Aceleração do algoritmo de backpropagation

VITOR PERA*, CARLOS ESPAIN*
* Centro de Estudos de Física, Acústica e Telecomunicações (CEFAT)
Departamento de Engenharia Electrotécnica e Computadores (DEEC)
Faculdade de Engenharia da Universidade do Porto (FEUP)
Rua dos Bragas, 4099 Porto Codex, Portugal
tel: +351 2 2041831 fax: +351 2 2008366
e-mail: espain@garfield.fe.up.pt

Resumo

O treino de MLPs através de simples algoritmos de gradiente é fácil de implementar, mas é também geralmente de convergência muito lenta. Existem métodos que conseguem reduzir fortemente o número de iterações, mas alguns deles acarretam em cada iteração um enorme custo computacional, facto que pode acabar por manter o treino muito lento.

Neste artigo apresenta-se um novo procedimento que, repetidamente, usa sucessivos vectores de parâmetros obtidos por um algoritmo-base de treino, por exemplo o backpropagation, para através do algoritmo ε estimar um novo vector de parâmetros susceptível de ser mais próximo da solução, tentando assim acelerar o treino.

São apresentados resultados que atestam a capacidade de aceleração deste procedimento, destacando-se três situações: enorme aceleração quando o gradiente apresenta pequena variação e valor reduzido; capacidade de convergência quando a taxa de aprendizagem tem um valor relativamente elevado, mesmo para valores que não permitem a convergência do algoritmo backpropagation; aumento da aceleração se a superficie do erro se aproxima duma superficie quadrática, resultado coerente com o facto de, como será mostrado, a aplicação dum caso particular do algoritmo ε aos termos obtidos pelo backpropagation resultar numa aproximação ao método de Newton.

Palavras chave: MLP, backpropagation, aceleração, Δ^2 de Aitken, algoritmo ε, 2ª ordem, Newton

Abstract

The paper reports on the acceleration of the backpropagation algorithm for training multilayer perceptrons (MLP) by applying the ε algorithm to the succession of parameter vectors obtained in the iterations. It is shown that a particular case of the ε algorithm, the Aitken Δ^2, approximates a Newton 2^{nd} order method. Remarkable acceleration is achieved whenever the error surface is nearly quadratic or the gradient is small and slowly varying. It was also found that the algorithm allows convergence of the backpropagation with higher learning rates, even when, without it, convergence was not reached.

Key words: neural networks, MLP, backpropagation, Aitken Δ^2, ε algorithm

I. INTRODUÇÃO

Neste artigo são apresentados resultados sobre a aceleração, recorrendo ao algoritmo ε, do treino de MLPs usando simplesmente o algoritmo backpropagation. Uma breve referência a alguns aspectos do treino de MLPs é feita na Secção II. Na Secção III mostra-se que a aplicação do algoritmo Δ^2 de Aitken aos vectores de parâmetros obtidos pelo backpropagation é um método aproximado de

Newton. Na Secção IV é apresentado, apenas ao nível operacional, o algoritmo ε. Com o objectivo de acelerar o treino, apresentam-se, na Secção V, procedimentos aplicando o algoritmo ε ao método backpropagation. Na Secção VI são apresentados alguns resultados preliminares utilizando esta técnica em treino off-line.

II. TREINO DE MLPS

Definida a estrutura de um MLP adequada para resolver um dado problema e conhecido um conjunto para treino supervisionado, T, é necessário calcular os parâmetros que satisfaçam, em certa medida, determinado critério. Embora nalgumas situações outros critérios de erro sejam mais adequados, frequentemente utiliza-se o chamado critério do erro quadrático médio (MSE) [1], o qual implica minimizar a função :

$$E = \sum_{\mathbf{x} \in T} \left\| F(\mathbf{x}) - s(\mathbf{x}) \right\|^2 \qquad (1)$$

representando $\mathbf{x}$ um vector de treino, $F(\mathbf{x})$ o respectivo vector-saída do MLP e $s(\mathbf{x})$ o vector-alvo pretendido. Não sendo conhecida solução analítica para o problema do cálculo dos parâmetros, os algoritmos de aprendizagem inicializam o vector de parâmetros $\mathbf{w}_o$ usando técnicas específicas (Nguyen, SVDs, …) ou por simples arbítrio e recorrem depois a um procedimento iterativo de ajuste do vector de parâmetros $\mathbf{w}$ [2]:

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \lambda_k \mathbf{d}_k \qquad (2)$$

sendo λ_k a taxa de aprendizagem e $\mathbf{d}_k$ a direcção de pesquisa na iteração k.

Um primeiro método de resolver o problema consiste em tomar como direcção de descida a do gradiente de E, com o sentido oposto.

$$\mathbf{d}_k = -\nabla_k \qquad (3)$$

O algoritmo backpropagation utiliza esta direcção, o que tem a vantagem de exigir em cada iteração apenas o cálculo das primeiras derivadas. Por outro lado, apresenta a desvantagem de convergência eventualmente lenta.

Para se conseguir uma mais rápida convergência é necessário usar mais informação local sobre a função E, o que pode ser feito considerando derivadas de ordem superior.

No método de Newton toma-se como direcção de descida

$$\mathbf{d}_k = -\mathbf{H}_k^{-1} \nabla_k \qquad (4)$$

direcção que no caso da função ser quadrática determina directamente o mínimo em primeira iteração.

Numa situação geral o método de Newton possibilita uma convergência mais rápida, quadrática, se iniciado a partir dum ponto suficientemente vizinho do mínimo para que a função possa ser adequadamente representada pelo seu desenvolvimento em série de Taylor até ao termo quadrático.

$$\begin{aligned} E(\mathbf{w}_{k+1}) &= E(\mathbf{w}_k + \Delta\mathbf{w}) \\ &= E(\mathbf{w}_k) + \nabla_k^T \Delta\mathbf{w} + \frac{1}{2} \Delta\mathbf{w}^T \mathbf{H}_k \Delta\mathbf{w} \end{aligned} \qquad (5)$$

De qualquer modo a direcção de Newton é uma direcção de descida se a matriz hesseana for definida positiva. De uma forma ainda mais geral a direcção definida por (4) é direcção de descida se a hesseana for substituída por uma matriz arbitrária definida positiva.

Apesar de geralmente reduzir o número de iterações do treino, o método não é muito usado, pois exige em cada iteração o cálculo de segundas derivadas e a inversão da matriz hesseana, operação muito custosa. Assim, especialmente se o número de parâmetros for elevado, recorre-se a soluções aproximadas.

É o caso da utilização apenas dos termos da diagonal da hesseana [3], o que equivale a usar a regra de Newton separadamente para cada parâmetro, podendo ainda utilizar-se taxas de aprendizagem diferentes para cada parâmetro e tomá-las inversamente proporcionais aos respectivos termos da diagonal, melhorando o condicionamento da matriz [4].

Nos métodos designados por quasi-Newton ou de métrica variável, a matriz hesseana invertida é aproximada através de uma matriz reestimada iterativamente, dispensando o cálculo explícito de derivadas de 2ª ordem.

Outro método de optimização, relacionado com este último e que também usa indirectamente informação de 2ª ordem é o chamado método do momento, o qual toma uma direcção intermédia entre a direcção definida em cada iteração e a anterior:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \lambda_k \nabla_k + m_k \Delta\mathbf{w}_k \qquad (6)$$

Também a determinação da taxa de aprendizagem é importante, pois dela depende em grande medida a estabilidade do processo de treino e a velocidade de convergência. Existem diversas técnicas para a sua adaptação, desde as que usam uma taxa única para todos os parâmetros e simplesmente a incrementam (decrementam) quando o valor de E é diminuído (aumentado), até à utilizaçao de taxas de

aprendizagem diferentes para cada parâmetro, permitindo em muitas situações um desejável re-escalonamento dos parâmetros.

III. ALGORITMO Δ^2 DE AITKEN

Suponhamos conhecidos os vectores dos parâmetros em três iterações seguidas $\mathbf{w}_1$, $\mathbf{w}_2$, $\mathbf{w}_3$ do algoritmo de backpropagation.

Admitamos que a partir do vector $\mathbf{w}_1$ e do gradiente ∇_1, a direcção de Newton **d** calcula um novo vector **w** :

$$\mathbf{w}=\mathbf{w}_1-\mathbf{H}_1^{-1}\nabla_1 \tag{7}$$

Entretanto do algoritmo de backpropagation sabe-se que é aproximadamente

$$\begin{aligned}\mathbf{w}_2&=\mathbf{w}_1-\nabla_1\\ \mathbf{w}_3&=\mathbf{w}_2-\nabla_2\end{aligned} \tag{8}$$

Fazendo ainda a aproximação do modelo quadrático para a função de erro resulta

$$\mathbf{H}_1\left(\mathbf{w}_2-\mathbf{w}_1\right)=\nabla_2-\nabla_1 \tag{9}$$

Exprimindo os gradientes nos vectores de acordo com as (8) vem

$$\mathbf{w}=\mathbf{w}_1-\mathbf{A}\left(\mathbf{B}-\mathbf{A}\right)^{-1}\left(\mathbf{w}_2-\mathbf{w}_1\right) \tag{10}$$

com

$$\begin{aligned}\mathbf{A}&=\left(\mathbf{w}_2-\mathbf{w}_1\right)\left(\mathbf{w}_2-\mathbf{w}_1\right)^{\mathrm{T}}\\ \mathbf{B}&=\left(\mathbf{w}_3-\mathbf{w}_2\right)\left(\mathbf{w}_2-\mathbf{w}_1\right)^{\mathrm{T}}\end{aligned}$$

Considerando apenas os termos diagonais da hesseanal resulta para a componente i:

$$\mathbf{w}^{\mathrm{i}}=\mathbf{w}^{\mathrm{i}}_{1}-\frac{(\mathbf{w}^{\mathrm{i}}_{2}-\mathbf{w}^{\mathrm{i}}_{1})^{2}}{(\mathbf{w}^{\mathrm{i}}_{2}-\mathbf{w}^{\mathrm{i}}_{1})-(\mathbf{w}^{\mathrm{i}}_{3}-\mathbf{w}^{\mathrm{i}}_{2})} \tag{11}$$

Esta equação não é mais do que a equação característica do método de aceleração de convergência de sucessões Δ^2 de Aitken.

Quando a hesseana é não definida positiva ou quase singular é usual construir a partir dela uma nova matriz definida positiva da forma [5]:

$$\mathbf{N}=\mathbf{H}+\alpha\mathbf{I} \tag{12}$$

As novas equações escrevem-se então

$$\mathbf{w}=\mathbf{w}_1-\mathbf{N}_1^{-1}\nabla_1 \tag{13}$$

Para valores grandes de α a direcção aproxima-se da do gradiente, para valores baixos, da de Newton. Um método eficaz de controlo do valor de α é o de Levenberg-Marquardt.

No método de Aitken a introdução de α conduz à equação para as componentes

$$\mathbf{w}^{\mathrm{i}}=\mathbf{w}^{\mathrm{i}}_{1}-\frac{(\mathbf{w}^{\mathrm{i}}_{2}-\mathbf{w}^{\mathrm{i}}_{1})^{2}}{(\mathbf{w}^{\mathrm{i}}_{2}-\mathbf{w}^{\mathrm{i}}_{1})-(\mathbf{w}^{\mathrm{i}}_{3}-\mathbf{w}^{\mathrm{i}}_{2})}-\alpha(\mathbf{w}^{\mathrm{i}}_{2}-\mathbf{w}^{\mathrm{i}}_{1}) \tag{14}$$

Procura-se no momento estabelecer formas eficazes de controlar α neste caso.

IV. ALGORITMO ε

O algoritmo ε é uma generalização, devida a Shanks [6], mas apresentada na forma mais operacional por Wynn [7], do método Δ^2 de Aitken e ligada intimamente aos aproximantes de Padé.

Suponha-se conhecida uma sucessão $\{S_n\}$, com $S_n\to S$, e armazenem-se os seus termos em $\varepsilon_0^{(n)}$, n=0,1, … .

Faça-se $\varepsilon_{-1}^{(n)}=0$, n=0,1,...

O algoritmo consiste em definir recursivamente [8]

$$\varepsilon_{k+1}^{(n)}=\varepsilon_{k-1}^{(n+1)}+\frac{1}{\varepsilon_{k}^{(n+1)}-\varepsilon_{k}^{(n)}}\qquad n,k=0,1,... \tag{15}$$

As quantidades obtidas podem ser dispostas num quadro, geralmente designado quadro ε, como é mostrado na Fig.1:

$$\begin{array}{ccccc}
\varepsilon_{-1}^{(0)} & & & & \\
 & \varepsilon_{0}^{(0)} & & & \\
\varepsilon_{-1}^{(1)} & & \varepsilon_{1}^{(0)} & & \\
 & \varepsilon_{0}^{(1)} & & \varepsilon_{2}^{(0)} & \\
\varepsilon_{-1}^{(2)} & & \varepsilon_{1}^{(1)} & & \varepsilon_{3}^{(0)} \\
 & \varepsilon_{0}^{(2)} & & \varepsilon_{2}^{(1)} & \vdots \\
\varepsilon_{-1}^{(3)} & & \varepsilon_{1}^{(2)} & \vdots & \\
 & \varepsilon_{0}^{(3)} & \vdots & & \\
\varepsilon_{-1}^{(4)} & \vdots & & &
\end{array}$$

Fig. 1. Quadro ε

As quantidades de índice inferior ímpar são apenas resultado de cálculos intermédios. Mas as quantidades de índice inferior par formam sucessões susceptíveis de convergir para S, e muito mais rapidamente que $\{S_n\}$. Isso acontece, por exemplo,

com a sucessão $\varepsilon_k^{(n)}$, n=0,1,..., (k par), isto é, a sucessão constituída por uma coluna par do quadro. Na maior parte das sucessões que o algoritmo é susceptível de acelerar, a quantidade mais próxima do limite da sucessão, S, é o valor extremo de índice inferior par da diagonal ascendente correspondente ao último termo da sucessão $\{S_n\}$. Como se pode verificar, este algoritmo exige o conhecimento de $\{S_n, S_{n+1}, ..., S_{n+k}\}$ para poder calcular $\varepsilon_k^{(n)}$.

Em particular o ε reduz-se ao método de Aitken quando k=2. Quando k=4,6, …, o algoritmo é ainda mais eficiente e surge assim como candidato natural a método de aceleração da convergência da backpropagation, definindo direcções de descida na superfície de erro.

O algoritmo ε pode apresentar algumas dificuldades quando dois elementos seguidos duma mesma coluna são muito proximos, pois vêm subtraídos na relação de recorrência (15), ocorrendo erros de "cancelamento", os quais ao se propagarem no quadro ε podem tornar o algoritmo numericamente instável. Apesar de existirem procedimentos de cálculo que evitam aquela subtracção, nem sempre são completamente eficazes e acarretam maior custo computacional. Refira-se, finalmente, que a natureza de $\{S_n\}$ pode ser diversa, podendo os termos ser escalares, vectores ou matrizes, entre outros objectos matemáticos.

V. ALGORITMO ε APLICADO AO BACKPROPAGATION

O algoritmo ε vectorial a k colunas (k par) é aplicado sempre que se dispõe de k+1 termos obtidos do método backpropagation, obtendo-se assim um vector de parâmetros 'acelerado' $\mathbf{w}_\varepsilon$. O erro é então calculado nesse ponto. Se for menor toma-se esse novo vector $\mathbf{w}_\varepsilon$ como vector de partida e retoma-se o algoritmo normal de backpropagation até dispor de novo de k+1 vectores para nova aplicação do ε. Se for maior rejeita-se o vector $\mathbf{w}_\varepsilon$ e segue-se com o backpropagation. Passo a passo o procedimento é:

1) executar o treino normal de backpropagation, correndo k iterações e armazenando os vectores de parâmetros em **w**(i), i=1,2,... k;

fazer J=k+1;

2) desde i=J até k+1: a partir do vector **w**(i-1) aplicar o algoritmo backpropagation, armazenando o novo vector dos parâmetros em **w**(i) ;

3) aplicar o algoritmo ε à sucessão de vectores {**w**(i), i=1,2, ..., k+1}, obtendo o vector de parâmetros (acelerado) $\mathbf{w}_\varepsilon$;

4) se o erro da rede em $\mathbf{w}_\varepsilon$ for inferior ao erro correspondente ao último vector obtido pelo backpropagation, **w**(k+1),

fazer J=2, **w**(1)= $\mathbf{w}_\varepsilon$ e actualizar o erro,

se não,

fazer J=k+1; desde p=1 até k, **w**(p)=**w**(p+1);

5) se o erro não tiver alcançado o valor do erro mínimo pretendido, voltar para 2)

Outro procedimento de aceleração possível consiste simplesmente em aplicar o ε sempre que o backpropagation calcula um novo vector de parâmetros. A vantagem desta segunda variante relativamente ao algoritmo acima apresentado é a de não ter que esperar que o backpropagation itere k vezes quando o ε consegue actuar com sucesso. A desvantagem é que o algoritmo é aplicado a termos que resultam do processo de aceleração e, portanto, podem não reflectir de maneira correcta a topologia local da superficie de erro.

Outra possibilidade é a de utilizar uma versão vectorial de qualquer um dos dois processos referidos, transformando simplesmente a equação (11) numa equação vectorial.

Também no algoritmo ε se manifesta uma dificuldade que surge em geral nos métodos de ordem superior à 1ª, e que consiste no aparecimento de singularidades quando as derivadas se tornam pequenas. Para ultrapassar esse problema, provou-se ser útil adicionar uma constante β ao denominador da fracção em (15) quando o k é ímpar. Estuda-se neste momento a forma eficaz de adaptar o valor desse parâmetro β de iteração para iteração.

No caso de localmente a função E ser convexa a matriz hesseana não é definida positiva, o que implicaria a não convergência deste método, podendo usar-se novamente a constante acima referida para forçar uma direcção de pesquisa com projecção negativa segundo o gradiente.

VI. RESULTADOS

Os resultados que aqui se apresentam referem-se ao treino de duas redes usadas em exemplos que acompanham a toolbox de redes neuronais do pacote de software Matlab v 4.0 [9]. Sempre que nada for referido em contrário, considere-se que os resultados foram obtidos pela versão vectorial do algoritmo detalhado em V, tendo sido impedido que a norma do denominador em (15), se k ímpar, tivesse valor inferior a 1, por escolha acertada do parâmetro β.

A. Avaliação do comportamento geral do algoritmo

A rede apresentada no ficheiro "Bckprop1" do Matlab v4.0 consiste num unico nó, usando a função logsig e permitindo visualizar a trajectória do vector dos parametros durante o treino, e portanto comodamente avaliar o comportamento de diferentes configurações do processo de aceleração em zonas da superfície de erro com topologias distintas.

Na Tabela 1 são apresentadas percentagens da redução relativamente ao treino sem aceleração do número de iterações necessárias para obter a convergência, obtidas a partir de 150 inicializações aleatórias dos parâmetros na rede "Bckprop1". Essas percentagens foram determinadas para três valores diferentes de k e para dois niveis de exigência do critério do erro.

TABELA I
Redução percentual do número de iterações com ε

erro final	*k=2 (Aitken)*	*k=4*	*k=6*
10^{-3}	43%	48%	32%
10^{-5}	63%	72%	74%

Embora essa propriedade não seja, na generalidade, de grande interesse prático, porque pode implicar um 'overfitting' da rede, é significativo o aumento da aceleração quando se exige menor erro final, o que confirma o melhor comportamento do ε próximo da solução. Isso deve-se ao facto de aí a superfície de erro ser melhor representada por uma função quadrática.

Foram efectuadas experiências substituindo a função logsig pela função linear e, constatou-se como era teoricamente esperável, que no caso de funções de erro quadráticas o ε acelera dramaticamente a convergência, mesmo quando o algoritmo backpropagation não é convergente.

B. Gradientes pequenos

Verificou-se experimentalmente uma enorme aceleração sempre que a superficie de erro se apresenta plana e com declive reduzido, ou seja, quando o gradiente se mantém praticamente constante e com valor pequeno.

Nessa situação, os métodos de simples gradiente são particularmente lentos, e pelo contrário o algoritmo ε apresenta um excelente comportamento, desde que se tenha o cuidado de considerar, como já referido, uma pequena constante no denominador de (15), pois caso contrário poder-se-ia tornar muito instável originando enormes "saltos".

A Fig. 2 mostra as trajectórias definidas no espaço dos parâmetros pelas iterações do algoritmo numa situação típica de gradiente pequeno.

Sem ε nem adaptação da taxa de aprendizagem l_r (l_r=1) a convergência dá-se em 6007 iterações; com o Δ^2 de Aitken (ε com k=2) dá-se em 22 iterações.

De referir que, com a taxa de aprendizagem adaptável com factor de incremento 1,05 e de decremento 0,7, o número de iterações foi de 137.

Todas as experiências de situações de gradiente pequeno e lentamente variável deram resultados no essencial semelhantes, caracterizados por eficiência do algoritmo ε comparável á técnica da taxa de aprendizagem adaptada e em muitos casos manifestamente superior.

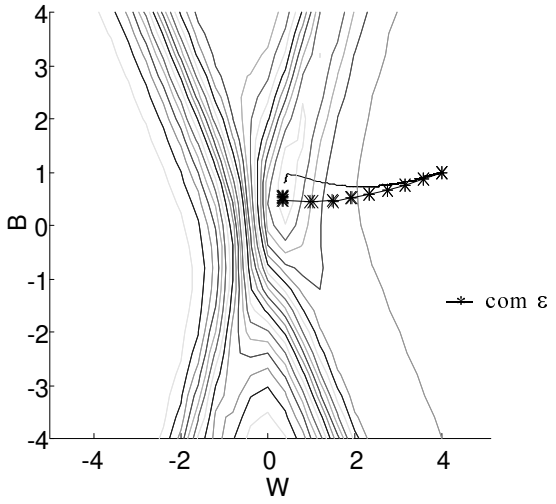

Fig. 2. Curvas de nível da superfície de erro e trajectória do algoritmo de treino no espaço bidimensional (1 nó) dos parâmetros (w-peso, b-viés) numa situação de gradiente lentamente variável e pequeno

Também se verificou um apreciável poder de aceleração em alguns casos onde a forma da superficie se aproxima dum "desfiladeiro", nao tendo no entanto sido completamente identificadas as condições em que o ε consegue produzir esses resultados.

C. Taxas de aprendizagem elevadas

O exemplo apresentado no ficheiro "Bckprop2" ilustra o problema da taxa de aprendizagem não poder ter um valor excessivo, pois se isso acontecer o backpropagation pode não conseguir convergir.

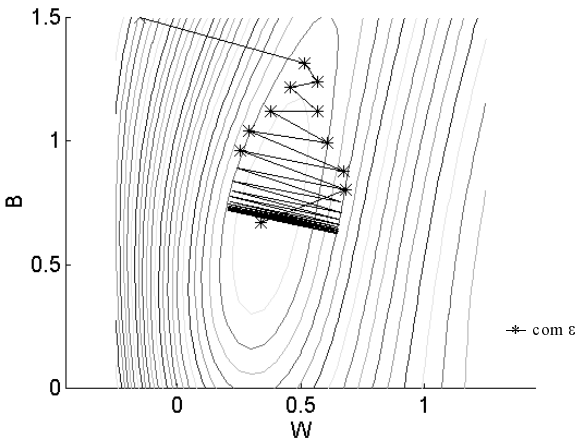

Fig. 3. Curvas de nível da superfície de erro e trajectória do algoritmo de treino no espaço bidimensional (1 nó) dos parâmetros (w-peso, b-viés) numa situação de taxa de aprendizagem elevada

Mantendo inalterados todos os parâmetros do treino e aplicando apenas o algoritmo ε, passou a verificar-se a convergência. Contatou-se que esse facto não é acidental. Basta referir que, incrementando sucessivamente o l_r de uma unidade a partir de 1, sem usar o algoritmo ε, o treino deixa de convergir para l_r=4. Pelo contrário, aplicando o ε o treino deixa apenas de convergir quando l_r=16.

A Fig. 3 mostra a trajectória, junto ao mínimo, do algoritmo sem ε, divergente, com l_r=4, e a do algoritmo com ε (k=6).

D. Aproximação de funções

A rede apresentada no ficheiro "Bckprop4" tem 5 nós (função tansig) na camada interior e 1 nó linear na saída e é usada para fazer a aproximação de uma função.

Na Tabela II são apresentados, para o treino sem aceleração e também com aceleração (k = 2, 4, e 6), o numero de iterações necessárias para satisfazer o critério de treino, o número de operações aritméticas realizadas para o efeito e tambem o número de operações por iteração. Como se pode verificar, o número de iterações foi reduzido dum factor entre 3 e 4, e o número de operações aritméticas reduzido aproximadamente a metade.

TABELA II

Comportamento do treino com e sem ε

	sem ε	*k=2*	*k=4*	*k=6*
nº de iterações	6809	2273	1908	1786
flops*10^6	18,4	9,2	8,9	8,7
flops/ iteração	2700	4000	4700	4900

Também através deste exemplo se pode verificar o comportamento muito positivo do algoritmo ε quando a taxa de aprendizagem apresenta um valor demasiado elevado.

Na Fig. 4 são apresentadas as curvas do erro obtidas com l_r=0,04.

Sem ε, o treino regista momentos de elevada instabilidade e não converge para o erro final E=0,02 desejado até pelo menos 8000 iterações (curva (1)).

Com o algoritmo ε (k=2) consegue convergir ao fim de 4396 iterações (curva (2)), e a versão que aplica o ε sempre que o backpropagation determina um novo vector de parâmetros converge em 1953 iterações (curva (3)).

Este melhor comportamento da 2ª versão relativamente à 1ª em situações de l_r demasiado elevado verificou-se na generalidade dos teste efectuados.

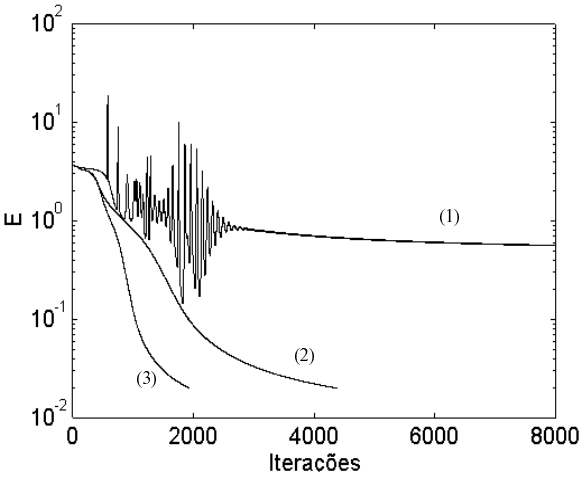

Fig. 4. Curvas de erro no treino de uma rede para aproximação de funções sem ε (curva(1)), com ε (curva(2)), e com a segunda variante do ε (curva(3))

VII. CONCLUSÕES

Face aos resultados obtidos é possível concluir-se que, genericamente, o treino de MLPs pelo algoritmo backpropagation pode ser acelerado através dum procedimento que utiliza o algoritmo ε.

Demonstrou-se que a aplicação dum caso particular do algoritmo ε, Δ^2 de Aitken, aos termos obtidos pelo backpropagation resulta numa aproximação ao método de Newton.

Foram identificadas situações em que a aceleração é particularmente interessante caracterizado por:

1) a superficie do erro praticamente plana e com reduzido declive;
2) a taxa de aprendizagem relativamente elevada, mesmo não permitindo a convergência do backpropagation;
3) a superficie de erro razoavelmente representada por um modelo quadrático.

VIII. AGRADECIMENTOS

Este trabalho desenvolveu-se ao abrigo do Acordo Cultural entre a JNICT(Portugal) e a CAPES(Brasil)

IX. BIBLIOGRAFIA

[1] B. Widrow, "30 years of adaptive neural networks: perceptron, madline and backpropagation," in *Proc. IEEE*, vol. 78, pp. 1415-1442, Sep. 1990.

[2] L. Tavares, *Optimização linear e nao linear*. Fund. Calouste Gulbenkian, 1984.

[3] H. Bourlard, *Connectionist speech recognition*. Kluwer Academic Publishers, 1994.

[4] Y. Bengio, *Neural Networks for Speech and Sequence Recognition*. Int. Thomson Computer Press, 1995.
[5] C. Espain, *Parâmetros aparentes em prospecção geoeléctrica: uma nova técnica de interpretação*. Porto: FEUP, 1984.
[6] D. Shanks, "Non linear transformation of divergent and slowly convergent series," in *J. Math. Phys*, vol. 34, pp.1-42.
[7] P. Wynn, "On a device for computing the em(Sn) transformation," in *M.T.A.C*, vol. 10, pp. 91-96.
[8] C. Brezinski, *Algorithmes d'accélétarion de la convergence: Étude numérique*. Paris: Éditions technip, 1978.
[9] *Matlab - Neural Network Toolbox User's Guide*. Math Works.