それぞれの入力信号へかける**重み** weight を w_0 から w_m で表すとします。ここで x_0 の入力は常に強度 +1 として，**バイアス** bias の入力として扱います。このとき，普通の意味で「入力信号」として扱えるのは x_1 から x_m の m 個の信号です。

複数のニューロンを並列に使うことを前提として，第 k 番目のニューロンの出力 y_k が，

$$y_k = \phi\left(\sum_{j=0}^{m} w_{kj}x_j\right) \tag{14.1.1}$$

と表されるとします。ここでは伝達函数を $\phi(x)$（ϕ はギリシャ小文字のファイ）と表します。バイアスを $b_k = x_0$ として分けて書いて，

$$y_k = \phi\left(b_k + \sum_{j=1}^{m} w_{kj}x_j\right) \tag{14.1.2}$$

としても同じことです。この関係を図示すると，Fig. 14.1.1 のようになります。

バイアスは生物の神経細胞が発火する際の閾値と似た意味を持ち，「バイアス値が b_k 」と言うことは，「閾値が $-b_k$ 」と言うことと同じです（補足 14.1.B）。

「重み付きの加算」と「伝達函数」とを合わせて，**伝播函数** propagation function と呼ばれることもあります（補足 14.1.C）。

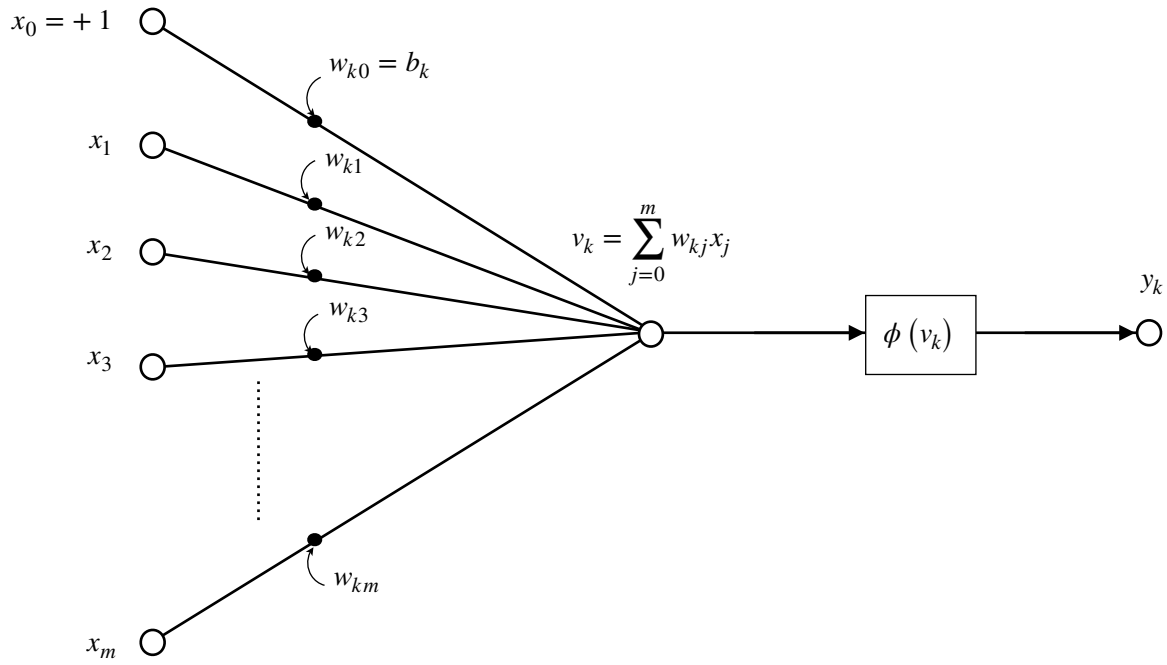

Fig. 14.1.1　人工ニューロンの仕組み。入力と出力の関係

シナプスを介した前段のニューロンからの信号の受け方の変化を，人工ニューロンでは「重み」 $\{w_0, w_1, \cdots, w_m\}$ の変化としてモデル化します。生物が「それぞれのシナプスの結合の強さを変えることで学習する」のだとして，人工ニューロンは「重み $\{w_0, w_1, \cdots, w_m\}$ を変えることにより学習のようなことができる」ことになります。

１４−２　伝達函数　Transfer Function

人工ニューロンで伝達函数として用いられる代表的な函数は以下の３つです（補足 14.2.A）。

(i) **シグモイド函数** sigmoid function

$$\phi_{\text{sigmoid}}(x) = \frac{1}{1 + \mathrm{e}^{-x}} \tag{14.2.1}$$

(ii) **ステップ函数** step function　（Heaviside 函数）

$$\phi_{\text{step}}(x) = \begin{cases} 1 & [0 \leq x] \\ 0 & [x < 0] \end{cases} \tag{14.2.2}$$

(iii)**ランプ関数** ramp function（ReLU (rectified linear unit) 函数；整流器 rectifier ）

$$\phi_{\mathrm{ramp}}(x) = \begin{cases} x & [0 \le x] \\ 0 & [x < 0] \end{cases} \tag{14.2.3}$$

これら３種類の函数のグラフを Fig. 14.2.1 に示します。

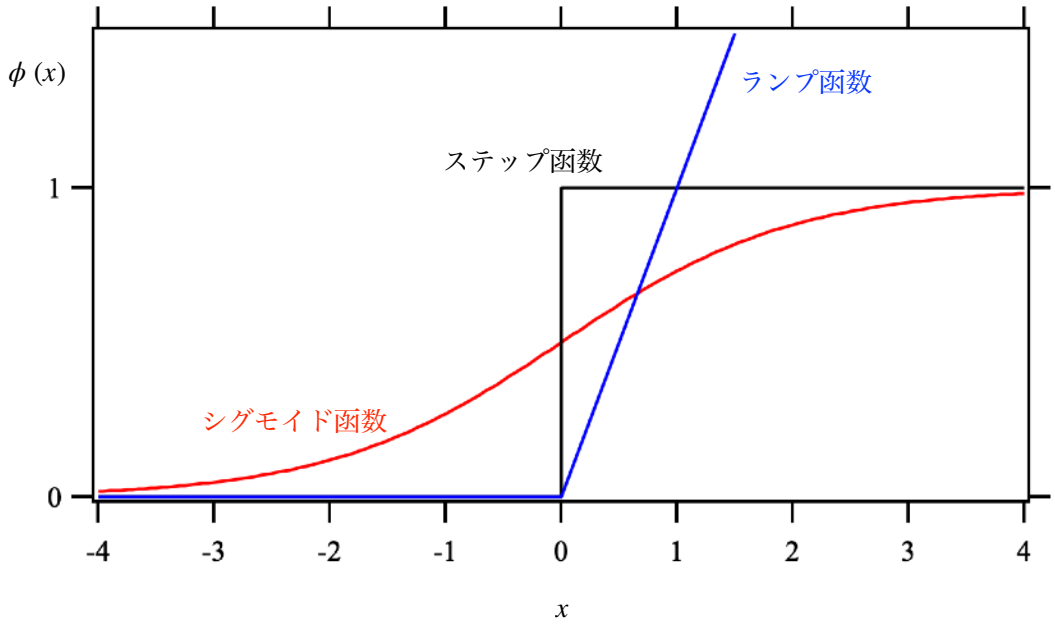

Fig. 14.2.1　３種類の伝達函数。シグモイド函数（赤線）とステップ函数（黒線），ランプ函数（青線）

これら３つの函数のうちどの函数を伝達函数として用いるのが良いのか（補足 14.2.B），別の函数を使うのが良いのかは，あまりはっきりしているわけではありません。実際には，このうちのランプ函数の用いられる例が比較的多いようです。

（補足 14.1.A）線形？線型？（↵）

英語の linear は数学・応用数学の分野で「線形」と訳される場合と「線型」と訳される場合があります。二つの表記は，ほぼ同じ意味で使われ，読み方はいずれも「せんけい」です。例えば linear algebra の訳語として「線形代数」という表記も用いられますが，「線型代数」と言う表記を用いるべきと言う意見もあります。日本の高校数学では「線形」の表記を用います。

漢字の成り立ち方からすると「形」の字は，井戸やその形を意味する「井」を変形した「开」と，刷毛でつけた模様の意味を持つ「彡」とを組み合わせたもので，特に「外から見たかたち」の意味があります。「図形」「形状」「形容」などの語では「形」の文字が使われ「型」の文字は使われません。「円形」「三角形」などにも「形」を使います。

「型」の字は「井」を変形した「开」と，「刀」を変形した「刂」と「土」から成り立ち，元々は土を削って形を作った「鋳型（いがた）」の意味のようです。例えば「模型」「典型」「類型」などには意味の通じる「型」の漢字だけが使われます。

中国語では linear algebra の訳語に「线性代数」の表記が用いられます。簡体字「线」に対応する繁体字は「綫」で，別体字が「線」とされているので，日本語表記にすれば線性代数となります。

線形加速器 linear accelerator（リニア アクセレレータ）の「せんけい」は SPring-8 の web サイトでは「線型」の表記が用いられています。

(⏎)

（補足 14.1.B）バイアスの意味，使い方（↵）

人工ニューロンにはバイアス入力（閾値（いきち）の設定）をできるようにするのが普通のようですが，それが本当に必要なのかは必ずしもはっきりしません。

画像処理の場合に入力データは画素ごとの明るさ（暗さ）を数値化したものなので，普通の意味ではゼロ zero か正 positive（ポジティヴ）になります。そのデータを受ける人工ニューロンに負のバイアス（正の閾値）を設定しないと，普通の重みの付け方では，常に活性化するか常に不活性化した状態になりそうなので，バイアスは付けた方が良さそうに見えます。ただし「重み」には負の値も使えることを前提とすれば，バイアスをつけない（あるいはゼロに固定する）のであっても，重みの付け方によって，異なる画像データを受けた時に異なる挙動を示すようにすることは可能です。

バイアス値は，固定値として扱う場合も可変値として扱う場合もあります。可変値として扱う場合には，他の入力信号値との従属性（依存性）が強いので，同時に最適な値を決めることが難しい場合のあることに注意した方が良いでしょう。

式 (14.1.2) の「重み付きの和」の部分の

$$v_k = b_k + \sum_{j=1}^{m} w_{kj} x_j \tag{14.1.B.1}$$

は「**アフィン変換** affine transformation（アフィン トランスフォーメイション）」と呼ばれる操作に相当し，(w_{kj}) で図形の拡大・縮小・回転・反転・鏡映などの変形に対応する操作，(b_k) で並進（平行移動）に対応する操作を表すことができます。同じことを行列を使って表現すれば，

$$\begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} + \begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1m} \\ w_{21} & w_{22} & \cdots & w_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n1} & w_{n2} & \cdots & w_{nm} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix} \tag{14.1.B.2}$$

と書けて，さらに同じことを式 (14.1.1)

$$v_k = \sum_{j=0}^{m} w_{kj} x_j \tag{14.1.1}$$

のように表現することは，行列を使って

$$\begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix} = \begin{pmatrix} b_1 & w_{11} & w_{12} & \cdots & w_{1m} \\ b_2 & w_{21} & w_{22} & \cdots & w_{2m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_n & w_{n1} & w_{n2} & \cdots & w_{nm} \end{pmatrix} \begin{pmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix} \tag{14.1.B.3}$$

と表現するのと同じことです。

式 (14.1.1) あるいは式 (14.1.B.3) のように表現することの主な利点は，「見た目がスッキリすること」にあると思われます。 (↵)

（補足 14.1.C）伝播？伝搬？ (↵)

伝播の意味で伝搬と書かれたり，英語の propagation が伝搬と訳される例も多いようですが，「『伝播』の正しい読み方がわからず『でんぱん』と誤読」→「『でんぱん』を『伝搬』と誤記」→「『伝搬』の方が読みやすいし，『でんぱ』だと『電波』と紛らわしくなる」などの事情から，「伝搬」と言う誤用や誤訳が広まったと言う説が有力なようです。英語の propagation は『搬』の字が持つ意味を含まないので，正しく伝播と訳すようにした方が良いでしょう。

誤訳から発生した「伝搬」と言う造語の「搬」の字の意味は，この記事で「伝達函数」としている transfer function の transfer の方に近く，紛らわしくなってしまいます。 (↵)

（補足 14.2.A）シグモイド函数・ステップ函数・ランプ函数 (↵)

シグモイド函数は，

$$\phi_{\text{sigmoid}}(x) = \frac{1}{2}\left(1 + \tanh\frac{x}{2}\right) \tag{14.2.A.1}$$

と表すこともできます。ここで $\tanh x$ は双曲正接函数 hyperbolic tangent function と呼ばれる初等函数であり，

$$\tanh x \equiv \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{14.2.A.2}$$

と定義されます。ただし，式 (14.2.A.2) の定義式をそのまま使って，普通の数値計算システムで計算すると $x \to 0$ の時に数値的な精度が落ちるので，多くの計算システムでは $x \to 0$ の時の精度も保証する「双曲正接函数 $\tanh x$ のライブラリ」を直接呼び出して使えるようになっています。一方で，シグモイド函数を計算するために双曲正接函数を使った式 (14.2.A.1) の形式を用いるのが良いとは限りません。式 (14.2.A.1) の形式で計算すると，$x \to -\infty$ の時に桁落ちによって精度が落ちてしまいます。

ステップ函数は「階段函数」と訳されることも多いようですが，英語の step には階段の「段」と言う意味はあっても「階段」と言う意味はないので，ここでは「ステップ函数」と呼ぶことにします。グラフを描いた時に階段状になるのは，むしろ床函数 floor function $\lfloor x \rfloor$ （小数点以下切り下げ）や天井函数 ceiling function $\lceil x \rceil$ （小数点以下切り上げ）です。床函数・天井函数という名称も紛らわしいのですが，それはそれとして，ステップ函数を「階段函数」と呼ぶのはやめた方が良いと思います。

ランプ函数は，

$$\phi_{\text{ramp}}(x) = \max(0, x) \tag{14.2.A.3}$$

と表現することもできます。ここで $\max(a, b)$ は 2 つの変数 a，b のうち大きい方の値（最大値 maximum）を返す函数です。この函数が使える処理系であれば，式 (14.2.A.3) の表現を使えば 1 行でこの函数を使えるので，少し楽です。この函数は「ReLU 函数」と呼ばれることも多いようですが，元々「ランプ函数」という名称があるのに，全く同じ意味のことを違う名前で呼ぶのは好ましくないと思われるので，ここでは「ランプ函数」と呼ぶことにします。 (↵)

（補足 14.2.B）伝達函数としてのシグモイド函数・ステップ函数・ランプ函数（↵）

生物のニューロンの「刺激の強さが閾値（いきち）を超えると発火する」挙動を再現するためだけなら，人工ニューロンの伝達函数としては，最も簡単な「ステップ函数」を使えば良いように思えるかもしれません。それで済むのなら「伝達函数」と言っても実際には計算らしいことをしなくて良いので，少し速く動作する人工ニューロンになるかもしれません。

しかし，現実の生物のニューロンの出力はパルス性の信号なので，出力信号では「時間あたりのパルス数（パルス速度）」を変えれば出力の「強さ」を伝えることができます。一方で，現在の典型的な「人工ニューロン」や「人工神経網」はパルス速度で強度を表現するものではないので，「任意の強さの出力」もできるようにしたければ，ステップ函数の次に簡単な「ランプ函数」を使えば良いでしょう。

ところが，現実のパルス性の信号は，パルスの時間幅 τ（ギリシャ小文字のタウ）が有限なので，その逆数 $1/\tau$ のパルス速度を超えることは不可能です。パルス信号システムの応答が，有限なパルス幅あるいは有限の応答時間の影響をどのように受けるかは「数え落とし missing count（ミッシング カウント）」「待ち行列理論 queueing theory（キューイング セオリー）」（待ち行列理論の「行列」とは線形代数の行列 matrix（マトリクス）と言う意味ではなく，人気ラーメン店の前などで待つ客の行列 queue（キュー）の意味）などとしても詳しく調べられていて，代表的なモデルで双曲線的あるいは指数函数的に飽和する（頭打ちになる）挙動を示すことも知られています。このことからすれば，それと似た挙動を示す「シグモイド函数」を伝達函数として用いれば，現実のニューロンの働き方に近くなるかもしれません。そして，このことには注意すべき点が含まれています。

「生物（特にヒト）の神経のシステムが一番効率の良い情報処理システムのはずだ」と思いがちな人は実際に多く，このことの背景に「ヒトが機械より優位な存在であってほしい」と願う心理があります。そして，それを根拠にしてシグモイド函数にこだわるとしたら，それはもはやカルト cult（カルト）（異端的な宗教集団）でしょう。生物のシステムを真似するのが一番良いとは限りません。伝達函数としては，目的や利用できる環境，入手しうるデータの性格に応じて「良さそうな函数」を選ぶのが現実的です。（↵）