工学実験 (2 年次) テキスト

［課題番号 12］ニューラルネットワークによるモデル化

担当: 生産システム工学系
システム創製研究室　（林、清水）
場所：情報通信実験棟　1F-107

1. ニューラルネットワーク

はじめに

今日、私たちが用いているコンピュータは、数値計算においては優秀な情報処理装置であるが、パターン認識に代表されるような多分に定性的であいまいな判断を苦手としている。一方、人間の脳は定形的で大量の数値計算においてはコンピュータに劣るが、パターン認識のような不定形で総合的判断においては優秀な情報処理装置といえる。（人工）ニューラルネットワーク (Neural Network; NN) はこのような脳における神経回路の動きを模した情報システムの総称であり、近年その基礎研究の発展とともに各種工学問題におけるモデル化、パターン認識・分類、制御・最適化などに応用されてきている。

本実験では、こうした NN に着目し、相関データのモデル化に適用することを試みる。NN の基本的な概念をまず学習する。次いで、NN によるモデル化の特徴について相関データのモデル化手法としてこれまで代表的な最小二乗法との比較を通じて、数値実験により考察する。

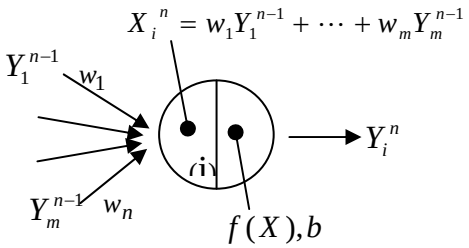


Fig. 1 ニューロンの構造

1.1 人工ニューラルネットのモデル化

NN は **Fig.1** に示すような人工ニューロンから構成される。図中央の円形がニューロンの一単位で、これに左側から入力信号（他のニューロンからの出力信号）が（刺激として）入り、各ニューロンの特性に応じた出力信号に変換される。そして、これが新たに別のニューロンへの入力信号となるといった連鎖によって最終的な出力が得られる。ニューロンが入力ユニットから出力ユニットまですべて順方向のみに結合され、フィードバック結合を持たないようなモデルを階層型ニュ

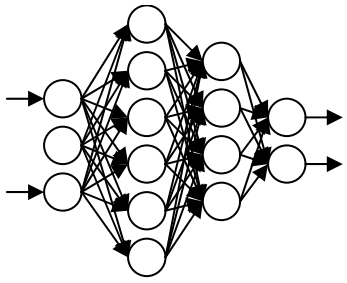


Fig. 2 階層型 NN

ーラルネットワークという（**Fig.2**）。一方、ユニット間の結合が必ずしも順方向のみとは限らない相互結合型ニューラルネットワークもある（**Fig.3**）。

以下では広く用いられている、Fig.2 に示す階層型ニューラルネットワークを取り上げモデル化の要点を述べる。すなわち第 n 層のニューロン i への入力 X_i^n は第 n-1 層のニューロン j の出力値 Y_j^{n-1} の重み付け総和として(1)式で与えられる。一方、第 n 層のニューロン i の出力 Y_i^n は入力値を基に伝達関数 f(x) としきい値 b_i によって(2)式のように計算される。

$$X_i^n = \Sigma_j W_{ij}^{n,n-1} Y_j^{n-1} \quad (n = 2, \cdots, N) \tag{1}$$

$$Y_i^n = f(X_i^n - b_i^n) \quad (n = 1, \cdots, N) \tag{2}$$

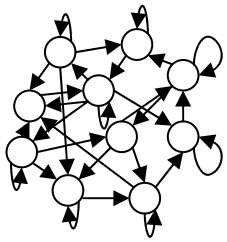


Fig. 3 相互結合型 NN

ここで、$W_{ij}^{n,n-1}$ は、n-1 層のニューロン j と n 層のニューロン i 間の結合における相対的な重要度を表わす係数で、この値が大きいほど結合関係が強く、そのニューロンにとって支配的な入力（刺激）といえる。また N は全階層数である。

ニューロンに貯えられた刺激に応じて出力を決めるものが伝達関数である。その代表的なものが **Fig. 4** に示すシグモイド関数で、生物の成長曲線としても知られている S 字カーブの曲線となる。実際の出力は(2)式で計算されるため、しきい値 b_i 分だけ右に平行移動したものとなる。この他、**Fig. 5** に示すような伝達関数が知られている。

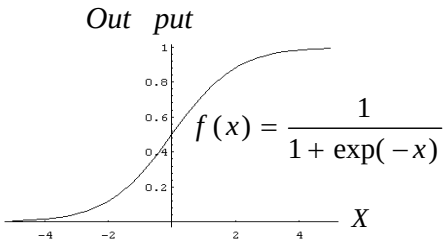
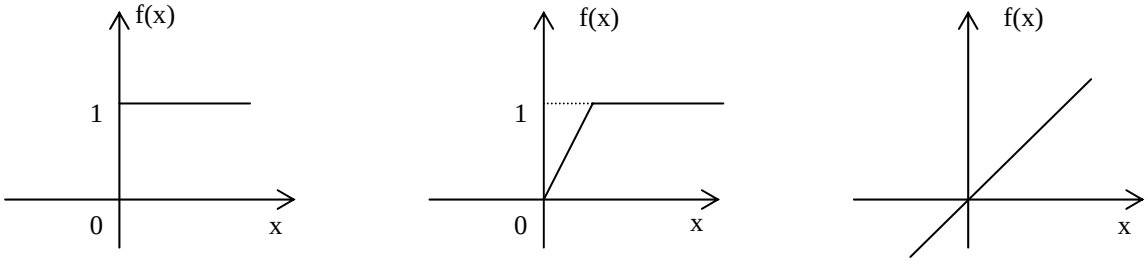


Fig. 4 伝達関数（シグモイド関数）



a. しきい関数　　b. 区分線形関数　　c. 恒等関数

Fig.5 典型的な伝達関数

1.2 ニューロンの学習

ニューラルネットの構造（例えば、3 層の階層型でそれぞれの層のニューロンの数が決まっている）が与えられている時、それが与えられた刺激に対して正しく反応するためには、人間の脳の場合と同様、何らかの学習が必要となる。これは、NN の場合、結合荷重 W やしきい値 b といったパラメータを調整することに相当する。このことを学習といい、学習によってネットワークを実効あるものにすることができる。人の脳の場合も、学習によって刺激を受けたニューロンの感受性は高まり、刺激の伝播をつかさどるシナプスは太くなることが知られている。

学習の方法は大きく 2 種類に分けられる。その一つは、望ましい出力（教師信号という）が外部から与えられて学習をするものであり、これを教師あり学習という。もう一方は、教師信号が与えられずに入力信号の統計的な性質からニューロンが自分で学習するものであり、これを教師なし学習という。

階層型のニューラルネットワークには教師データを用いた誤差伝播法（BP： Back Propagation）と呼ばれる学習アルゴリズムが一般に用いられる。これは、ニューラルネットワークからの出力信号 $\mathbf{Y}^N$ と教師信号 $\mathbf{d}$ を比較し、この差が小さくなるように結合荷重 W を修正するものである。実際には、出力誤差の二乗和である(3)式を最小とするように結合加重が決められる。

$$r = \Sigma_i (Y_i^N - d_i)^2/2, \ (i = 1, \cdots, tr) \tag{3}$$

ここで tr は教師データ d_i の総数を表わす。

この最小化アルゴリズムを与えるために、r の W に関する傾きを計算すると

$$(\partial r / \partial W_{ij}^{n,n-1}) = (\partial r / \partial X_i^n)(\partial X_i^n / \partial W_{ij}^{n,n-1}) = (\partial r / \partial X_i^n) Y_i^{n-1} \tag{4}$$

となる。さらに

$$\begin{aligned}(\partial r / \partial X_i^n) &= \Sigma_k (\partial r / \partial X_k^{n+1})(\partial X_k^{n+1} / \partial Y_i^n)(\partial Y_i^n / \partial X_i^n) \\ &= \Sigma_k (\partial r / \partial X_k^{n+1}) W_{ki}^{n,n+1} f'(X_i^n)\end{aligned} \tag{5}$$

であり、ここで、$(\partial r / \partial X_i^n) = \delta_i^n$ とおくと、(5)式は、

$$\delta_i^n = f'(Y_i^n) \Sigma_k W_{ki}^{n+1,n} \delta_k^{n+1} \tag{6}$$

と表現できる。ただし最終出力層 N では、$\delta_i^N = (d_i - Y_i^N) f'(X_i^N)$ である。ここで(6)式を(1)式と比較してよく観察してほしい。そしてここで、δ_i^n を n+1 層のニューロン i から戻される学習信号とみなせば、学習データと NN の出力との誤差を入力として、出力層から入力層へ丁度逆方向の計算を行なう形をとっていることが知れる。これよりこうした探索法に対して誤差伝播(BP)の名がある。

(4)式に(5)、(6)式を用いて計算される傾きを用いる W の探索アルゴリズム（一般に山登り法と呼ばれる）において、振動を減らしながら、学習の収束を早めるために実際には次式のように修正量が決められる。

$$\Delta W_{ij}^{n,n-1}(t) = \eta\ \delta_i{}^{n} Y_j^{n-1} + \alpha\ \Delta W_{ij}^{n,n-1}(t-1) \qquad (7)$$

ここで、α と η はそれぞれ慣性定数、学習定数と呼ばれるパラメータを、そして t は探索の繰り返し数を表わす。α は通常の山登り法における探索ステップ幅に相当し、収束の早さに関係する。一方、η は解の振動の調整に用いられる。

1.3 学習効果と汎化能力

与えられた学習データについては、上での NN の学習法から十分正確な出力が得られることが期待できる。しかし、「未学習の入力データに対して（学習済みの NN から ）どの程度正確な出力が得られるか」という問題は、NN の汎化能力としてその性質の中で特に重要な問題の一つである。このために一般に採られる方法は、学習データ全てを学習に用いずに、その一部を残しておきそれらを検証用に用いようとするものである。すなわち学習誤差を(3)式で計算される値の平均値（r/tr）で評価しながら、同時に検証用データ $\underline{d}_i$(このデータ数を ts とする)に対する推定誤差を(8)式で評価し、これを汎化能力の判定に用いようとするものである。

$$r_s = \Sigma_i (\underline{Y}_i^N - \underline{d}_i)^2/2/ts \qquad (8)$$

NN の学習において、学習と汎化に関してよく観察される現象を **Fig.6** に示す。これは学習データと検証データの誤差関数値の履歴をプロットしたものである。学習データは学習が進むと学習誤差が減少する。しかし検証データはある学習誤差を境に学習誤差が増加に転じることがある。これは過剰学習として知られる現象であり、学習が進みすぎると、汎化能力が下がってしまうもので、応用面からみれば不適切な現象の一つである。

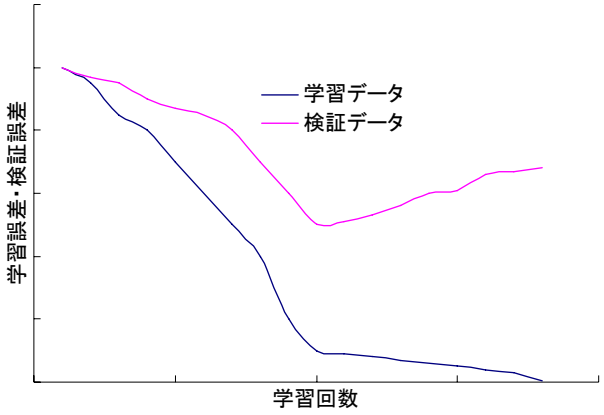


Fig.6 学習回数と学習・検証誤差の関係

以上のような階層型ニューラルネットワークの特徴としては

①入出力関係を学習させるだけで非線形性の強い写像をネットワーク内の結合係数として構築ができるという便利性

②入力変数すなわち入力層ユニット数の増加に容易に対応できる拡張性
③学習パターンを滑らかに内挿し、未学習の入力に対しても妥当な出力値を出力できる汎化能力
④学習が終了してしまえば積和演算を行うだけで出力が得られるという応用時の迅速性など
があげられる。

これらの特徴（特に①、③）は非線形性が非常に強く、未知な構造を同定するのに非常に適している。また、このような特徴を有することから NN は臭い、うまい、好ましいといった人間の感性を同定するのにも応用されてきている。

1.4 NN による OR と XOR の学習

論理和 OR($y = x_1 + x_2$)、および排他的論理和 XOR($y = x_1 + x_2$)はそれぞれ **Table 1, Table 2** の関係を満たす演算である。

Table 1 OR

x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	1

Table 2 XOR

x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	0

これを図的に表現すれば、それぞれ **Figs. 7(a), (b)** のように表わされる。このとき Fig. 7(a)においては y=0 と 1（白丸と黒丸）を分離するのに直線 ab で十分であるが、Fig. 7(b)においては複雑な曲線 AB が必要になることがわかる。このことから XOR のモデル化の方が OR よりはるかに難しいこと、また XOR のモデル化自体けっして単純でないことが推測される。

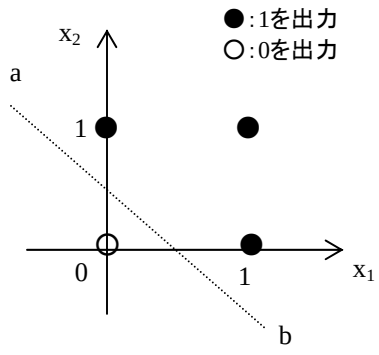


Fig.7(a) **線形分離可能な例**(OR)

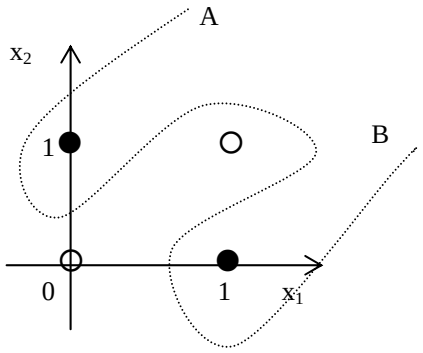


Fig.7(b) **線形分離可能でない例**(XOR)

ここでは **Fig. 8** に示すような入力数 2、中間ユニット数 2、出力数 1 の 3 層の NN を与え、NN によるモデル化を試みる。このとき、学習係数と慣性係数は同じにしたとき、両者の学習時の誤差の収束の様子を **Fig. 9** に示す（ここで横軸は(7)式の修正の反復回数を縦軸は、平均二乗平方誤差(Root Mean Square Error: RMSE)を表わす。これは(3)式または(8)式の平方の正値）。これからも OR 問題の方が早く（より容易に）モデル化できること、そして XOR 問題も NN によって、うまくモデル化できることがわかる。以

上の簡単な例から NN が現象の線形、非線形にとらわれず効果的なモデル化手法であることがわかる。またこの際、モデルの構造（関数形）を必要としないモデル化手法であることにも注目しておいてほしい。

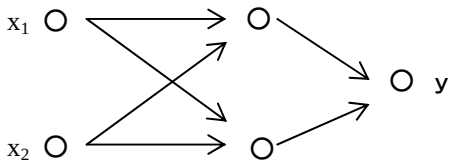


Fig.8 用いるNN構造

2. 最小二乗法によるモデル化

最小二乗法も入力 x と出力 y に関する観測データの組, (x^i、y^i), (i=0、･･･、n)を用いて、x と y の関数関係をモデル化するためによく使われる。そのデータはふつう現象を観測し、変化を定量的に測定することから得られる。最小二乗法によるモデル化は、データを取得するまでの種々の段階で含まれる誤差によるデータのバラつきをならす効果をもっており、ある程度までこれらの誤差が打ち消された結果が得られる。これは最小二乗法の最も重要な特徴であり、補間法にはない側面である。補間法で用いられる補間多項式は全てのデータに合うように作られるから、データの含む誤差はそのまま保持される。

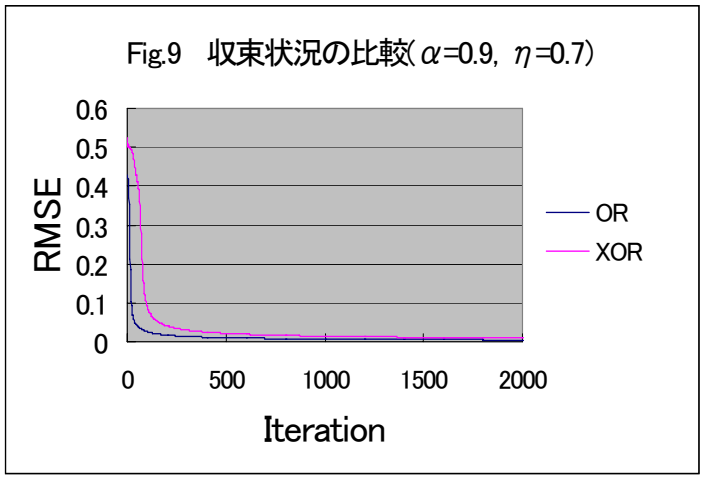


Fig.9 収束状況の比較(α=0.9, η=0.7)

未知の関数、y=f(x)をある既知の関数、$f_0(x)$、$f_1(x)$、･･･、$f_m(x)$の線形結合からなる h(x)で近似することを考える。

$$h(x)=a_0f_0(x)+a_1f_1(x)+ \cdots a_mf_m(x) \qquad (9)$$

このとき、n＞m であるような測定値、(x_i、y_i)(i=0、･･･、n)を用いて、h(x)が f(x)にできるだけ合うように，未知の係数 a_0，a_1，･･･，a_m を決めることがここでの問題となる。この一つの解決策は、近似関数の値 $h(x_i)$と x_i に対応するデータの値 y_i との差の平方和、すなわち誤差の二乗和をパラメータ **a** に関して最小化する事によって与えられる。

$$\text{Min } Q= \Sigma_{i=0} r_i^2 = \Sigma_{i=0} (h(x_i) - y_i)^2$$
$$= \Sigma_{i=0} (\Sigma_{j=0} a_j f_j(x_i) - y_i)^2 \qquad (10)$$

この実際の解法手順を、以後ベクトル表現によって示す。すると上式は、

$$\text{Min } Q= \| r \|^2 = (\mathbf{Fa-y})^T(\mathbf{Fa-y}) \qquad (11)$$

のように簡潔に表現できる。ここで **F** は、

$$\mathbf{F} = \begin{bmatrix} f_0(x_0), \cdots, f_m(x_0) \\ \cdots, \cdots, \cdots \\ f_0(x_n), \cdots, f_m(x_n) \end{bmatrix} \quad (12)$$

を、T はベクトルや行列の転置を表す。

ところで良く知られているように，Q を **a** に関して最小化するときの必要条件は、Q の **a** に関する停留(極値)条件、$(\partial Q/\partial \mathbf{a})^T=\mathbf{0}$ として与えられる。今の場合、

$$(\partial Q/\partial \mathbf{a})=2(\mathbf{Fa}-\mathbf{y})^T\mathbf{F} \quad (13)$$

であるから、

$$(\partial Q/\partial \mathbf{a})^T=2\mathbf{F}^T(\mathbf{Fa}-\mathbf{y})=\mathbf{0} \quad (14)$$

より、通常，正規方程式と呼ばれる

$$\mathbf{F}^T\mathbf{Fa}=\mathbf{F}^T\mathbf{y} \quad (15)$$

が得られる。ここで $\mathbf{F}^T\mathbf{F}$ が正則なら、逆行列（添字-1 で示す）が存在し

$$\mathbf{a}=(\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\mathbf{y} \quad (16)$$

と求められる。

ところで，近似関数 h(x)を構成する既知の関数 $f_k(x)$を、$f_k(x)=x^k$ とすれば、

$$h(x)=a_0+a_1x+\cdots+a_mx^m \quad (17)$$

と表現でき、上記と同様の取扱は、統計解析でよく知られている回帰分析に帰結する。

(例 2.1)

次のデータに、1 次の最小二乗多項式を合わせてみよ。

x	-1	0	1	2
y	0.75	2.25	3.25	3.75

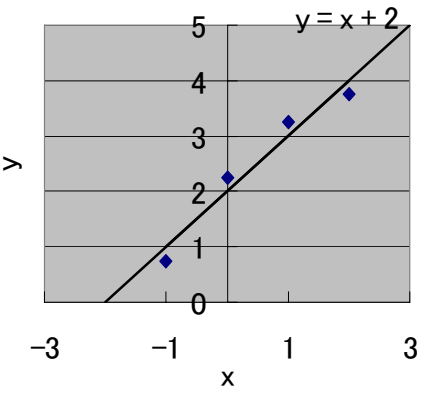
Fig.10 最小自乗の結果

(解)　$y = a_0+a_1x$ の a_0 ,a_1 を求める。

$$\mathbf{F}^T = \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 0 & 1 & 2 \end{bmatrix} \qquad \mathbf{F}^T\mathbf{F} = \begin{bmatrix} 4 & 2 \\ 2 & 6 \end{bmatrix}$$

$$\mathbf{F}^T\mathbf{y} = (10, 10)^T \qquad 20\,(\mathbf{F}^T\mathbf{F})^{-1} = \begin{bmatrix} 6 & -2 \\ -2 & 4 \end{bmatrix}$$

$$(a_0, a_1)^T = (\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\mathbf{y} = (2, 1)^T$$



より、結局、y=2+x となり、得られたモデルは **Fig.10** に示すように所与のデータの誤差を平均化してモデル化（RMSE=0.25）していることがわかる。

（Appendix 1）ベクトル・行列演算に関する用語と表記法

ベクトル・行列に関する基本的な用語や表記法について以下に整理しておく。

（和と差）

$$\mathbf{A} = (\mathbf{a}_1,\cdots,\mathbf{a}_n) = \begin{bmatrix} a_{11},\cdots, a_{1n} \\ \cdot\ ,\cdots,\ \cdot \\ a_{m1},\cdots,a_{mn} \end{bmatrix}, \qquad \mathbf{B} = (\mathbf{b}_1,\cdots,\mathbf{b}_n) = \begin{bmatrix} b_{11},\cdots,b_{1n} \\ \cdot\ ,\cdots,\ \cdot \\ b_{m1},\cdots,b_{mn} \end{bmatrix}$$

とするとき、

$$\mathbf{A}\pm\mathbf{B}=(\mathbf{a}_1\pm\mathbf{b}_1,\cdots,\mathbf{a}_n\pm\mathbf{b}_n) = \begin{bmatrix} a_{11}\pm b_{11},\cdots, a_{1n}\pm b_{1n} \\ \cdot\ ,\ \cdots,\ \cdot \\ a_{m1}\pm b_{m1},\cdots, a_{mn}\pm b_{mn} \end{bmatrix} \qquad (18)$$

となる。これについて以下の関係が成立する。

$$\mathbf{A}\pm\mathbf{B}=\mathbf{B}\pm\mathbf{A} \qquad (19)$$

$$(\mathbf{A}\pm\mathbf{B})\pm\mathbf{C}=\mathbf{A}\pm(\mathbf{B}\pm\mathbf{C}) \qquad (20)$$

$$a(\mathbf{A}\pm\mathbf{B})=a\mathbf{A}\pm a\mathbf{B} \qquad (21)$$

$$(a\pm b)\mathbf{A}=a\mathbf{A}\pm b\mathbf{A} \tag{22}$$

(積)

$$\mathbf{A}=(a_1,\cdots,a_n)=\begin{bmatrix} a_{11},\cdots,a_{1n} \\ \cdot,\cdots,\cdot \\ a_{m1},\cdots,a_{mn}\end{bmatrix},\quad \mathbf{B}=(\mathbf{b}_1,\cdots,\mathbf{b}_m)=\begin{bmatrix} \underline{\mathbf{b}}_1^{\,T} \\ \cdot \\ \underline{\mathbf{b}}_k^{\,T}\end{bmatrix}=\begin{bmatrix} b_{11},\cdots,b_{1m} \\ \cdot,\cdots,\cdot \\ b_{k1},\cdots,b_{km}\end{bmatrix}$$

とするとき,

$$\mathbf{BA}=\begin{bmatrix} (\underline{\mathbf{b}}_1,\mathbf{a}_1),\cdots,(\underline{\mathbf{b}}_1,\mathbf{a}_n) \\ \cdot,\cdots,\cdot \\ (\underline{\mathbf{b}}_k,\mathbf{a}_1),\cdots,(\underline{\mathbf{b}}_k,\mathbf{a}_n)\end{bmatrix}=\begin{bmatrix} \Sigma_j^{\,m}\, b_{1j}\,a_{j1}\,,\ \Sigma_j^{\,m}\, b_{1j}\,a_{j2},\cdots,\ \Sigma_j^{\,m}\, b_{1j}\,a_{jn} \\ \cdot,\cdots,\cdot \\ \Sigma_j^{\,m}\, b_{kj}\,a_{j1}\,,\ \Sigma_j^{\,m}\, b_{kj}\,a_{j2},\cdots,\ \Sigma_j^{\,m}\, b_{kj}\,a_{jn}\end{bmatrix} \tag{23}$$

従って、任意の大きさの行列、**A**、**B** に対して常に積が定義されるわけでない。

(正方行列)　列と行の数が等しい行列

(単位行列)　正方行列で対角要素のみが 1 で、残りは全て 0 の行列で、**I** と表す。

(転置行列)　行列の列と行を入れ換えたもので、$\mathbf{A}^T$ のように表し、以下が成立する。

$$\mathbf{A}=\begin{bmatrix} a_{11},\cdots,a_{1n} \\ \cdot,\cdots,\cdot \\ a_{m1},\cdots,a_{mn}\end{bmatrix}=(\mathbf{a}_1,\cdots,\mathbf{a}_n)=\begin{bmatrix} \underline{\mathbf{a}}_1^{\,T} \\ \cdot \\ \underline{\mathbf{a}}_m^{\,T}\end{bmatrix} \tag{24}$$

$$\mathbf{A}^T=\begin{bmatrix} a_{11},\cdots,a_{1m} \\ \cdot,\cdots,\cdot \\ a_{n1},\cdots,a_{nm}\end{bmatrix}=(\underline{\mathbf{a}}_1,\cdots,\underline{\mathbf{a}}_m)=\begin{bmatrix} a_1^{\,T} \\ \cdot \\ a_n^{\,T}\end{bmatrix} \tag{25}$$

$$(\mathbf{A}+\mathbf{B})^T=\mathbf{A}^T+\mathbf{B}^T \tag{26}$$

$$(\mathbf{AB})^T=\mathbf{B}^T\mathbf{A}^T \tag{27}$$

(階数)　線形変換、**y**=**Ax** において **x** が n 次元空間全体をうごくとき、**y** のうごく範囲、S の次元 r を行列 **A** の階数(Rank)という。

(正則)　**A** が n 次元の正方行列であって、Rank　**A**=n のとき、**A** は正則(regular)であるという。正則でない行列は特異(singular)と呼ばれる。

(逆行列)　$\mathbf{A}\mathbf{A}^{-1}=\mathbf{A}^{-1}\mathbf{A}=\mathbf{I}$ を満たす $\mathbf{A}^{-1}$ を **A** の逆行列と呼ぶ。逆行列をもつためには **A** は正則(Rank=n)でなければならない。また以下の関係が成立する。

$$(\mathbf{A}^{-1})^{-1}=\mathbf{A} \tag{28}$$

$$(\mathbf{AB})^{-1}=\mathbf{B}^{-1}\mathbf{A}^{-1} \tag{29}$$

$$(\mathbf{A}^{T})^{-1}=(\mathbf{A}^{-1})^{T} \tag{30}$$

(ノルム)　$\|\mathbf{x}\| \geqq 0$、$\|\mathbf{x}+\mathbf{y}\| \leqq \|\mathbf{x}\| + \|\mathbf{y}\|$ を満たす $\|\cdot\|$ をノルムと呼び、次式で定義する。これは一種の距離の概念を与える。

$$\|\mathbf{x}\|^2=(\mathbf{x}、\mathbf{x})=\mathbf{x}^{T}\mathbf{x} \tag{31}$$

(微分の表記法)

$$\partial g(\mathbf{x})/\partial \mathbf{x}=(\partial g/\partial x_1,\cdots,\partial g/\partial x_n), \qquad d\mathbf{x}/dt = \begin{bmatrix} dx_1/dt \\ \cdot \\ dx_n/dt \end{bmatrix} \tag{32}$$

$$\partial \mathbf{f}(\mathbf{x})/\partial \mathbf{x} = \begin{bmatrix} \partial f_1/\partial x_1,\cdots,\partial f_1/\partial x_n \\ \cdot \quad ,\cdots, \quad \cdot \\ \partial f_m/\partial x_1,\cdots,\partial f_m/\partial x_n \end{bmatrix} \tag{33}$$

$$\partial [\mathbf{f}(\mathbf{x})^{T}\mathbf{g}(\mathbf{x})]/\partial \mathbf{x}=\mathbf{g}(\mathbf{x})^{T}(\partial \mathbf{f}(\mathbf{x})/\partial \mathbf{x})+\mathbf{f}(\mathbf{x})^{T}(\partial \mathbf{g}(\mathbf{x})/\partial \mathbf{x}) \tag{34}$$

（Appendix 2）実験用 NN ソフトウェアの使用法

本実験で用いる NN モデル化のプログラムは、3 層の階層型 NN を BP 法によって学習させるものである。フォートラン言語を用いてプログラムされており、実行形式のファイルを DOS 上で作動させて数値実験を行う。DOS プロンクトから NN（改行）と入力すると最初にデモを行うかどうか問合せがある。TRUE（改）と入力すれば XOR 問題が所定のパラメータの下で実行される。

一方、FALSE（改）の場合、順次、以下のようなキー入力が求められる。

① 学習繰り返し数の最大値、収束判定条件（RMSE の上限値）

② 学習係数 α と慣性係数 η の値

③ 学習データの入力

ここでは、(a)既に作成済みのファイルから入力する場合と、(b)引き続きキー入力する場合、が取り扱える。

(a) ファイル入力

1:‘ファイル名’(改)を入力

（ただし、データは以下のキー入力の場合と同様のものを用意すること）

(b) キー入力

1. 学習データ数、入力層数、中間層数、出力層数（改）
2. 出力値の最大値、最小値の組を出力層数回だけ入力する。
3. 学習データ数回だけ以下の入力を繰り返す。

入力データ(1)、…、入力データ（入力層数）（改）

教師データ(1)、…、教師データ（出力層数）（改）

以上で実行される。

なお、本プログラムにおける最大値はそれぞれ以下のようである。

（データ数＝50、入力層数＝50、中間層数＝30、出力層数＝6）

また、繰り返し数　vs. RMSE は、RMS.DAT

結合係数は、WEIGT.DAT

学習効果は、CHECT.DAT

とそれぞれ名付けられたファイルに保存される。

参考書等

Chitra, S.P., Use neural networks for problem solving., Chemical Engineering Progress, April,p.44-52 (1993)

Bhagat, Phiroz, An introduction to neural nets., Chemical Engineering Progress, August, p.5-60 (1990)

Rumelhart, David E., Hinton, Geoffrey E. and Williams, Ronald J., Learning representations by back-propagating errors., Nature, 323, 9, p.533-536 (1986)

麻生英樹, ニューラルネットワーク情報処理, 産業図書 (1988)

平野広美, C でつくるニューラルネットワーク, パーソナルメディア (1991)

実験課題

［1］シグモイド曲線の形をみて、それが生物学の分野で生成曲線と呼ばれる由来について考えてみよ。

［2］伝達関数がしきい関数のとき、しきい値 b の意味はより明確となる。しからば b の働きについて述べよ。

［3］シグモイド関数、f(x) = 1/(1+exp[-x])において、 df(x)/dx = f(x)(1-f(x))となることを示せ。

［4］(例 2.1)を NN でモデル化し、得られたモデルを用いて x（入力）∈[-0.75, 2.25]の範囲において x を 0.25 刻みで変化させたときの y（出力）の値を計算し、結果を図示せよ。

［5］OR 問題を、最小二乗によって線形モデル($y=a_0+a_1x_1+a_2x_2$)としてモデル化せよ。

［6］XOR 問題を、最小二乗によって線形モデル($y=a_0+a_1x_1+a_2x_2$)としてモデル化せよ。

［7］XOR 問題において NN の構成パラメータ（学習係数、慣性係数、中間層数）を変化させ、これらと性能（誤差、収束速度）との関係を調べよ。

［8］NN に「じゃんけん」を教えるにはどのような学習データを準備すればよいか考えよ。そして実際に学習させて結果を確かめよ。

［9］［4］～［6］の結果を参考にして、最小二乗法と NN によるモデル化の特徴や留意点について述べよ。

［10］本実験全体についての考察及び感想を述べよ。