複素ニューラルネットワークの高速学習アルゴリズム

正員 宮嶋 照行†　正員 長谷川孝明†

A Fast Learning Algorithm of Complex Valued Neural Networks

Teruyuki MIYAJIMA† *and* Takaaki HASEGAWA†, *Members*

† 埼玉大学工学部電気電子工学科，浦和市

Faculty of Engineering, Saitama University, Urawa-shi, 338 Japan

あらまし　本論文では，入出力信号と重み係数が複素数であるニューロンからなる複素ニューラルネットワークの高速学習アルゴリズムを提案している．計算機シミュレーションにより本学習アルゴリズムの有効性を示している．

キーワード　複素ニューラルネットワーク，学習アルゴリズム，高速化，RLSアルゴリズム，評価関数

1. まえがき

近年，入出力信号と重み係数が複素数であるニューロンからなる階層形の複素ニューラルネットワークがいくつか独立に提案され[1]~[3]，通信方式等の複素数を扱う分野での応用が期待されている．しかしながらこれまでに提案されている学習アルゴリズムは，従来の実数の Back Propagation (BP)[4] を複素数に拡張したものであり，実数の場合と同様に学習に時間がかかるという欠点を有している．そのため実用に際し，高速な学習アルゴリズムが必要となる．

Recursive Least-Squares (RLS) アルゴリズムは線形適応フィルタの適応アルゴリズムとして高速な収束特性を有することが知られている[7]．そこで本論文では，RLSアルゴリズムを用いた複素ニューラルネットワークの高速学習アルゴリズムを提案する．これまでに実数のニューラルネットワークの学習にRLSアルゴリズムを用いる手法が提案されている[5],[6]．文献(5)では，出力関数にシグモイド関数の代わりに区分線形関数を用いることで，また文献(6)では，各ニューロンの望ましい内部ポテンシャルと望ましい出力を推定することでRLSアルゴリズムの使用を可能にしていた．本論文で提案するアルゴリズムは，評価関数に出力層のニューロンの内部ポテンシャルと望ましい内部ポテンシャル(出力関数の逆関数より求めることができる)の2乗誤差和を用いることで，RLSアルゴリズムの使用を可能にしている．この点は，従来の学習アルゴリズムで評価関数に出力層のニューロンの出力と望ましい出力の2乗誤差を用いることと対照的である．

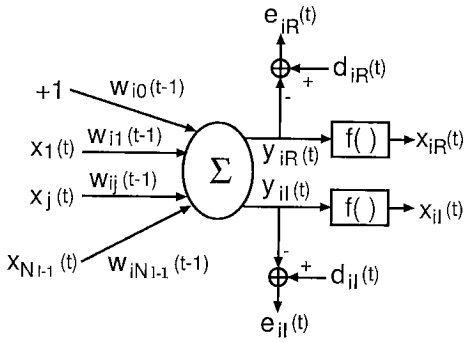

図1　ニューロンモデル
Fig. 1　Neuron model.

2. 複素ニューラルネットワークの高速学習アルゴリズムの提案

2.1 複素ニューラルネットワーク

L 層のネットワークについて検討する．本論文では入出力関係が次式で定義されるニューロン[1],[2]について検討する(図1)．時刻 t の第 l 層の i 番目のニューロンの内部ポテンシャル $y_i^{(l)}(t)$ と出力 $x_i^{(l)}(t)$ は次式で表される．

$$y_i^{(l)}(t)=\sum_{j=0}^{N_{l-1}} w_{ij}^{(l)}(t-1)x_j^{(l-1)}(t) \qquad (1)$$

$$x_i^{(l)}(t)=f\left(y_{iR}^{(l)}(t)\right)+jf\left(y_{iI}^{(l)}(t)\right) \qquad (2)$$

ここで N_l は第 l 層のニューロン数，$w_{ij}^{(l)}(t)$ は時刻 t の第 $l-1$ 層の j 番目のニューロンから第 l 層の i 番目のニューロンへの結合の重み係数である．式(2)の $f(\cdot)$ は出力関数，右辺第2項の j は虚数単位である．$x_0^{(l)}(t)$ は常に1であり，しきい値のための定数である．入力層 ($l=1$) のニューロンは入力信号をそのまま出力する．$y_i^{(l)}(t)$，$x_i^{(l)}(t)$，$w_{ij}^{(l)}(t)$ は複素数である．本論文では複素数変数の実部および虚部を例えば $y_{iR}^{(l)}(t)$，$y_{iI}^{(l)}(t)$ のように下付きの R と I で表す．また，式(2)と同様，式(6)，(12)，(13)の j は虚数単位を表す．

2.2 学習アルゴリズム

評価関数を次式で定義し，これを最小にすることを考える．

$$E(t)=\frac{1}{2}\sum_{n=0}^{t}\lambda^{t-n}\sum_{i=1}^{N_L}|e_i^{(L)}(n,t)|^2 \qquad (3)$$

ここで，

$$e_i^{(L)}(n,t)=d_i^{(L)}(n)-y_i^{(L)}(n,t) \qquad (4)$$

$\lambda(0<\lambda\leqq 1)$ は忘却の係数であり，時変環境での追従性を良くするために用いられる．$y_i^{(L)}(n,t)$，$e_i^{(L)}(n,t)$ はそれぞれ，時刻 n の入力データに対する，時刻 t の結

合の重みを有するネットワークの出力層の i 番目のニューロンの実際の内部ポテンシャルおよび誤差である．$d_i^{(L)}(n)$ は時刻 n の入力データに対する出力層の i 番目のニューロンの望ましい内部ポテンシャルであり，望ましい出力に出力関数の逆関数を適用し得られる．式(3)を各重み係数 $w_{ij}^{(l)}(t)$ で偏微分したものを0とすることで次式が得られる．

$$\sum_{n=0}^{t}\lambda^{t-n}e_i^{(l)}(n,t)x_j^{(l-1)*}(n,t)=0 \qquad (5)$$

ここで $x_j^{(l-1)}(n,t)$ は，時刻 n の入力データに対する，時刻 t の結合の重みを有するネットワークの第 $l-1$ 層の j 番目のニューロンの出力である．$*$ は複素共役を表す．$e_i^{(l)}(n,t)$ は出力層の場合は式(4)であり，中間層(第 l 層)の場合は次式で表される．

$$e_i^{(l)}(n,t)=f'(y_{iR}^{(l)}(n,t))\sigma_{iR}^{(l)}(n,t)+jf'(y_{iI}^{(l)}(n,t))\sigma_{iI}^{(l)}(n,t) \qquad (6)$$

ここで，

$$\sigma_i^{(l)}(n,t)=\sum_{k=1}^{N_{l+1}}w_{ki}^{(l+1)*}(t)e_k^{(l+1)}(n,t) \qquad (7)$$

ここで図1に示すように，中間層の $e_i^{(l)}(n,t)$ を出力層と同様に望ましい内部ポテンシャルと実際の内部ポテンシャルの差と考える．

$$e_i^{(l)}(n,t)=d_i^{(l)}(n)-\sum_{k=0}^{N_{l-1}}w_{ik}^{(l)}(t)x_k^{(l-1)}(n,t) \qquad (8)$$

式(8)を式(5)に代入することで決定論的正規方程式[7]が得られる．この決定論的正規方程式からRLSアルゴリズム[7]を用いて再帰的な学習アルゴリズムが得られる．

得られた学習アルゴリズムを以下にまとめる．

[Step 1] 初期化

すべての重み係数 $w_{ij}^{(l)}(0)$ を小さいランダムな値に初期化する．第 $l\in[2,L]$ 層の $(N_{l-1}+1)\times(N_{l-1}+1)$ 相関行列 $\boldsymbol{P}^{(l)}(0)$ を単位行列に初期化する．

[Step 2] 出力の計算

各ニューロンの内部ポテンシャル $y_i^{(l)}(t)$ および出力 $x_i^{(l)}(t)$ を式(1)，式(2)に従って計算する．

[Step 3] カルマンゲインと相関行列の更新

第 l 層のカルマンゲイン $\boldsymbol{K}^{(l)}(t)$ と相関行列 $\boldsymbol{P}^{(l)}(t)$ を更新する．

$$\boldsymbol{K}^{(l)}(t)=\frac{\boldsymbol{P}^{(l)}(t-1)\boldsymbol{X}^{(l-1)*}(t)}{\lambda+\boldsymbol{X}^{(l-1)T}(t)\boldsymbol{P}^{(l)}(t-1)\boldsymbol{X}^{(l-1)*}(t)} \qquad (9)$$

$$\boldsymbol{P}^{(l)}(t)=\frac{1}{\lambda}\{\boldsymbol{P}^{(l)}(t-1)-\boldsymbol{K}^{(l)}(t)\boldsymbol{X}^{(l-1)T}(t)\boldsymbol{P}^{(l)}(t-1)\} \qquad (10)$$

ここで $\boldsymbol{X}^{(l-1)}(t)$ は第 $l-1$ 層のニューロンの出力 $x_i^{(l-1)}(t)$ $(i\in[0,N_{l-1}])$ を要素とするベクトル．T は転置を表す．

[Step 4] 誤差信号の計算

出力層の場合，次式を誤差信号とする．

$$e_i^{(L)}(t)=d_i^{(L)}(t)-y_i^{(L)}(t) \qquad (11)$$

ここで，

$$d_i^{(L)}(t)=f^{-1}(t_{iR}(t))+jf^{-1}(t_{iI}(t)) \qquad (12)$$

$t_i^{(L)}(t)$ は出力層の i 番目のニューロンの望ましい出力を表す．中間層(第 $l\in[2,L-1]$ 層)の場合は次式を用いる．

$$e_i^{(l)}(t)=f'(y_{iR}^{(l)}(t))\sigma_{iR}^{(l)}(t)+jf'(y_{iI}^{(l)}(t))\sigma_{iI}^{(l)}(t) \qquad (13)$$

ここで，

$$\sigma_i^{(l)}(t)=\sum_{j=1}^{N_{l+1}}w_{ji}^{(l+1)*}(t-1)e_j^{(l+1)}(t) \qquad (14)$$

[Step 5] 重みの更新

第 $l-1$ 層から第 l 層の i 番目のニューロンへ結合している重みベクトルを次式に従って更新する．

$$\boldsymbol{W}_i^{(l)}(t)=\boldsymbol{W}_i^{(l)}(t-1)+\boldsymbol{K}^{(l)}(t)e_i^{(l)}(t) \qquad (15)$$

誤差が十分収束するまで，Step 2 から Step 5 を繰り返す．

本論文では出力関数に次式を用いる．

$$f(x)=(1-e^{-x})/(1+e^{-x}) \qquad (16)$$

式(16)の逆関数は次式で与えられる．

$$f^{-1}(x)=\ln(1+x)/(1-x) \qquad (17)$$

望ましい出力が2値の場合，通常用いられる ±1.0 とする式(12)が無限大になってしまうので，次章のシミュレーションでは望ましい出力を ±0.99 とした．

3. 計算機シミュレーション

2.で提案した学習アルゴリズムの性能を計算機シミュレーションにより評価する．文献(1)，(2)の複素BPアルゴリズムと比較を行う．ここでは複素BPアルゴリズムについて，実数のBPで通常用いられるように，慣性項を加えて高速化を図った．表1に示す学習パターン[8]について検討する．この学習パターンは単一の複素ニューロンでは分離不可能で，中間層が必要な問題である．ネットワークのニューロン数は，入力層が1，中間層が3，出力層が1とした．重みの初期値は -0.5 から $+0.5$ の一様乱数で与えた．提案する学習アルゴリズムと複素BPアルゴリズムともに同じ重みの初期値を用いた．学習パターンは#1から#4の順で呈示した．提案する学習アルゴリズムの忘却係数は0.8から1.0まで0.01刻みで，また複素BPの学習係数と慣性項の係数はそれぞれ0.1から0.9まで0.1刻みでシミュレーショ

表1 学習パターン

パターン番号	入力パターン	出力パターン
1	$-1-j$	$-0.99-j0.99$
2	$-1+j$	$0.99-j0.99$
3	$1-j$	$0.99+j0.99$
4	$1+j$	$-0.99+j0.99$

表2 シミュレーション結果

	提案学習法	複素BP
平均学習回数	37	3834
学習回数の標準偏差	12	8081
最小学習回数	22	624
最大学習回数	85	43704
収束率 [%]	100	99
1回の学習に要するCPU時間 [msec]	1.23	0.31

ンを行い最も早く収束したものを用いた．忘却係数は0.9，学習係数は0.3，慣性項の係数は0.8とした．

図2に学習曲線の一例を示す．この図より提案する学習アルゴリズムは複素BPに比べ，格段に少ない学習回数で2乗誤差が小さくなることがわかる．

次に収束に要する学習回数，時間について検討する．出力の2乗誤差が4回続けて，0.01以下になったとき学習を終了した．100,000回の学習で収束しない場合はローカルミニマムに陥ったとみなし学習を打ち切った．重み係数の初期値を変えて100回の実験を行った．

結果を表2にまとめる．収束に要する学習回数について，提案する学習アルゴリズムは複素BPに比べ約104倍高速である．また学習回数のばらつきが少ないこともわかる．1回の学習に要するCPU時間について，提案する学習アルゴリズムは複素BPの約4倍である．従って収束に要するCPU時間について，提案する学習アルゴリズムは複素BPに比べ約26倍高速である．

ニューロンへの入力の数をNとするとき，BPの計算量がNに比例するのに対し，RLSアルゴリズムの計算量はN^2に比例する[7]．従ってNが非常に大きくなった場合，提案する学習アルゴリズムの収束に要するCPU時間は非常に長くなる．しかしながら適応等化器[7]などの，一定の時間間隔で学習データがシステムに与えられ，短時間で環境の変化に追従しなければならないシステムでは，学習回数が少ないことが必要である．このような応用分野においては提案する学習アルゴリズムは有用であると考えられる．

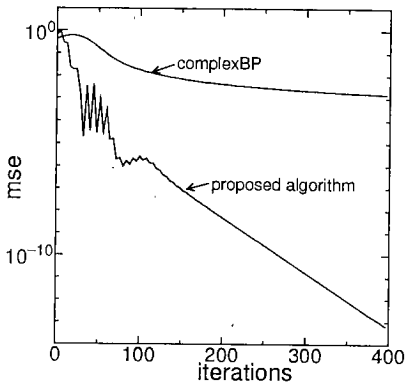

図2 学習曲線
Fig. 2 Learning curves.

4. むすび

本論文ではRLSアルゴリズムを用いた複素ニューラルネットワークの高速学習アルゴリズムを提案し，その動作を確認した．本アルゴリズムは，学習回数が少ないことが望まれる応用分野で特に有用である．本アルゴリズムの計算量の削減，および通信方式等への応用が今後の課題である．

謝辞 日ごろ御指導頂く埼玉大学工学部電気電子工学科羽石操教授，小林禧夫教授に深謝致します．

文献

(1) 新田 徹，古谷立美："複素バックプロパゲーション学習"，情処学論，**32**, 10, pp. 1319-1328 (1991-10).
(2) Benvenuto N. and Piazza F. : "On the Complex Backpropagation Algorithm", IEEE Trans. Signal Process., **40**, 4, pp. 967-969 (April 1992).
(3) Georgiou G. M. and Koutsougeras C. : "Complex Domain Backpropagation", IEEE Trans. Circuits & Syst., II, **39**, 5, pp. 330-334 (May 1992).
(4) Rumelhart D. E., Hinton G. E. and Williams R. J. : "Learning representations by back-propagating errors", Nature, **323**, pp. 533-536 (1986).
(5) Azimi-Sadjadi M. R. and Liou R. : "Fast Learning Process of Multilayer Neural Networks Using Recursive Least Squares Method", IEEE Trans. Signal Process., **40**, 2, pp. 446-450 (Feb. 1992).
(6) Scalero R. S. and Tepedelenlioglu N. : "A Fast New Algorithm for Training Feedforward Neural Networks", IEEE Trans. Signal Process., **40**, 1, pp. 202-210 (Jan. 1992).
(7) Haykin S.著，武部幹訳："適応フィルタ入門"，現代工学社 (1987).
(8) 新田 徹，古谷立美："複素BP学習の高速性"，信学技報，**NC91**-127 (1992).

(平成4年10月9日受付)