Data-Reusing Recurrent Neural Adaptive Filters

Danilo P. Mandic
d.mandic@uea.ac.uk
School of Information Systems, University of East Anglia, Norwich, NR4 7TJ, U.K.

A class of data-reusing learning algorithms for real-time recurrent neural networks (RNNs) is analyzed. The analysis is undertaken for a general sigmoid nonlinear activation function of a neuron for the real time recurrent learning training algorithm. Error bounds and convergence conditions for such data-reusing algorithms are provided for both contractive and expansive activation functions. The analysis is undertaken for various configurations that are generalizations of a linear structure infinite impulse response adaptive filter.

1 Introduction

Recurrent neural networks (RNN)s represent an emerging technique in nonlinear adaptive signal processing. They are also suitable for implementing nonlinear autoregressive moving average (NARMA) models (Connor, Martin, & Atlas, 1994; Nerrand, Roussel-Ragot, Peresonnaz, & Dreyfus, 1993; Nerrand, Roussel-Ragot, Urbani, Personnaz, & Dreyfus, 1994). However, RNNs for real-time adaptive filtering applications, such as prediction, encounter problems due to the slow convergence of their gradient adaptive algorithms (Bengio, Simard, & Frasconi, 1994). The so-called data-reusing algorithms, which have been considered for linear adaptive filters, offer an increased convergence rate as compared to standard algorithms (Treichler, Johnson, & Larimore, 1987; Roy & Shynk, 1989; Schnaufer & Jenkins, 1993).

For a recurrent perceptron, which is a nonlinear version of an infinite impulse response (IIR) linear filter, whose nonlinear activation function is Φ, the output of a neural adaptive filter y is given by

$$y(k) = \Phi(\mathbf{u}^T(k)\mathbf{w}(k)), \tag{1.1}$$

where $\mathbf{u}(k)$, $\mathbf{w}(k)$, and $(\cdot)^T$ denote, respectively, the input vector, weight vector, and vector transpose operator. Since the updated weight vector $\mathbf{w}(k+1)$ is available before the next input vector $\mathbf{u}(k+1)$, a new estimate $\bar{y}$, which is also known as an a posteriori estimate, can be calculated as (Mandic &

Chambers, 1998)

$$\bar{y}(k) = \Phi(\mathbf{u}^T(k)\mathbf{w}(k+1)). \tag{1.2}$$

The corresponding instantaneous output errors for the two cases above are given respectively as $e(k) = d(k) - y(k)$, and $\bar{e}(k) = d(k) - \bar{y}(k)$, where $d(k)$ is some teaching signal.

This technique can be repeated and is called a data-reusing technique (Roy & Shynk, 1989). A data-reusing algorithm for the recurrent nonlinear case can hence be expressed as

$$\begin{aligned} \mathbf{w}_{i+1}(k) &= \mathbf{w}_i(k) - \eta\nabla_{\mathbf{w}_i(k)}E(e_i(k)) \\ e_i(k) &= d(k) - \Phi(\mathbf{u}^T(k)\mathbf{w}_i(k)), \qquad i = 1, \ldots, L, \end{aligned} \tag{1.3}$$

where the cost function $E(e_i(k))$ is typically $E_i(k) = \frac{1}{2}e_i^2(k)$, and index i denotes the ith iteration of the algorithm 1.3 and η is the learning rate. Nerrand et al. (1993) provide an extensive study of learning algorithms for neural networks for nonlinear adaptive filtering. A case with only one iteration of equation 1.3 is addressed in Mandic and Chambers (2000b).

We wish to preserve the useful feature of a data-reusing algorithm that the magnitude of an output error uniformly converges along the iteration 1.3, that is,

$$|e_{i+1}(k)| \leq \gamma\,|e_i(k)|, \quad 0 < \gamma < 1, \quad i = 1, \ldots, L, \tag{1.4}$$

which represents a constraint on equation 1.3. It follows that for $L = 1$, algorithm 1.3 reduces to the standard gradient-descent algorithm: $\mathbf{w}_1(k) = \mathbf{w}(k)$. The $(L+1)$th iteration of equation 1.3 provides the values of quantities that relate to the time instant $(k+1)$, that is, $\mathbf{w}_{L+1}(k) = \mathbf{w}(k+1)$. Unlike for the case of linear adaptive filters (Roy & Shynk, 1989; Schnaufer & Jenkins, 1993), the data-reusing approach for RNNs working as nonlinear adaptive filters depends also on the characteristics of the nonlinear activation function of a neuron, that is, whether it is a contraction.

Here, an extension to the approach of Mandic and Chambers (2000b) is provided and considers bounds on the ith data-reusing prediction error $e_i(k)$, $i = 1, 2, \ldots, L$ for recurrent neural networks, starting from a recurrent perceptron, to a fully connected recurrent neural network. Both the contractive and expansive nonlinear activation function of a neuron are addressed. In this context, the range of values for the learning rate that provide convergence and the change undergone by the weights of a network in the data-reusing case are analyzed.

2 Bounds on the Data-Reusing Error Adaptation for Recurrent Neural Networks

From equation 1.3, a gradient-descent algorithm for a single-neuron recurrent network (recurrent perceptron) is given by Haykin (1999),

$$\begin{aligned} \mathbf{w}_{i+1}(k) &= \mathbf{w}_i(k) + \eta(k)e_i(k)\boldsymbol{\Pi}_i(k) \\ e_i(k) &= d(k) - \Phi(\mathbf{u}^T(k)\mathbf{w}_i(k)), \quad i = 1, \ldots, L, \end{aligned} \tag{2.1}$$

where $\mathbf{w}_i(k)$ is the weight vector at the ith iteration of equation 2.1, η is the learning rate, $\mathbf{u}(k)$ is the input vector consisting of external data, feedback, and bias, $d(k)$ is some desired signal, $\boldsymbol{\Pi}_i(k)$ is a gradient (sensitivities) vector, and $e_i(k)$ is the prediction error after the ith iteration of the data-reusing algorithm, 2.1, at the time instant k. A brief overview of the real time recurrent learning (RTRL) algorithm is given in the appendix.

An insight into the RTRL algorithm and algorithm 2.1 shows that the standard gradient algorithm that runs in discrete time k represents an outer loop, whereas the data-reusing part, equation 2.1, which runs iteratively on the input data from the discrete time instant k, represents an inner loop of the whole algorithm. This way, the data-reusing algorithm described here is a combination of a recursive and iterative algorithm. In Nerrand et al. (1993, 1994) and Nerrand, Roussel-Ragot, Personnaz, and Dreyfus (1991), a unifying concept of algorithms for training neural networks is provided. In their taxonomy, the data-reusing algorithms presented here are referred to as unidirected–directed algorithms. Starting from the last iteration in equation 2.5, for $i = L$, we obtain the final data-reusing weight update as

$$\begin{aligned} \mathbf{w}(k+1) = \mathbf{w}_{L+1}(k) &= \mathbf{w}_L(k) + \eta(k)e_L(k)\boldsymbol{\Pi}_L(k) \\ &= \mathbf{w}_{L-1}(k) + \eta(k)e_{L-1}(k)\boldsymbol{\Pi}_{L-1}(k) \\ &\quad + \eta(k)e_L(k)\boldsymbol{\Pi}_L(k) \\ &= \mathbf{w}(k) + \sum_{i=1}^{L} \eta(k)e_i(k)\boldsymbol{\Pi}_i(k). \end{aligned} \tag{2.2}$$

The consecutive output errors in the data-reusing iteration are expressed as

$$\begin{aligned} e_1(k) &= d(k) - \Phi(\mathbf{u}^T(k)\mathbf{w}(k)) \\ e_2(k) &= d(k) - \Phi(\mathbf{u}^T(k)\mathbf{w}_2(k)) \\ &\cdots\cdots\cdots \\ e_L(k) &= d(k) - \Phi(\mathbf{u}^T(k)\mathbf{w}_L(k)). \end{aligned} \tag{2.3}$$

The instantaneous error at the output neuron can be further expressed as

$$\begin{aligned} e_i(k) &= d(k) - \Phi(\mathbf{u}^T(k)\mathbf{w}_i(k)) \\ &= [d(k) - \Phi(\mathbf{u}^T(k)\mathbf{w}_{i-1}(k))] \\ &\quad - [\Phi(\mathbf{u}^T(k)\mathbf{w}_i(k)) - \Phi(\mathbf{u}^T(k)\mathbf{w}_{i-1}(k))], \end{aligned} \tag{2.4}$$

clearly indicating the dependence of the ith data-reusing error on the features of a nonlinear activation function[1] of the neuron, Φ. Premultiplying the first equation in equation 2.1 by $\mathbf{u}^T(k)$ and applying the nonlinear activation function Φ on either side, we obtain

$$\Phi(\mathbf{u}^T(k)\mathbf{w}_{i+1}(k)) = \Phi(\mathbf{u}^T(k)\mathbf{w}_i(k) + \eta(k)e_i(k)\mathbf{u}^T(k)\mathbf{\Pi}_i(k)). \tag{2.5}$$

Further analysis depends on whether Φ is a contraction or an expansion. Brief insight into the contraction mapping theorem (CMT) and its consequences on stability in a recurrent perceptron is given in the appendix.

2.1 The Case of a Contractive Activation Function. For a broad class of contractive sigmoid activation function we have (Mandic & Chambers, 2000b)

$$\Phi(a + b) \leq \Phi(a) + \Phi(b). \tag{2.6}$$

Notice that both $e(k)$ and $e_i(k), i = 2, \ldots, L$ have the same sign (Roy & Shynk, 1989; Douglas & Rupp, 1997), as seen in Figure 1.

From equation 1.3, the direction of the vectors $\Delta\mathbf{w}_i(k)$ can be assumed the same as the direction of the input vector $\mathbf{u}(k)$. As a gradient-descent algorithm gives only an approximate solution to the optimization problem, the quadratic surface defined by $e^2(k)$ has a solution set that is a linear variety

[1] Clearly, equation 2.4 can be expressed as

$$\begin{aligned} e_i(k) &= [d(k) - \Phi(\mathbf{u}^T(k)\mathbf{w}_{i-1}(k))] - [\Phi(\mathbf{u}^T(k)\mathbf{w}_i(k)) - \Phi(\mathbf{u}^T(k)\mathbf{w}_{i-1}(k))] \\ &= e_{i-1}(k) - [\Phi(\mathbf{u}^T(k)\mathbf{w}_i(k)) - \Phi(\mathbf{u}^T(k)\mathbf{w}_{i-1}(k))]. \end{aligned}$$

By the CMT (Gill, Murray, & Wright, 1981; (see the Appendix)

$$\exists \xi \in (\mathbf{u}^T(k)\mathbf{w}_{i-1}(k), \mathbf{u}^T(k)\mathbf{w}_i(k)),$$

such that

$$|\Phi(\mathbf{u}^T(k)\mathbf{w}_i(k)) - \Phi(\mathbf{u}^T(k)\mathbf{w}_{i-1}(k))| \leq \Phi'(\xi)|\mathbf{u}^T(k)\Delta\mathbf{w}_i(k)|,$$

which connects the output errors at the ith and $(i-1)$th iteration, the first derivative of the nonlinear activation function of neuron and the norm of the input data. This relationship is elaborated later in the article. A logistic sigmoid function is assumed, although the results are general.

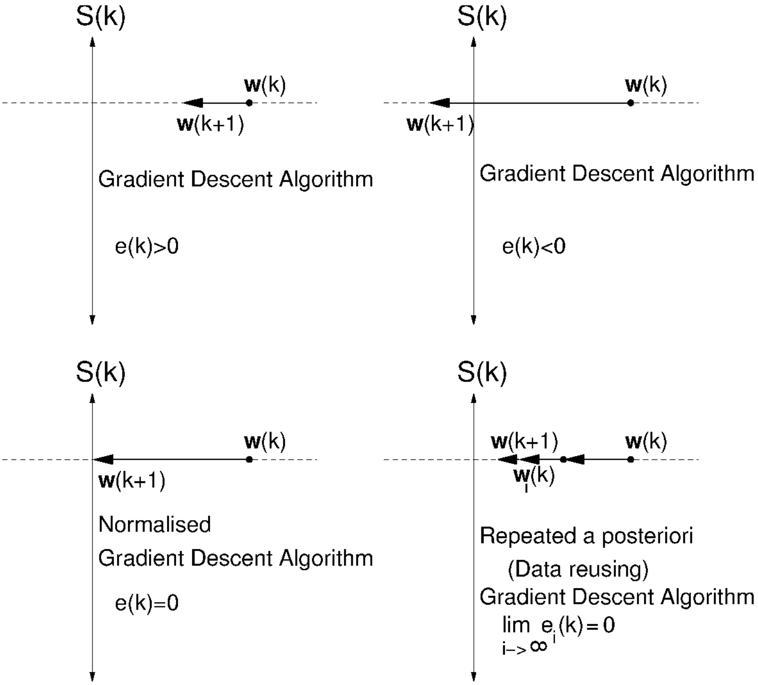

Figure 1: Geometric interpretation of data-reusing techniques.

instead of a single minimum. Hence, the solution space is a hypersurface (Schnaufer & Jenkins, 1993), whose dimension is one less that the space on which it rests, and vector $\mathbf{u}(k)$ is perpendicular to the solution hypersurface $S(k)$. Figure 1 gives a geometric interpretation of the relation between the standard gradient, normalized gradient (Mandic & Chambers, 2000a), and data-reusing gradient algorithm for on-line training of neural nonlinear adaptive filters.

From Figure 1, it is evident that repeating an a posteriori technique for a sufficient number of iterations approaches the normalized gradient-based algorithm (Mandic & Chambers, 2000a).

Theorem 1. *The lower bound for the data-reusing estimation error obtained by algorithm 2.1 with L iterations and a contractive nonlinear activation function* Φ *is*

$$e_{L+1}(k) > [1 - \eta(k)\mathbf{u}^T(k)\mathbf{\Pi}(k)]^L e(k), \tag{2.7}$$

where $\mathbf{\Pi}(k)$ *is such that* $\max_{i=1,\ldots,L}(\mathbf{u}^T(k)\mathbf{\Pi}_i(k))$ *is obtained.*

Proof. Let $a = \mathbf{u}^T(k)\mathbf{w}_i(k)$ and $b = \eta(k)e(k)\mathbf{u}^T(k)\mathbf{\Pi}_i(k)$. Applying equation 2.6 to 2.5, and subtracting $d(k)$ from both sides of the resulting equation, and recognizing that for Φ a contraction, $|\Phi(\xi)| < |\xi|, \forall \xi \in \mathbb{R}$, we obtain

$$e_{i+1}(k) > [1 - \eta(k)\mathbf{u}^T(k)\mathbf{\Pi}_i(k)]e_i(k). \tag{2.8}$$

Notice that the whole process is a fixed-point iteration around the fixed point defined by the information vector $\mathbf{u}(k)$, and hence the sequence $\mathbf{u}^T(k)\mathbf{\Pi}_i(k)$, $i = 1, \dots, L$ has its maximum $\mathbf{u}^T(k)\mathbf{\Pi}(k)$ along the iteration. This yields

$$\begin{aligned} e_i(k) &> [1 - \eta(k)\mathbf{u}^T(k)\mathbf{\Pi}(k)]e_{i-1}(k) \\ &\cdots\cdots\cdots \\ e_i(k) &> [1 - \eta(k)\mathbf{u}^T(k)\mathbf{\Pi}(k)]^{i-1}e(k) \end{aligned} \tag{2.9}$$

and equation 2.8 finally becomes

$$e_{L+1}(k) > [1 - \eta(k)\mathbf{u}^T(k)\mathbf{\Pi}(k)]^{L}e(k), \tag{2.10}$$

which is the lower bound for the data-reusing algorithm with a contractive activation function of a neuron.

For the algorithm given in equation 2.1 with the constraint $|e_i(k)| < |e_{i-1}(k)|$ to be feasible, the term $[1 - \eta(k)\mathbf{u}^T(k)\mathbf{\Pi}(k)]$ in equation 2.9 must have a norm of less than unity, that is, it must be a contraction operator (Bharucha-Reid, 1976). Notice that for $L \to \infty$, the error from equation 2.10 becomes $e_\infty(k) = 0$, that is, after an infinite number of iterations, this algorithm becomes a normalized RTRL (NRTRL) algorithm, as derived in Mandic and Chambers (2000a). In that case, the whole procedure is a fixed-point iteration, and the necessary condition that guarantees convergence is $|1 - \eta(k)\mathbf{u}^T(k)\mathbf{\Pi}(k)| < 1$ (Mandic & Chambers, 1998). The following corollary gives the range of the learning rate $\eta(k)$ so that the contraction mapping properties are satisfied.

Corollary 1. *The range allowed for the learning rate $\eta(k)$ in a data-reusing algorithm, 2.1, to ensure convergence for the conditions given in theorem 1 is*

$$0 < \eta(k) < \frac{2}{|\mathbf{u}^T(k)\mathbf{\Pi}(k)|}. \tag{2.11}$$

It is assumed that the value of the learning rate $\eta(k)$ is held fixed during the fixed-point iteration.

2.2 The Case of an Expansive Activation Function. If function Φ is an expansion, then (Mandic & Chambers, 2000b)

$$\Phi(a+b) \geq \Phi(a) + \Phi(b). \tag{2.12}$$

This is not desirable, since the second term in equation 2.4 can grow without bound. However, an analog statement to that of theorem 1 can be expressed, replacing the sign $>$ in equation 2.7 by $<$. Although this provides an upper bound, it is not of practical significance, since the errors grow with the number of iterations due to the expansion of Φ in equation 2.12. An account of the relationship between the norm of a weight matrix, learning rate and a slope of the activation function is given in Mandic and Chambers (1999b).

2.3 The Case of a Linear Activation Function. In this case, the problem degenerates into the problem of data-reusing linear adaptive filters, which are extensively studied in Roy and Shynk (1989), Schnaufer and Jenkins (1993), and Sheu et al. (1992).

3 A General RNN

In the case of a general RNN, we have a weight matrix $\mathbf{W}$ comprising the weight vectors $\mathbf{w}$ of individual neurons. For the case of a single-output network, as is common in nonlinear prediction, the gradient matrix of the first neuron $\mathbf{\Pi}^1$ is of special interest. Due to the similarity of expressions for a recurrent perceptron and a general RNN, the equation of interest now becomes (see the appendix)

$$\Delta\mathbf{W}(k) = \eta(k)e(k)\frac{\partial y_1(k)}{\partial \mathbf{W}(k)} = \eta(k)e(k)\mathbf{\Pi}^1(k), \tag{3.1}$$

where $\mathbf{\Pi}^1(k)$ represents the matrix of gradients at the output neuron with respect to $\mathbf{W}(k)$. The correction to the weight vector of the jth neuron, at the time instant k, becomes

$$\Delta\mathbf{w}^j(k) = \eta(k)e(k)\mathbf{\Pi}^{1(j)}(k), \tag{3.2}$$

where $\mathbf{\Pi}^{1(j)}$ represents the jth row of the gradient matrix $\mathbf{\Pi}^1(k)$. Hence, from the analysis for a recurrent perceptron and defining the gradients $\mathbf{\Pi}^{1(1)}(k)$ in a way similar to that in theorem 1, we have theorem 2.

Theorem 2. *For a data-reusing algorithm based on equation 2.1 and a contractive nonlinear activation function Φ, the lower bound for the error obtained by equation 2.1 and the range of the allowed step size for which the algorithm converges* *are, respectively,*

$$e_{L+1}(k) > [1 - \eta(k)\mathbf{u}^T(k)\mathbf{\Pi}^{1(1)}(k)]^L e(k) \tag{3.3}$$

$$0 < \eta(k) < \frac{2}{|\mathbf{u}^T(k)\mathbf{\Pi}^{1(1)}(k)|}. \tag{3.4}$$

4 On the Weight Change in the Data-Reusing Algorithm

For simplicity, we consider the case of a recurrent perceptron. After applying the data-reusing algorithm L times, from equation 2.9, we have

$$\begin{aligned}\sum_{i=1}^{L} e_i(k) &> \sum_{i=1}^{L}[1 - \eta(k)\mathbf{u}^T(k)\mathbf{\Pi}(k)]^{i-1}e(k)\\ &= \frac{1 - [1 - \eta(k)\mathbf{u}^T(k)\mathbf{\Pi}(k)]^L}{\eta(k)\mathbf{u}^T(k)\mathbf{\Pi}(k)}e(k).\end{aligned} \tag{4.1}$$

This sum converges for a contractive activation function Φ and $|1-\eta(k)\mathbf{u}^T(k)$ $\mathbf{\Pi}(k)| < 1$. Now, from equations 2.2 and 4.1, we obtain the amount for which the weights change after L iterations of equation 2.1 (element by element):

$$\Delta\mathbf{w}(k) < \frac{1 - [1 - \eta(k)\mathbf{u}^T(k)\mathbf{\Pi}(k)]^L}{\mathbf{u}^T(k)\mathbf{\Pi}(k)}e(k)\mathbf{\Pi}(k). \tag{4.2}$$

A similar relationship can be obtained for the case of a general Williams-Zipser type RNN.

5 Convergence of the Data-Reusing Approach

In the area of linear adaptive filters, the analysis of convergence consists of the analysis of the convergence in the mean, convergence in the mean square, and convergence in the steady state. However, in the nonlinear case, a Wiener solution does not generally exist, and hence, the convergence is mainly considered by Lyapunov stability (DeRusso, Roy, Close, & Desrochers, 1998; Zurada & Shen, 1990), or through contraction mapping. Here, due to the assumption that the standard and the data-reusing errors have the same sign throughout the iteration, convergence of the data-reusing prediction error is defined by convergence of the underlying learning algorithm for the standard error, given a contractive activation function of a neuron. Additional stability and robustness of the data-reusing algorithms is ensured due to the denominators of the corresponding relationships between the two errors, which are greater than unity (Mandic & Chambers, 1999a).

From equation 4.2 we have

$$\mathbf{w}(k+1) < \mathbf{w}(k) + \frac{1-[1-\eta(k)\mathbf{u}^T(k)\boldsymbol{\Pi}(k)]^L}{\mathbf{u}^T(k)\boldsymbol{\Pi}(k)} e(k)\boldsymbol{\Pi}(k). \tag{5.1}$$

Introducing the weight error vector $\mathbf{v}$ as $\mathbf{v}(k) = \mathbf{w}(k) - \mathbf{w}^*(k)$, where $\mathbf{w}^*(k)$ is some optimal value, and using the contractivity condition of the activation function, we have

$$e(k) \approx e^*(k) - \Phi(\mathbf{u}^T(k)\mathbf{v}(k)). \tag{5.2}$$

Hence, we obtain

$$\begin{aligned}\mathbf{v}(k+1) < \mathbf{v}(k) &+ \frac{1-[1-\eta(k)\mathbf{u}^T(k)\boldsymbol{\Pi}(k)]^L}{\mathbf{u}^T(k)\boldsymbol{\Pi}(k)} e^*(k)\boldsymbol{\Pi}(k)\\ &- \frac{1-[1-\eta(k)\mathbf{u}^T(k)\boldsymbol{\Pi}(k)]^L}{\mathbf{u}^T(k)\boldsymbol{\Pi}(k)} \Phi(\mathbf{u}^T(k)\mathbf{v}(k))\boldsymbol{\Pi}(k).\end{aligned} \tag{5.3}$$

With the assumption of contractivity of Φ ($|\Phi(\xi)| < |\xi|, \quad \forall\xi$), the homogeneous part of equation 5.3 becomes

$$\mathbf{v}(k+1) = \left[I - \frac{1-[1-\eta(k)\mathbf{u}^T(k)\boldsymbol{\Pi}(k)]^L}{\mathbf{u}^T(k)\boldsymbol{\Pi}(k)} \boldsymbol{\Pi}(k)\mathbf{u}^T(k)\right]\mathbf{v}(k). \tag{5.4}$$

In order for equation 5.4 to converge, the term in square parentheses has to be a contraction mapping operator, which has already been shown. On the other hand, the data-reusing algorithms presented here have been shown to have a uniformly smaller prediction error than the standard ones, and hence they converge by continuity. Further details on convergence of standard algorithms can be found in Kuan and Hornik (1991), Bershad, Shynk, and Feintuch (1993a, 1993b), and Yuille and Kosowsky (1993), whereas convergence of normalized gradient-descent nonlinear algorithms is addressed in Mandic and Chambers (2000a).

6 Discussion

It has been shown that repeating the data-reusing algorithm leads to a normalized gradient algorithm; however, the NRTRL algorithm for recurrent neural networks derived in Mandic and Chambers (2000a) is sensitive to the characteristics of the input signals and the choice of the constant in the algorithm. One way to achieve a performance similar to that of the normalized algorithm is to use a contractive activation function of a neuron on the class of data-reusing algorithms (see Figure 1). Mandic and Chambers (1999b), illustrate the data-reusing approach for only one iteration (a posteriori). Other algorithms, such as the extended Kalman filter (EKF) training for recurrent neural networks, have been analyzed and compared with standard RTRL (Mandic, Baltersee, & Chambers, 1998). A comparison between NRTRL and the EKF algorithm shows that the NRTRL exhibits as good a performance as the EKF for nonlinear signals. Hence, the benefit of data-reusing techniques is that they achieve a near NRTRL performance in few iterations, albeit with an additional computational burden, as compared to the NRTRL training. Unlike the NRTRL, however, this class of algorithms does not suffer from stability problems for contractive nonlinear activation functions. Therefore, the computational complexity of the data-reusing RTRL increases with the number of iterations but is still of the order $\mathcal{O}(N^4)$ for a relatively small number of iterations. The computational complexity considerations and numerical examples for a simple data-reusing technique for RNNs are given in Mandic and Chambers (1998).

In Figure 2, the RTRL and data-reusing RTRL were compared for prediction of a benchmark nonlinear input given by Narendra and Parthasarathy (1990),

$$z(k) = \frac{z(k-1)}{1 + z^2(k-1)} + r^3(k), \tag{6.1}$$

where $r(k)$ was a normally distributed $\mathcal{N}(0, 1)$ white noise $n(k)$ passed through a stable AR filter given by

$$r(k) = 1.79r(k-1) - 1.85r(k-2) + 1.27r(k-3) - 0.41r(k-4) + n(k). \tag{6.2}$$

The logarithm of the averaged squared prediction error, obtained by a Monte Carlo simulation with 100 trials of the experiment for both contractive and expansive activation function, is shown in Figure 2.

The top part of Figure 2 shows the performance of a data-reusing algorithm for a recurrent perceptron with a contractive activation function for $L = 1$, $L = 2$, $L = 5$, and $L = 10$. The data-reusing algorithm outperformed the standard algorithm ($L = 1$). The performance of this algorithm improves with increasing order of the data-reusing iteration and saturates for large L, confirming the analysis and the diagram shown in Figure 1. The bottom part of Figure 2 shows the performance of a data-reusing algorithm for a recurrent perceptron with an expansive activation function for $L = 1$, $L = 3$, and $L = 10$. The performance deteriorates with the order of iteration, confirming the above analysis.

7 Conclusion

Relationships among the prediction error, learning rate, and slope of the nonlinear activation function of a neuron have been provided for a nonlinear adaptive filter realized by a recurrent neural network, trained with a data-reusing gradient algorithm. Such relationships establish a lower bound on

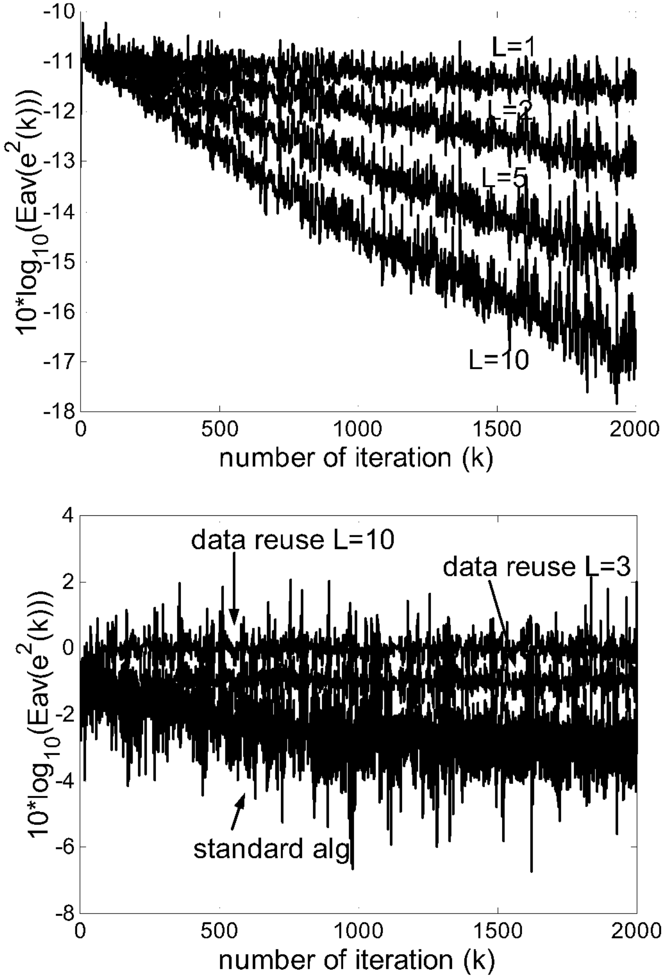

Figure 2: (Top) Performance of RTRL and data-reusing RTRL with $L = 2$, $L = 5$, and $L = 10$ for prediction of a nonlinear input for a contractive nonlinear activation function. (Bottom) Performance of RTRL and data-reusing RTRL with $L = 3$ and $L = 10$ for prediction of a nonlinear input for an expansive nonlinear activation function.

the error in recurrent neural predictors, whose instantaneous output error is uniformly smaller in magnitude along the data-reusing iteration. This has been achieved for the RTRL learning algorithm for a recurrent perceptron and a general RNN. The analysis is general, although it has been performed for a logistic sigmoid. It has been shown that convergence in this case is preserved for a nonlinear activation function exhibiting contractive behavior. Dynamical behavior of such data-reusing gradient-based algorithms lies between the standard and normalized gradient algorithm.

Appendix

A.1 The RTRL Algorithm. For nonlinear adaptive prediction of nonstationary signals through recurrent neural networks, the RTRL algorithm is best suited for real-time applications (Williams & Zipser, 1989; Haykin, 1994). The weight matrix update of a general RNN can be expressed as (Haykin, 1994)

$$\Delta \mathbf{W}(k) = \eta e(k) \frac{\partial y_1(k)}{\partial \mathbf{W}(k)} = \eta e(k) \mathbf{\Pi}^1(k), \tag{A.1}$$

where η is a learning rate, and $\mathbf{\Pi}^1(k)$ represents the matrix of gradients at the output neuron y_1 with respect to $\mathbf{W}(k)$.

To simplify the presentation, we introduce three new matrices—the $N \times (N+p+1)$ matrix $\mathbf{\Pi}^{(j)}(k)$, where p is the number of external input signals, the $N \times (N+p+1)$ matrix $\mathbf{U}_j(k)$, and the $N \times N$ diagonal matrix $\mathbf{F}(k)$—as

$$\mathbf{\Pi}^{(j)}(k) = \frac{\partial \mathbf{y}(k)}{\partial \mathbf{w}_j(k)}, \quad \mathbf{y} = [y_1(k), \ldots, y_N(k)], \quad j = 1, 2, \ldots, N \tag{A.2}$$

$$\mathbf{U}_j(k) = \begin{bmatrix} \mathbf{0} \\ \vdots \\ \mathbf{u}(k) \\ \vdots \\ \mathbf{0} \end{bmatrix} \leftarrow j\text{th row}, \quad j = 1, 2, \ldots, N \tag{A.3}$$

$$\mathbf{F}(k) = diag(\Phi'(\mathbf{u}^T(k)\mathbf{w}^{(1)}(k)), \ldots, \Phi'(\mathbf{u}^T(k)\mathbf{w}^{(N)}(k))). \tag{A.4}$$

Hence, the gradient updating equation regarding the recurrent neuron can be symbolically expressed as (Williams & Zipser, 1989; Haykin, 1994)

$$\mathbf{\Pi}^{(j)}(k+1) = \mathbf{F}(k)[\mathbf{U}_j(k) + \mathbf{\Pi}^{(j)}(k)\mathbf{W}_a(k)], \quad j = 1, 2, \ldots, N, \tag{A.5}$$

where $\mathbf{W}_a$ denotes the set of those entries in $\mathbf{W}$ that correspond to the feedback connections. The correction to the weight vector of the jth neuron, at the time instant k becomes

$$\Delta \mathbf{w}^{(j)}(k) = \eta(k) e(k) \mathbf{\Pi}^{1(j)}(k) \tag{A.6}$$

where $\mathbf{\Pi}^{1(j)}$ represents the jth row of the gradient matrix $\mathbf{\Pi}^1(k)$.

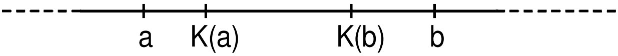

Figure 3: The contraction mapping.

A.2 Contraction Mapping and Nonlinear Activation Functions. By the CMT, function K is a contraction on $[a, b] \in \mathbb{R}$ if (Gill et al., 1981):

i) $x \in [a, b] \Rightarrow K(x) \in [a, b]$

ii) $\exists \gamma \ < 1 \in \mathbb{R}^{+} \quad s.t. \quad |K(x) - K(y)| \leq \gamma \, |x - y|, \quad \forall x, y \in [a, b],$

as shown in Figure 3. Using the mean value theorem, for $\forall x, y \in [a, b], \ \exists \xi \in (a, b)$ such that

$$|K(x) - K(y)| = |K'(\xi)(x - y)| = |K'(\xi)||x - y|. \tag{A.7}$$

Now, the clause $\gamma \ < \ 1$ in *ii*) becomes $\gamma \ \geq \ |K'(\xi)|, \quad \xi \ \in \ (a, b)$. For the example of the logistic nonlinear activation function of a neuron $\Phi(v) = \frac{1}{1+e^{-\beta v}}$, with slope $\beta, \gamma \ < 1 \quad \Leftrightarrow \quad \beta < 4$ is the condition for function Φ to be a contraction.

Acknowledgments

I thank Andrew Hanna of the University of East Anglia, Norwich, for his help in simulations and useful comments.

References

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, *5*(2), 157–166.

Bershad, N. J., Shynk, J. J., & Feintuch, P. L. (1993a). Statistical analysis of the single-layer backpropagation algorithm: Part I—Mean weight behaviour. *IEEE Transactions on Signal Processing*, *41*(2), 573–582.

Bershad, N. J., Shynk, J. J., & Feintuch, P. L. (1993b). Statistical analysis of the single-layer backpropagation algorithm: Part II—MSE and classification performance. *IEEE Transactions on Signal Processing*, *41*(2), 583–591.

Bharucha-Reid, A. T. (1976). Fixed point theorems in probabilistic analysis. *Bulletin of the American Mathematical Society*, *82*(5), 641–657.

Connor, J. T., Martin, R. D., & Atlas, L. E. (1994). Recurrent neural networks and robust time series prediction. *IEEE Transactions on Neural Networks*, *5*(2), 240–254.

DeRusso, P. M., Roy, R. J., Close, C. M., & Desrochers, A. A. (1998). *State variables for engineers* (2nd ed.). New York: Wiley.

Douglas, S. C., & Rupp, M. (1997). A posteriori updates for adaptive filters. In *Conference Record of the Thirty-First Asilomar Conference on Signals, Systems and Computers* (Vol. 2, pp. 1641–1645). Pacific Grove, CA.

Gill, P. E., Murray, W., & Wright, M. H. (1981). *Practical optimization*. London: Academic Press.

Haykin, S. (1994). *Neural networks—A comprehensive foundation*. Upper Saddle River, NJ: Prentice Hall.

Haykin, S. (1999). *Neural networks—A comprehensive foundation* (2nd ed.). Upper Saddle River, NJ: Prentice Hall.

Kuan, C.-M., & Hornik, K. (1991). Convergence of learning algorithms with constant learning rates. *IEEE Transactions on Neural Networks*, *2*(5), 484–489.

Mandic, D. P., Baltersee, J., & Chambers, J. A. (1998). Nonlinear prediction of speech with a pipelined recurrent neural network and advanced learning algorithms. In A. Prochazka, J. Uhlir, P. J. W. Rayner, & N. G. Kingsbury (Eds.), *Signal analysis and prediction* (pp. 291–309). Boston: Birkhauser.

Mandic, D. P., & Chambers, J. A. (1998). A posteriori real time recurrent learning schemes for a recurrent neural network based non-linear predictor. *IEE Proceedings—Vision, Image and Signal Processing*, *145*(6), 365–370.

Mandic, D. P., & Chambers, J. A. (1999a). A posteriori error learning in non-linear adaptive filters. *IEE Proceedings—Vision, Image and Signal Processing*, *146*(6), 293–296.

Mandic, D. P., & Chambers, J. A. (1999b). Relationship between the slope of the activation function and the learning rate for the RNN. *Neural Computation*, *11*(5), 1069–1077.

Mandic, D. P., & Chambers, J. A. (2000a). A normalised real time recurrent learning algorithm. *Signal Processing*, *80*(11), 1909–1916.

Mandic, D. P., & Chambers, J. A. (2000b). Relations between the a priori and a posteriori errors in nonlinear adaptive neural filters. *Neural Computation*, *12*(6), 1285–1292.

Narendra, K. S., & Parthasarathy, K. (1990). Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*, *1*(1), 4–27.

Nerrand, O., Roussel-Ragot, P., Personnaz, L., & Dreyfus, G. (1991). Neural network training schemes for non-linear adaptive filtering and modelling. In *Proceedings of the International Joint Conference on Neural Networks* (Vol. 1, pp. 61–66). Seattle, WA.

Nerrand, O., Roussel-Ragot, P., Personnaz, L., & Dreyfus, G. (1993). Neural networks and nonlinear adaptive filtering: Unifying concepts and new algorithms. *Neural Computation*, *5*, 165–199.

Nerrand, O., Roussel-Ragot, P., Urbani, D., Personnaz, L., & Dreyfus, G. (1994). Training recurrent neural networks: Why and how? An illustration in dynamical process modelling. *IEEE Transactions on Neural Networks*, *5*(2), 178–184.

Roy, S., & Shynk, J. J. (1989). Analysis of the data-reusing LMS algorithm. In *Proceedings of the 32nd Midwest Symposium on Circuits and Systems* (Vol. 2, pp. 1127–1130). Champaign, IL.

Schnaufer, B. A., & Jenkins, W. K. (1993). New data-reusing LMS algorithms for improved convergence. In *Conference Record of the Twenty-Seventh Asilomar Conference on Signals and Systems* (Vol. 2, pp. 1584–1588). Pacific Grove, CA.

Sheu, M.-H., Wang, J.-F., Chen, J.-S., Suen, A.-N., Jeang, Y.-L., & Lee, J.-Y. (1992). A data-reuse architecture for gray-scale morphologic operations. *IEEE Transactions on Circuits and Systems—II: Analog and Digital Signal Processing*, *39*(10), 753–756.

Treichler, J. R., Johnson, Jr., C. R., & Larimore, M. G. (1987). *Theory and design of adaptive filters*. New York: Wiley.

Williams, R., & Zipser, D. (1989). A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, *1*, 270–280.

Yuille, A. L., & Kosowsky, J. J. (1993). Statistical physics algorithms that converge. In R. J. Mammone (Ed.), *Artificial neural networks for speech and vision* (pp. 19–35). London: Chapman & Hall.

Zurada, J. M., & Shen, W. (1990). Sufficient condition for convergence of a relaxation algorithm in actual single-layer neural networks. *IEEE Transactions on Neural Networks*, *1*(4), 300–303.

Received May 9, 2000; accepted April 18, 2002.