ELSEVIER

Signal Processing 80 (2000) 1909–1916

SIGNAL PROCESSING

www.elsevier.nl/locate/sigpro

A normalised real time recurrent learning algorithm

Danilo P. Mandic[a,*], Jonathon A. Chambers[b]

[a]School of Information Systems, University of East Anglia Norwich, Earlham Road, NR4 7TJ, UK
[b]Department of Electrical and Electronic Engineering, Imperial College of Science, Technology and Medicine, Exhibition Road, SW7 2BT, UK

Received 23 November 1998; received in revised form 31 March 2000

Abstract

A real time recurrent learning (RTRL) algorithm with an adaptive-learning rate for nonlinear adaptive filters realised as fully connected recurrent neural networks (RNNs) is derived. The algorithm is obtained by minimising the instantaneous squared error at the output neuron for every time instant while the network is running. The algorithm normalises the learning rate with the $\mathscr{L}_2$ norm of the external input vector and a measure of the gradients at the neurons within the network, and is hence referred to as the normalised RTRL (NRTRL) algorithm. Indeed, the algorithm degenerates into the normalised least mean square (NLMS) algorithm for a linear-single-neuron network. For a neuron with a contractive nonlinear activation function, the algorithm is shown to impose additional stability and faster convergence to the RTRL, without significant demands on additional computational complexity. The bounds imposed on the learning rate which preserve convergence of the algorithm are also provided. © 2000 Elsevier Science B.V. All rights reserved.

Zusammenfassung

Es wird ein rekursiver Lernalgorithmus in Echtzeit (RTRL) mit adaptiver Lernrate für nichtlineare adaptive Filter abgeleitet, wobei es als vollständig verbundenes rekursives Neuronales Netzwerk realisiert wird. Der Algorithmus basiert auf der Minimierung des augenblicklichen quadratischen Fehlers am Ausgangsneuron, der, während das Netzwerk arbeitet, für jeden Zeitpunkt berechnet wird. Der Algorithmus normalisiert die Lernrate mit Hilfe einer Kombination aus der $\mathscr{L}_2$-Norm des externen Eingangsvektors und einer Messung der Gradienten an den Neuronen innerhalb des Netzwerkes. Deshalb wird er als normalisierter RTRL-Algorithmus (NRTRL) bezeichnet. Der Algorithmus vereinfacht sich zu einem normalisierten Kleinste-Quadrate Algorithmus mit gleitender Mittelung (NLMS), wenn das Netzwerk nur aus einem einzelnen linearen Neuron besteht. Für ein Neuron mit kontrahierender nichtlinearer Anregungsfunktion liefert der Algorithmus im Vergleich zum RTRL zusätzliche Stabilität und schnellere Konvergenz, ohne signifikant höhere Rechenleistung zu benötigen. Die Grenzen der Lernrate, die eine Konvergenz des Algorithmus sicherstellen, werden ebenfalls gegeben. © 2000 Elsevier Science B.V. All rights reserved.

Résumé

Nous dérivons un algorithme d'apprentissage récurrent en temps réel (RTRL) à taux d'apprentissage adaptatif pour du filtrage adaptatif non linéaire réalisé comme des réseaux de neurones récurrents entièrement connectés. L'algorithme est obtenu par minimisation de l'erreur quadratique instantanée au neurone de sortie pour chaque instant temporel lorsque le réseau est en marche. L'algorithme normalise le taux d'apprentissage avec la norme $\mathscr{L}_2$ du vecteur d'entrée externe et une mesure du gradient aux neurones dans le réseau, et est donc référence comme étant

* Corresponding author. Tel.: +44-1603592569; fax: +44-1603593345
E-mail address: j.chambers@ic.ac.uk (J.A. Chambers).

0165-1684/00/$ - see front matter © 2000 Elsevier Science B.V. All rights reserved.
PII: S0165-1684(00)00101-8

l'algorithme RTRL normalisé (NRTRL). En effet, l'algorithme dégénère en l'algorithme des moindres carrés moyens normalisés pour un réseau à neurone unique linéaire. Pour un neurone avec une fonction d'activation non linéaire contractive, on montre que l'algorithme impose une stabilité additionnelle et une convergence plus rapide par rapport au RTRL, sans exigences significatives sur la complexité de calcul additionnelle. Nous fournissons également les limites imposées au taux d'apprentissage qui préservent la convergence de l'algorithme.

Keywords: Nonlinear adaptive filtering; Recurrent neural networks; Real time recurrent learning; Adaptive learning rate

1. Introduction

The most popular gradient-based algorithms for training recurrent neural networks (RNNs) are the backpropagation through time (BPTT) [24], recurrent backpropagation (RBP) [21], and real time recurrent learning (RTRL) [25] algorithms. Among them, the RTRL is best suited for real time applications [9]. However, it suffers from a slow convergence, which is typical for all gradient-based algorithms.

In the area of linear adaptive filters, the problem of slow convergence of the least mean square (LMS) algorithm has been addressed by utilising an adaptive learning rate (step-size). The idea behind the variable learning-rate LMS is that the algorithm should be normalised by a function of the input data to the filter, in order to provide faster convergence of the algorithm, and a smaller value of misadjustment [10]. Some of the criteria which have been proposed for the adaptive step size LMS are: squared instantaneous error [12]; sign changes of successive samples of the gradient [7]; reducing the squared error at each instant [19]; cross correlation of input and error [23]; and the square of a time-averaged estimate of the autocorrelation of two consecutive error terms [2]. The most popular variant of the LMS algorithm which uses an adaptive step size is the normalised LMS (NLMS) algorithm, whose derivation involves optimisation by the method of Lagrange multipliers [10].

In the area of feedforward neural networks, backpropagation is the most widely used gradient-based algorithm. The most popular algorithm for backpropagation with an adaptive learning rate is the delta-bar-delta rule [11], which also suffers from sensitivity to noise and relative instability [9,6]. A further attempt to improve convergence of backpropagation-based algorithms was via a posteriori gradient algorithms [14] and normalised backpropagation [18].

Hence, there is a need for an RTRL-based learning algorithm with an adaptive learning rate which would impose similar stabilisation and convergence effects on training of RNNs as normalisation imposes on the LMS algorithm. Here, we derive such an algorithm, which is optimal in the sense that it is a solution of an optimisation task, which minimises the instantaneous squared error at the output neuron for every discrete time instant, while the network is running.

2. The problem of learning rate adaptation

A single RNN is shown in Fig. 1. For the ith neuron, its weights form a $(p+N+1)\times 1$-dimensional weight vector $\mathbf{w}_i^{\mathrm{T}}(k) = [w_{i,1}(k), \ldots, w_{i,p+N+1}(k)]$, where p is the number of external inputs, N is the number of feedback connections, and $(\cdot)^{\mathrm{T}}$ denotes the vector transpose operation. One additional element of the weight vector $\mathbf{w}$ is the bias input weight. The feedback consists of the delayed output signals of the RNN. The following equations fully describe the RNN from Fig. 1

$$y_i(k) = \Phi(v_i(k)), \quad i = 1,2,\ldots,N \tag{1}$$

$$v_i(k) = \sum_{l=1}^{p+N+1} w_{i,l}(k)u_l(k), \tag{2}$$

$$\begin{aligned}\mathbf{u}_i^{\mathrm{T}}(k) = [&s(k-1),\ldots,s(k-p),1,\\ &y_1(k-1),y_2(k-1),\ldots,y_N(k-1)],\end{aligned} \tag{3}$$

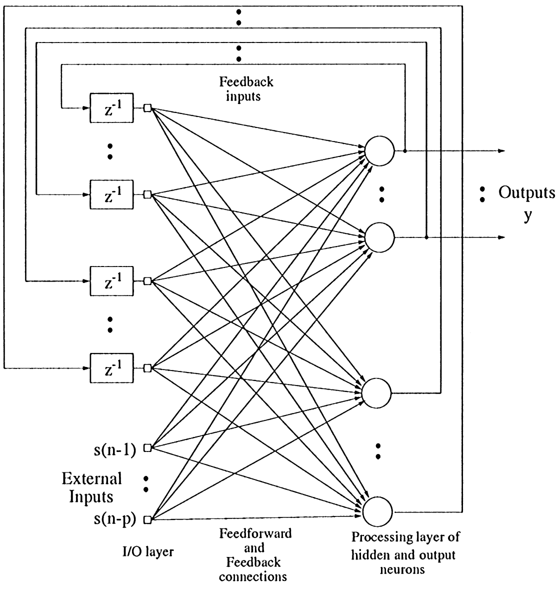
Feedback inputs
z^{-1}
Outputs y
s(n-1)
External Inputs
s(n-p)
I/O layer
Feedforward and Feedback connections
Processing layer of hidden and output neurons

Fig. 1. Single recurrent neural network.

where the $(p+N+1)\times 1$ dimensional vector $\boldsymbol{u}$ comprises both the external and feedback inputs to a neuron, as well as the unity valued constant bias input. The nonlinear activation function of a neuron is denoted by Φ and is assumed to be continuously differentiable.

2.1. Weight adaptation in RNNs

The equations that define the adaptation in single-output recurrent neural networks are

$$e(k) = s(k) - \Phi(v_1(k)), \tag{4}$$

$$\boldsymbol{W}(k+1) = \boldsymbol{W}(k) - \eta \nabla_{\boldsymbol{W}(k)} e^2(k), \tag{5}$$

where $e(k)$ is the instantaneous error at the output neuron, $s(k)$ is some teaching (desired) signal, $\boldsymbol{W}(k) = [\boldsymbol{w}_1(k), \ldots, \boldsymbol{w}_N(k)]$ is the weight matrix which consists of N column vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_N$ which represent the set of weights associated with every neuron. The learning rate η is supposed to be a small positive real number chosen by the user. Notice that the weight vectors consist of two subvectors, namely

$$\boldsymbol{w}_j = \begin{bmatrix} \boldsymbol{w}_{a,j} \\ \boldsymbol{w}_{b,j} \end{bmatrix}, \quad j = 1, \ldots, N \tag{6}$$

where $\boldsymbol{w}_{a,j}$ corresponds to the weights associated with the feedback inputs, whereas the weights with the index b corresponds to the weights associated with the external and bias inputs. Hence, the weight matrix $\boldsymbol{W}$ can be split into two submatrices, $\boldsymbol{W}_a$ and $\boldsymbol{W}_b$.

For a particular initial choice of η, the problem of gradient-based training can be generalised as [5]

$$\text{minimise} \quad \|\boldsymbol{W}(k+1) - \boldsymbol{W}(k)\|_p \tag{7}$$

$$\text{subject to} \quad s(k) - \Phi(\boldsymbol{w}_1^{\mathrm{T}}(k+1)\boldsymbol{u}(k)) = 0 \tag{8}$$

where $\|\cdot\|_p$ denotes the $\mathscr{L}_p$ norm. Notice that the term (8) actually represents the a posteriori error, as shown in [14].

In order to obtain the expression for an optimal adaptive learning rate, which would minimise the instantaneous squared error at the output neuron, we consider the expression for the instantaneous error at the output neuron (4). By expanding the error term (4) with a Taylor series, we obtain

$$e(k+1) = e(k) + \sum_{i=1}^{N} \sum_{j=1}^{p+N+1} \frac{\partial e(k)}{\partial w_{i,j}(k)} \Delta w_{i,j}(k) + \frac{1}{2} \sum_{i=1}^{N} \sum_{m=1}^{p+N+1} \sum_{j=1}^{N} \sum_{n=1}^{p+N+1} \frac{\partial^2 e(k)}{\partial w_{i,m}(k) \partial w_{j,n}(k)} \times \Delta w_{i,m}(k) \Delta w_{j,n}(k) + \cdots \tag{9}$$

where only the first two terms will be considered.

Due to the internal feedback in RNNs, the partial derivatives $\partial e(k)/\partial w_{i,j}(k)$ are not straightforward to calculate. However, since both $\boldsymbol{W}(k)$ in (5) and the Taylor series expansion in (9) include the same partial derivatives, as does the RTRL algorithm for an RNN, we can obtain the necessary partial derivatives directly from the RTRL.

3. The RTRL algorithm

The RTRL training algorithm for RNNs is based upon minimising the instantaneous squared error at the output of the first neuron of the RNN [25,9], which can be expressed as

$$\min(e^2(k)) = \min([s(k) - y_1(k)]^2). \tag{10}$$

Hence, the correction for the lth weight of neuron n at the time instant k can be derived as

$$\Delta w_{n,l}(k) = -\eta \frac{\partial}{\partial w_{n,l}(k)} e^2(k)$$
$$= -2\eta e(k) \frac{\partial e(k)}{\partial w_{n,l}(k)} = 2\eta e(k) \frac{\partial y_1(k)}{\partial w_{n,l}(k)}. \quad (11)$$

This can be further evaluated as

$$\frac{\partial y_1(k)}{\partial w_{n,l}(k)} = \Phi'(v_1(k)) \frac{\partial v_1(k)}{\partial w_{n,l}(k)}$$
$$= \Phi'(v_1(k)) \left(\sum_{\alpha=1}^{N} \frac{\partial y_\alpha(k-1)}{\partial w_{n,l}(k)} w_{1,\alpha+p+1}(k) + \delta_{nl} u_l(k) \right) \quad (12)$$

where

$$\delta_{nl} = \begin{cases} 1, & n = l, \\ 0, & n \neq l. \end{cases} \quad (13)$$

Under the assumption, commonly used in gradient based optimisation algorithms, and also used in the RTRL algorithm [25,20], that when the learning rate η is sufficiently small, we have

$$\frac{\partial y_\alpha(k-1)}{\partial w_{n,l}(k)} \approx \frac{\partial y_\alpha(k-1)}{\partial w_{n,l}(k-1)}, \quad (14)$$

and the recursive equation for gradient updates becomes

$$\pi^j_{n,l}(k+1) = \Phi'(v_j) \left[\sum_{m=1}^{N} w_{j,m}(k) \pi^m_{n,l}(k) + \delta_{nj} u_l(k) \right], \quad (15)$$

with the values for j, n, and l as in (17) and the initial conditions

$$\pi^j_{n,l}(0) = 0, \quad (16)$$

where

$$\pi^j_{n,l}(k) = \frac{\partial y_j(k)}{\partial w_{n,l}(k)},$$
$$1 \leqslant j,\ n \leqslant N,\ 1 \leqslant l \leqslant p + 1 + N. \quad (17)$$

Finally, the correction to the, say, $w_{n,l}(k)$ weight can be expressed as

$$\Delta w_{n,l}(k) = 2\eta e(k) \pi^1_{n,l}(k). \quad (18)$$

4. An improved learning rate for the RTRL algorithm

From the previous analysis, the weight matrix update can be expressed as

$$\Delta \boldsymbol{W}^{\mathrm{T}}(k) = 2\eta e(k) \frac{\partial y_1(k)}{\partial \boldsymbol{W}(k)} = 2\eta e(k) \boldsymbol{\Pi}_1(k), \quad (19)$$

where $\boldsymbol{\Pi}_1(k)$ represents the matrix of gradients at the output neuron with respect to $\boldsymbol{W}(k)$.

To simplify the presentation, we introduce three new matrices, the $N \times (N+p+1)$ matrix $\boldsymbol{\Pi}_j(k)$, the $N \times (N+p+1)$ matrix $\boldsymbol{U}_j(k)$, and the $N \times N$ diagonal matrix $\boldsymbol{F}(k)$, as

$$\boldsymbol{\Pi}_j(k) = \frac{\partial \boldsymbol{y}(k)}{\partial \boldsymbol{w}_j(k)},$$
$$\boldsymbol{y} = [y_1(k), \ldots, y_N(k)],\ j = 1,2,\ldots,N, \quad (20)$$

$$\boldsymbol{U}_j(k) = \begin{bmatrix} 0 \\ \vdots \\ \boldsymbol{u}(k) \\ \vdots \\ 0 \end{bmatrix} \leftarrow j\text{th row}, \quad j = 1,2,\ldots,N, \quad (21)$$

$$\boldsymbol{F}(k) = \mathrm{diag}(\Phi'(\boldsymbol{w}_1^{\mathrm{T}}(k)\boldsymbol{u}(k)), \ldots, \Phi'(\boldsymbol{w}_N^{\mathrm{T}}(k)\boldsymbol{u}(k))). \quad (22)$$

From (15), the gradient updating equation regarding the recurrent neuron can be symbolically expressed as

$$\boldsymbol{\Pi}_j(k+1) = \boldsymbol{F}(k)[\boldsymbol{U}_j(k) + \boldsymbol{\Pi}_j(k)\boldsymbol{W}_a(k)],$$
$$j = 1,2,\ldots,N. \quad (23)$$

The correction to the weight vector of the jth neuron, at the time instant k becomes

$$\Delta \boldsymbol{w}_j^{\mathrm{T}}(k) = 2\eta(k) e(k) \boldsymbol{\Pi}_1^{(j)}(k) \quad (24)$$

where $\boldsymbol{\Pi}_1^{(j)}$ represents the jth row of the gradient matrix $\boldsymbol{\Pi}_1(k)$. In addition, the terms $\partial e(k)/\partial w_{i,j}(k)$ become $\Delta w_{i,j}(k)/\eta(k)e(k)$.

Furthermore, the instantaneous output error $e(k)$ can be expressed as

$$e(k) = s(k) - \Phi(\boldsymbol{w}_1^{\mathrm{T}}(k)\boldsymbol{u}(k))$$
$$= [s(k) - \Phi(\boldsymbol{w}_1^{\mathrm{T}}(k+1)\boldsymbol{u}(k))]$$
$$+ [\Phi(\boldsymbol{w}_1^{\mathrm{T}}(k+1)\boldsymbol{u}(k)) - \Phi(\boldsymbol{w}_1^{\mathrm{T}}(k)\boldsymbol{u}(k))] \quad (25)$$

where the first term in square brackets represents the a posteriori error [14], and vanishes if the condition (8) is satisfied. In order to insure that the second term in (25) is small in magnitude, it is desirable to have a contractive nonlinear activation function of a neuron. Function Φ is a contraction on $[a,b] \in \mathbb{R}$, if $|\Phi(b) - \Phi(a)| \leqslant \alpha |b-a|$, where $0 < \alpha < 1$ is a small, positive number [8,13]. In this, although, rather generalised case, the instantaneous output error becomes proportional to $\|\boldsymbol{W}(k) - \boldsymbol{W}(k-1)\|_2$.

5. The optimal learning rate

Combining (9), (23), and (24), and neglecting the higher-order terms in the Taylor series expansion (9), we obtain

$$\begin{aligned} e(k+1) &= e(k) - 2\eta(k)e(k)\sum_{i=1}^{N}\sum_{j=1}^{p+N+1}\left[\frac{\partial y_1(k)}{\partial w_{i,j}(k)}\right]^2 \\ &= e(k) - 2\eta(k)e(k)\sum_{i=1}^{N}\|\boldsymbol{\Pi}_1^{(i)}(k)\|_2^2. \end{aligned} \tag{26}$$

The instantaneous squared error is therefore given by

$$e^2(k+1) = e^2(k)\left[1 - 2\eta(k)\sum_{i=1}^{N}\|\boldsymbol{\Pi}_1^{(i)}(k)\|_2^2\right]^2. \tag{27}$$

In order to obtain the minimum of (27), we differentiate with respect to η, and obtain the optimal value of learning rate $\eta_{\mathrm{OPT}}(k)$ for an RTRL trained RNN as

$$\eta_{\mathrm{OPT}}(k) = \frac{1}{2\sum_{i=1}^{N}\|\boldsymbol{\Pi}_1^{(i)}(k)\|_2^2}. \tag{28}$$

Notice that this relationship is closely related to the learning rate in the NLMS algorithm for linear adaptive filters [16]. Indeed, for a linear activation function of a neuron, and the network with only one neuron, the adaptive learning rate from (28) becomes exactly the learning rate obtained in the NLMS algorithm. For a linear-NLMS filter, the learning rate is normalised by the tap input power of the input signal, whereas in the nonlinear, recurrent case, the normalisation factor is the tap input power to an RNN multiplied by the derivative of the nonlinear activation function at the current point $net(k) = \boldsymbol{w}_1^{\mathrm{T}}(k)\boldsymbol{u}(k)$, and augmented by the product of gradients and feedback weights. Hence, we will refer to the result from (28) as the normalised real time recurrent learning (NRTRL) algorithm.

Due to the relationship between the a posteriori learning in the RNNs, and the optimal adaptive learning rate, obtained by the NRTRL algorithm, the performance of such a learning algorithm is expected to be better than the performance of the corresponding RTRL algorithm with the fixed learning rate [14,16].

Although the derived optimal learning rate does not include any constraint on the nonlinear activation function of a neuron, the slope of the nonlinear activation function and the learning rate are not independent [17]. It is therefore desirable to preserve contractivity of the nonlinear activation function of a neuron.

It should be noted that depending on the chosen sigmoid activation function (logistic, tanh, etc.), the term in the denominator of η_{OPT} (28) might need to have a small positive constant added. This also complies with the properties of higher order derivatives of sigmoid activation functions, where the truncated term in the Taylor series expansion is always positive.

6. Convergence of the NRTRL algorithm

Although the optimal learning rate for the RTRL algorithm is given by (28), the algorithm should converge for a range of values of η. Since we are dealing with nonlinear adaptive filters for prediction of nonstationary signals, no assumptions commonly used in the analysis of convergence of linear adaptive filters, such as, Gaussianity, wide sense stationarity, or independence between the learning rate and the weights in the network, are allowed. Hence, we first consider the asymptotic convergence of the magnitude of the instantaneous prediction error.

Consider again (26), written as

$$\begin{aligned} e(k+1) &= e(k) - 2\eta(k)e(k)\sum_{i=1}^{N}\|\boldsymbol{\Pi}_1^{(i)}(k)\|_2^2 \\ &= e(k)\left[1 - 2\eta(k)\sum_{i=1}^{N}\|\boldsymbol{\Pi}_1^{(i)}(k)\|_2^2\right]. \end{aligned} \tag{29}$$

An adaptive process converges asymptotically if [10]

$$|e(k)| \to 0 \quad \text{as } k \to \infty, \tag{30}$$

which gives

$$|e(k+1)| = \left| e(k)\left[1 - 2\eta(k)\sum_{i=1}^{N} \|\boldsymbol{\Pi}_1^{(i)}(k)\|_2^2\right]\right| \leqslant |e(k)|\left|1 - 2\eta(k)\sum_{i=1}^{N} \|\boldsymbol{\Pi}_1^{(i)}(k)\|_2^2\right|. \tag{31}$$

This will converge uniformly to zero for $k \to \infty$ if and only if [4,26]

$$\left|1 - 2\eta(k)\sum_{i=1}^{N} \|\boldsymbol{\Pi}_1^{(i)}(k)\|_2^2\right| < 1. \tag{32}$$

Hence, the bounds on the learning rate $\eta(k)$ which provide convergence of the NRTRL algorithm are

$$0 < \eta(k) < \frac{1}{\sum_{i=1}^{N} \|\boldsymbol{\Pi}_1^{(i)}(k)\|_2^2}. \tag{33}$$

6.1. Mean squared error convergence and convergence bounds

Following the approach from [3], we derive the mean squared error convergence conditions and the bounds on convergence of the proposed algorithm. In order to introduce noise in the system, observe the relationship between the NLMS and NRTRL, and reduce complexity of the derivation, we consider a single-neuron adaptive recurrent perceptron. Let us first introduce noise into the system by assuming

$$s(k) = q(k) + \Phi(\tilde{\boldsymbol{w}}^{\mathrm{T}}\boldsymbol{u}(k)), \tag{34}$$

where $\tilde{\boldsymbol{w}}(k)$ are some optimal filter weights. Consequently,

$$e(k) = q(k) + \Phi(\tilde{\boldsymbol{w}}^{\mathrm{T}}\boldsymbol{u}(k)) - \Phi(\boldsymbol{w}^{\mathrm{T}}(k)\boldsymbol{u}(k)). \tag{35}$$

The weight update equation now becomes

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + 2\eta(k)q(k)\boldsymbol{\Pi}(k) + 2\eta(k)\Phi(\tilde{\boldsymbol{w}}^{\mathrm{T}}(k)\boldsymbol{u}(k))\boldsymbol{\Pi}(k) - 2\eta(k)\Phi(\boldsymbol{w}^{\mathrm{T}}(k)\boldsymbol{u}(k))\boldsymbol{\Pi}(k). \tag{36}$$

If we introduce the misalignment vector as $\boldsymbol{v}(k) = \boldsymbol{w}(k) - \tilde{\boldsymbol{w}}$ and subtract $\tilde{\boldsymbol{w}}$ from either side of (36), we have

$$\boldsymbol{v}(k+1) = \boldsymbol{v}(k) + 2\eta(k)q(k)\boldsymbol{\Pi}(k) - 2\eta(k)[\Phi(\boldsymbol{w}^{\mathrm{T}}(k)\boldsymbol{u}(k)) - \Phi(\tilde{\boldsymbol{w}}^{\mathrm{T}}\boldsymbol{u}(k))]\boldsymbol{\Pi}(k). \tag{37}$$

For a contractive Φ, the term in the square brackets in (37) is bounded from above by $\alpha\|\boldsymbol{u}^{\mathrm{T}}(k)\boldsymbol{v}(k)\|$, $0 < \alpha \leqslant 1$. To improve clarity, let us restrict the analysis to an approximation $\boldsymbol{\Pi}(k) \approx \Phi'(\boldsymbol{w}^{\mathrm{T}}(k)\boldsymbol{u}(k))\boldsymbol{u}(k)$, which does not affect the generality of the result. Now, we have

$$\boldsymbol{v}(k+1) \leqslant \boldsymbol{v}(k) + 2\eta(k)q(k)\Phi'(\mathrm{net}(k))\boldsymbol{u}(k) - 2\eta(k)\boldsymbol{u}^{\mathrm{T}}(k)\boldsymbol{v}(k)\alpha\Phi'(\mathrm{net}(k))\boldsymbol{u}(k). \tag{38}$$

Providing contractivity of Φ, $\Phi'(\mathrm{net}(k))$ is bounded as $0 < |\Phi'(\mathrm{net}(k))| \leqslant 1$ [15], and can be replaced by its bound $\Phi'(\cdot) < \gamma \leqslant 1$.

To derive the upper bound for the RTRL, we include standard assumptions, such as the zero mean noise assumption, and the independence assumption between η, $\boldsymbol{u}$, and $\boldsymbol{v}$, and the above derived bounds, we consider the homogeneous part of (38)

$$E[\boldsymbol{v}(k+1)] = E[\boldsymbol{v}(k)]E[\boldsymbol{I} - 2\gamma\eta\boldsymbol{u}(k)\boldsymbol{u}^{\mathrm{T}}(k)\alpha], \tag{39}$$

where $E[\cdot]$ is the expectation operator. For convergence,

$$0 < E[\|\boldsymbol{I} - 2\gamma\eta(k)\boldsymbol{u}(k)\boldsymbol{u}^{\mathrm{T}}(k)\alpha\|] < 1 \tag{40}$$

which for the upper limit of α and γ gives

$$0 < \eta(k) < E\left[\frac{1}{\boldsymbol{u}^{\mathrm{T}}(k)\boldsymbol{u}(k)}\right]. \tag{41}$$

This means that the IIR version of the NLMS algorithm [22] is the upper bound for the single-neuron RTRL algorithm. The mean-square and steady-state mean-square convergence analysis can be derived in the same spirit, and shown to be bounded from above by the NRTRL.

7. Experimental result

An experiment was undertaken to support the above analysis. For a simple single-neuron RNN,

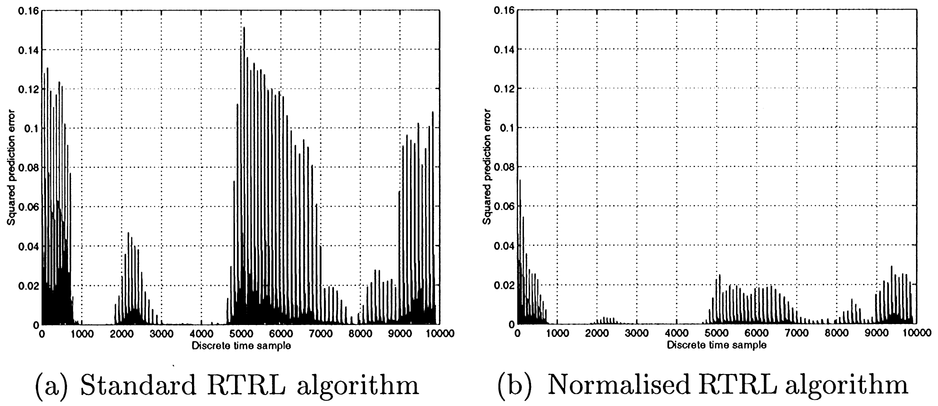

Fig. 2. Squared instantaneous prediction errors for the RTRL and NRTRL algorithms.

a comparison is made between the gradient learning with the fixed learning rate, and the NRTRL algorithm. The initialisation procedure for both experiments was the same, i.e. epochwise with 100 epochs of 100 samples, with the fixed learning rate. After that, the squared instantaneous prediction errors were calculated for both the standard RTRL and the NRTRL algorithm. The signal on which the simulation was performed was speech, because of its nonstationary and nonlinear behaviour. The content of the speech signal was *Oak is strong and* …, whose sampled version can be found on the author's WWW homepage [1]. Fig. 2 shows the instantaneous squared prediction errors for the RTRL and NRTRL. Indeed, the NRTRL algorithm from Fig. 2(b), clearly achieves significantly better performance than the RTRL algorithm (Fig. 2(a)). To quantify this, for the measure of performance in the form of the standard prediction gain, as defined by the ratio of the original signal to prediction error variances, the NRTRL achieved 7 dB better performance than the RTRL algorithm.

8. Conclusions

We have derived an optimal adaptive learning rate real time recurrent learning (RTRL) algorithm for continually running fully connected recurrent neural networks (RNNs). The algorithm normalises the learning rate of the RTRL and is hence referred to as the normalised RTRL (NRTRL) algorithm. The algorithm is optimal in the sense that it minimises the instantaneous squared error at the output neuron for every time interval while the network is running. No assumptions on the input signal, such as Gaussianity, or wide sense stationarity, or statistical independence between the learning rate and the weights in the network are required. The NRTRL algorithm, which is derived based upon the Taylor series expansion of the instantaneous output error, is shown to exhibit behaviour correspondent to that of the normalised LMS (NLMS) algorithm. The NRTRL is stabilised by the $\mathscr{L}_2$ norm of the input data vector and local gradients in the network, and converges for a contractive nonlinear activation function of a neuron. The bound on the range of adaptive learning rates which preserve convergence is shown to be the reciprocal of the $\mathscr{L}_2$ norm of gradients at the output neuron. The additional computational complexity involved is not significant, when compared to the entire computational complexity of the RTRL algorithm, which makes the NRTRL suitable for real time applications. Simulation results support the analysis.

References

[1] http://www.dsp.ee.ic.ac.uk/ ~ mandic.
[2] T. Aboulnasr, K. Mayyas, A robust variable step-size LMS-type algorithm: analysis and simulations, IEEE Trans. Signal Process. 45 (3) (1997) 631–639.

[3] J.A. Chambers, W. Sherliker, D.P. Mandic, A normalised gradient algorithm for an adaptive recurrent perceptron, ICASSP-2000, 2000, in press.

[4] P.G. Ciarlet, Introduction to Numerical Linear Algebra and Optimization, Cambridge University Press, Cambridge, 1989.

[5] S.C. Douglas, A family of normalized LMS algorithms, IEEE Signal Process. Lett. 1 (3) (1994) 49–51.

[6] S.C. Douglas, A. Cichocki, On-line step-size selection for training of adaptive systems, IEEE Signal Process. Mag. 14 (6) (1997) 45–46.

[7] J.B. Evans, P. Xue, B. Liu, Analysis and implementation of variable step size adaptive algorithms, IEEE Trans. Signal Process. 41 (8) (1993) 2517–2535.

[8] P.E. Gill, W. Murray, M.H. Wright, Practical Optimization, Academic Press, London, 1981.

[9] S. Haykin, Neural networks – a comprehensive foundation, Prentice-Hall, Englewood Cliffs, NJ, 1994.

[10] S. Haykin, Adaptive Filter Theory, 3rd Edition, Prentice-Hall, Englewood Cliffs, NJ, 1996.

[11] R.A. Jacobs, Increased rates of convergence through learning rate adaptation, Neural Networks 1 (1988) 295–307.

[12] R.H. Kwong, E.W. Johnston, A variable step size LMS algorithm, IEEE Trans. Signal Process. 40 (7) (1992) 1633–1641.

[13] D.G. Luenberger, Optimization by Vector Space Methods, Wiley, New York, 1969.

[14] D.P. Mandic, J.A. Chambers, A posteriori real time recurrent learning schemes for a recurrent neural network based non-linear predictor, IEEE Proc. – Vision, Image and Signal Process. 145 (6) (1998) 365–370.

[15] D.P. Mandic, J.A. Chambers, Global asymptotic stability of nonlinear relaxation equations realised through a recurrent perceptron, Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP-99), Vol. 2, 1999, pp. 1037–1040.

[16] D.P. Mandic, J.A. Chambers, Relations between the a priori and a posteriori errors in nonlinear adaptive neural filters, Neural Comput. 2000, in press.

[17] D.P. Mandic, J.A. Chambers, Relationship between the slope of the activation function and the learning rate for the RNN, Neural Comput. 11 (5) (1999) 1069–1077.

[18] D.P. Mandic, J.A. Chambers, Towards an optimal learning rate for backpropagation, Neural Process. Lett. 11 (1) (2000) 1–5.

[19] V.J. Mathews, Z. Xie, A stochastic gradient adaptive filter with gradient adaptive step size, IEEE Trans. Signal Process. 41 (6) (1993) 2075–2087.

[20] K.S. Narendra, K. Parthasarathy, Identification and control of dynamical systems using neural networks, IEEE Trans. Neural Networks 1 (1) (1990) 4–27.

[21] F.J. Pineda, Generalization of backpropagation to recurrent neural networks, Phys. Rev. Lett. 59 (1987) 2229–2232.

[22] P.A. Regalia, Adaptive IIR filtering in Signal Processing and Control, Marcel Dekker, New York, 1994.

[23] T.J. Shan, T. Kailaith, Adaptive algorithms with an automatic gain control feature, IEEE Trans. Acoust. Speech Signal Process. 35 (1) (1988) 122–127.

[24] P. Werbos, Backpropagation through time: What it does and how to do it, Proc. IEEE 78 (10) (1990) 1550–1560.

[25] R. Williams, D. Zipser, A learning algorithm for continually running fully recurrent neural networks, Neural Comput. 1 (1989) 270–280.

[26] E. Zeidler, Nonlinear Functional Analysis and its Applications, Vol. 1, Fixed-point theorems, Springer, New York, 1986.