Sufficient Conditions for Error Backflow Convergence in Dynamical Recurrent Neural Networks

Alex Aussem
alex@isima.fr
LIMOS (FRE CNRS 2239), University Blaise Pascal, Clermont Ferrand II, 63173 Aubiere Cedex, France

This article extends previous analysis of the gradient decay to a class of discrete-time fully recurrent networks, called dynamical recurrent neural networks, obtained by modeling synapses as finite impulse response (FIR) filters instead of multiplicative scalars. Using elementary matrix manipulations, we provide an upper bound on the norm of the weight matrix, ensuring that the gradient vector, when propagated in a reverse manner in time through the error-propagation network, decays exponentially to zero. This bound applies to all recurrent FIR architecture proposals, as well as fixed-point recurrent networks, regardless of delay and connectivity. In addition, we show that the computational overhead of the learning algorithm can be reduced drastically by taking advantage of the exponential decay of the gradient.

1 Introduction

This article extends the analysis of the gradient error back flow and the forgetting behavior (Bengio, Simard, & Frasconi, 1994; Aussem, Murtagh, & Sarazin, 1995; Aussem, 1999, 2000; Hochreiter & Schmidhuber, 1997; Lin, Horne, Tino, & Giles, 1996) to a fairly general class of fully recurrent architectures obtained by modeling the recurrent synapses as finite impulse response filters (FIR) (Wan, 1993; Aussem et al., 1995). The resulting FIR-recurrent structure encompasses a large variety of synaptic delay-based forward, locally, and globally recurrent neural architecture proposals that have emerged in the literature (see, for instance, Piche, 1994; Tsoi & Back, 1994; Baldi, 1995; Campolucci, Uncini, Piazza, & Rao, 1999; Duro & Santos Reyes, 1999) as well as fixed-point recurrent networks (Pineda, 1987). By virtue of the recurrency, this versatile model is formally defined as a dynamical system described by a system of nonlinear difference equations, whose internal variables are the hidden unit activations. Therefore, these state-space models are also referred to as dynamical recurrent neural networks (DRNN) (Aussem et al., 1995).

The architecture being defined in section 2, the temporal recurrent back-prop algorithm, is recalled in section 3. The error backflow is defined as the
error gradient, with respect to past neuron activations, that flows backward in time through the unfolded error-propagation network in order to adjust the weights. Careful algebraic manipulations are then conducted in section 4 with a view toward formally identifying explicit and interpretation-friendly sufficient conditions for the convergence of the error backflow for all neural architectures that are special cases of the DRNN framework, regardless of whether they necessitate a relaxation phase to settle down to a fixed equilibrium point—the existence of which is out of the scope of this article. An upper bound for the backpropagation through time (BPTT) (Rumelhart, Hinton, & Williams, 1986) truncation length that guarantees a given precision on the weight updates is also derived. Extensive simulations are then conducted to validate these results. A practical measure of the tightness of the bound, called the truncation length error rate, is introduced to gauge the tightness of the upper bound for the BPTT truncation length. The latter is easily interpreted: a truncation length error rate of 2 means that the error backflow was propagated twice as far as necessary into the unfolded error network for the weights to be adjusted with some given precision. It appears experimentally that the truncation length is acceptable as long as the error gradient decay is small.

2 The DRNN Model

DRNN are an extension of conventional recurrent neural networks (RNN) operating in discrete time, allowing the use of arbitrary synaptic time delays. A DRNN with synaptic delays of 0 reduces to the conventional RNN; the information flows through the synapse instantly, while in DRNN the information coming from neuron \( i \) will arrive at neuron \( j \) after a certain delay associated with that particular synaptic link. In other words, the multiplicative interactions of scalars are substituted by temporal convolution operations, necessitating a time-dependent weight matrix (Wan, 1993; Aussem et al., 1995). Assuming the existence of stable fixed points at each iteration \( k \), a general formulation is a coupled set of discrete equations in vector form,

\[
\begin{align*}
\mathbf{v}_k &= g(\mathbf{u}_k) + \mathbf{i}_k \\
\mathbf{u}_k &= \sum_{d=1}^{D} \mathbf{W}_k^T \mathbf{v}_{k-d} + \mathbf{W}_0^T \mathbf{v}_k,
\end{align*}
\]

(2.1)

where \( N \) is the number of units and \( \mathbf{v}_k \) and \( \mathbf{u}_k \) are the \( N \)-dimensional vector activity and internal inputs of the units at iteration \( k \). \( \mathbf{i}_k \) denotes the external input vector of dimension \( N \).

The \( N \times N \) matrix \( \mathbf{W}_k^d \) contains all adaptive weights of delay \( d \leq D \) at iteration \( k \) (see Table 1 for notations). For notational convenience, we do not distinguish among input, hidden, output, and bias units in the following formulas. Consistent with our notations, we note \( g(\mathbf{u}_k) = [g(u_{k1}^1), \ldots, g(u_{kN}^1)]^T \).
under the mild assumption that the nonlinear activation function, \( g() \), is \( C^1 \) with bounded derivatives.

### 2.1 Existence of Fixed Points.

In the formulation above, nondelayed recurrent links were allowed on purpose so that fixed-point networks fall within the DRNN class. Indeed, we show later in this article that the conditions for the error backflow convergence in the DRNN are also necessary for the asymptotic stability of the fixed points in so-called fixed-point networks. This was essentially the only reason for adding relaxation. However, there are a priori no advantages, in terms of processing capabilities, of adding relaxation. Relaxation is time-consuming, and the network is not even guaranteed to settle down to a stable equilibrium solution. To get rid of the relaxation, just assume that \( W_0 \) is triangular.

Due to the nondelayed recurrent links, it is not guaranteed that equation 2.1 always settles down to a stable equilibrium solution. For the system to settle down to a steady state ultimately for any choice of initial conditions, it should be globally asymptotically stable (Khalil, 1996). Networks for which a Lyapunov function can be exhibited, such as the Hopfield model or pure feedforward networks, are guaranteed to be globally asymptotically stable. Unfortunately, little is known about the stability of equation 2.1 for arbitrary connectivity. The only statement that can be made in this regard is that the fixed points—if any—are asymptotically stable provided that all eigenvalues of the linearized system satisfy \( \forall i, \Re \lambda_i < 0 \) (Khalil, 1996), which is assured if \( \| W_0 \| < 1/\mu \) where \( \mu = \max(g'(h)) \). Interestingly, this condition is fulfilled provided the conditions for the error backflow convergence are also fulfilled, as we will see.

Note, however, that the focus of this article is not the existence of stable fixed points, which is assumed. Instead, we study the behavior of the
gradient propagation in this rather general architecture in order to calculate upper bounds on the weight matrix that apply to trajectory recurrent networks (by simply letting $W^0_k$ be the zero matrix) as well as fixed-point recurrent networks, regardless of delay and connectivity.

2.2 Related Neural Models. Many discrete-time neural models discussed in the literature are special cases of DRNN. For instance, if $W^0_k = 0$ and $W^T_k$ are upper triangular for all $d \neq 0$, the model collapses to Wan’s FIR feedforward network (Wan, 1993). Several locally recurrent globally feedforward networks fall into the DRNN class (e.g., the generalized Frasconi-Gori-Soda architecture of Frasconi, Gori, & Soda, 1992; the Poddar-Unnikrishnan architecture of Tsoi & Back, 1994; and the architectures studied by Duro & Santos Reyes, 1999, and Cohen, Saad, & Maromet, 1997). Fixed-point (asymmetric) recurrent networks studied by Pineda (1987) are also encompassed in this framework by setting $W^d_k = 0$ for $d > 0$.

Continuing with examples, if $\forall d \neq 1, W^d_k = 0$, the model reduces to a standard globally recurrent network (Williams & Zipser, 1989; Lin et al., 1996). Nonlinear autoregressive (moving average) with eXogeneous inputs (NARX and NARMAX) models (Lin et al., 1996) are also included in this representation.

We stress that even if infinite impulse response (IIR) synapse architectures (e.g., Tsoi and Back IIR; Tsoi & Back, 1994) do not fall within the DRNN class, IIR networks may be seen as equivalent to expanded DRNN with extra linear activation units and extra local FIR feedback. Therefore, the training scheme and the upper bounds developed later will also hold provided the IIR network is transformed into a DRNN network. All of the architectures mentioned above suffer from exponential forgetting behavior, as shown in the next section.

3 Error-Derivative Calculation

The introduction of time delays leads to a somewhat more complicated training procedure because the current state of a unit may depend on several events having occurred in the neighboring neurons at different times. (See Table 2 for the calculations.) Classically, the goal is to find a procedure to adjust the weight components $w^d_k$ so that a given fixed initial condition $v_0$ and a set of input vectors $\{i_k\}$ taken from the training set result in a set of fixed points (the existence of which is assumed), whose components along the output units have a desired set of values $\{d_k\}$. This will be accomplished by our minimizing a quadratic function $E = \frac{1}{2} \sum_{k=0}^{k_f} e_k^T e_k$, which measures the errors $e_k = d_k - v_k$ between the desired and the actual fixed-point values along an epoch (each pass through the training set is called an epoch) starting at iteration 0 up to $k_f$. In an epochwise adaptation, a new weight is
Table 2: Notations for Error Derivative Calculations.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(d_k)</td>
<td>Target vector at iteration (k)</td>
</tr>
<tr>
<td>(e_k)</td>
<td>Error vector at iteration (k)</td>
</tr>
<tr>
<td>(E_k)</td>
<td>(Scalar) cost at iteration (k)</td>
</tr>
<tr>
<td>(y_k)</td>
<td>So-called error backflow vector at iteration (k)</td>
</tr>
<tr>
<td>(\delta_k)</td>
<td>Delta vector at iteration (k)</td>
</tr>
<tr>
<td>(\mu)</td>
<td>Defined as (\max_h (g'(h)))</td>
</tr>
</tbody>
</table>

generated at the end of the epoch by \(\Delta w = -\eta \frac{\partial E}{\partial w}\), where \(\eta\) is the learning rate. Without loss of generality, we adopt the ordinary gradient-descent method and assume \(\eta\) is constant over an epoch.

The primary goal is to find an expression for the term \(\frac{\partial E}{\partial w}\). In order to clarify our notation, let \(G_k\) and \(G'_k\) denote the \(N \times N\) diagonal matrices constructed from \(g(u_k(j))\) and \(g'(u_k(j))\), respectively, for \(j = 1, \ldots, N\). Note that for sigmoid functions, \(G'_k = \beta G_k (1 - G_k)\) (recall that \(g'(h) = \beta g(h)(1 - g(h))\)).

The following generalized \(\frac{\partial}{\partial \theta}\) rule for DRNN networks provides a backward-difference equation in order to compute recursively the error gradient with respect to the weights.

**Theorem 1.** If the DRNN model is convergent over the epoch \(0 \leq k \leq k_f\), then the gradient of the cost \(E = \frac{1}{2} \sum_{k=0}^{k_f} e_k^T e_k\) can be computed at time step \(k_f\) by

\[
\frac{\partial E}{\partial \theta} = \sum_{k=d}^{k_f} \delta_k \mathbf{v}_{k-d}^T, \quad \forall d \leq D,
\]

where the sequence of vectors \(\delta_k\) for \(k = d, \ldots, k_f\), is given by \(\delta_k = G'_k y_k\) and the sequence of vectors \(y_k\) is the solution of the following backward difference equation:

\[
y_k = [I - W_k^d G'_k]^{-1} \left[ e_k + \sum_{j=1}^{D} W_{k+j}^j G'_{k+j} y_{k+j} \right]
\]

with the boundary conditions: \(y_{k_d} = [I - G'_{k_d} W_{k_d}^0]^{-1} e_{k_d}\) and \(y_j = 0\) for \(j > k_f - D\).

The demonstration of this proposition has been given elsewhere (Aussem et al., 1995). This generalized \(\frac{\partial}{\partial \theta}\) rule puts recurrent backprop (Pineda, 1987),
temporal backprop (Wan, 1993), and backprop-through-time (BPTT) (Werbos, 1974; Rumelhart et al., 1986) into a single framework. Like BPTT in its general form, the procedure suggested here makes necessary the artifact of unfolding the error-propagation network (see Figure 1) in a reverse way in time (Rumelhart et al., 1986). Typically, all the individual weight adjustment resulting from the duplication of the connections in the unfolded error-propagation network must be recombined to compute the overall change. This is achieved by summing the individual gradient components over the overall unfolding interval.

4 Conditions for Backprop Convergence

Classically, the BPTT-like procedure proposed in theorem 1 requires the duplication of the connections in the unfolded error-propagation network and the propagation of the vectors, $y_k$, backward in time over $k_f, k_f - 1, \ldots, 1$ to compute equation 3.1. However, because the storage and memory requirements grow proportionally with $k_f$, the procedure becomes swiftly inhibitory from an implementation standpoint when long epochs are to be considered. This is also the major flaw of BPTT. Fortunately, we will show here that the cost due to the duplication of the hardware can be maintained within acceptable limits while computing the error gradient as precisely as desired.
Define $y_{k}^{k+n}$ as the solution of equation 3.2 at time $k$ when all $e_{k+j}$ are set to 0 except for $j = n$. Let $\delta_{k}^{k+n} = G_{k}y_{k}^{k+n}$. We can now expand the delta rule, equation 3.1, by incorporating the terms $y_{k}^{k+n}$ and $\delta_{k}^{k+n}$:

$$\Delta W^{d} = \eta \sum_{k=d}^{k_{f}} \delta_{k}v_{k-d}^{T} = \eta \sum_{n=d}^{n} \delta_{n}^{d}v_{n-d}^{T}. \quad (4.1)$$

In batch mode, the question we are faced with is whether the sum is diverging or remains bounded as $k_{f} \to \infty$. Answering this question in the general framework is a formidable task since the weight evolution is strongly dependent on the data set itself. In practice, the weights may blow up if no regularizer is used, only to be limited by their saturation limits. In fact, it seems that no general answers can be formulated.

In on-line mode or pattern mode, the weights are adjusted after each pattern presentation. At iteration $n$, the weights are adjusted by summing the individual components $\delta_{k}^{n}v_{k-d}^{T}$ where the index $k$ runs over all the prior time lags in the unfolded network, that is, $\sum_{k=d}^{n} \delta_{k}^{n}v_{k-d}^{T}$. Therefore, the question we are faced with is whether this sum is diverging or remains bounded as $n \to \infty$. Clearly, for the sum to converge, not only shall $\delta_{n-j}^{n}$ tend to zero with respect to $j$, but the decay rate must be large enough. More explicitly,

**Definition 1.** The gradient-based training procedure is said to be convergent at each intermediate time step on the epoch if $\lim_{n \to \infty} \sum_{k=d}^{n} \delta_{k}^{n}v_{k-d}^{T}$ exists.

We therefore restrict our attention to the so-called error backflow, which is formally defined below:

**Definition 2.** The error backflow, due to error $E_{n}$, is the sequence of vectors, $\delta_{k}^{n}$ for $k = n, n-1, \ldots, 1$, used in equation 4.1 to adjust the weights.

Given an error $E_{n}$ at time $n$, the error backflow for that error can intuitively be viewed as the sequence of vectors that carry all the information necessary to adjust the past weights due to that error.

With the aim of establishing sufficient conditions for the sum to converge, we now state the fundamental result of this article:

**Theorem 2.** Let the instantaneous scalar cost be $E_{n} = \frac{1}{2}e_{n}^{T}e_{n}$ and consider the error vector $y_{n-k}^{T} = \frac{\partial E_{n}}{\partial W_{n-k}}[I - G_{n-k}^{T}W_{n-k}^{0}]^{-1}$. Let $k_{f} > 0$ denote the epoch length.
of the system and \( \mu = \max(g'(h)) \). Let

\[
a_d = \max_k \left( \frac{\mu \| W^d \|}{1 - \mu \| W^0 \|} \right), \quad 1 \leq d \leq D.
\]

(4.2)

If \( \rho = \sum_{d=1}^{D} a_d < 1 \), then \( \| y_{kD+i} - (kD+i) \| < \rho^{k+1} \| y_k \|, \forall k \geq 0, \forall j = 1, \ldots, D \).

**Proof.** This technical result is deferred to the appendix.

The theorem provides a sufficient condition for the convergence of the training procedure. It is worth remarking that the exponential decay of \( y_{kD+i} \) is equivalent to the exponential decay of the error backflow \( \delta_{kD+i} \) (recall that \( \delta_k^{k+n} = G_k^{k+n} y_k \) and \( G_k^{-1} \) always exists) and is also equivalent to the exponential decay of the gradient with respect to the past unit activation, \( \partial E_k / \partial v_{kD+i} \).

The theorem says, in essence, that the terms \( y_{kD+i} \) (as well as \( \delta_{kD+i} \) and \( \partial E_k / \partial v_{kD+i} \)) are bounded above by a series decaying at exponential rate, \( \rho \), with respect to \( n \) provided \( \rho < 1 \), hence the definition:

**Definition 3.** \( \rho = \sum_{d=1}^{D} \max_k \left( \frac{\mu \| W^d \|}{1 - \mu \| W^0 \|} \right) \) will be called the error backflow exponential decay rate or, more simply, the gradient decay rate.

The analytical error backflow exponential decay rate, \( \rho \), bounds above the true error backflow decay rate. In batch mode (i.e., when the weights are adjusted after each pass through the training set), \( \rho < 1 \) translates into very appealing conditions:

**Corollary 1.** In batch mode, \( \sum_{d=1}^{D} \| W^d \| < 1/\mu \) is a sufficient condition for the convergence of the training procedure.

The above inequality set a penalty on the large weights, regardless of delay and connectivity, to ensure the error backflow decay. Typically, when the network admits fixed points at each time step, the residual backpropagated errors drop off exponentially through the past time slices, rapidly becoming negligible—hence, the forgetting behavior of (standard) recurrent neural networks. The gradient decay rate, \( \rho \), can then be written as

\[
\rho = \sum_{d=1}^{D} \frac{\mu \| W^d \|}{1 - \mu \| W^0 \|}.
\]

(4.3)
More specifically, with compatible norms $\|A\|_1 = \max \sum_j |w_{ij}|$, $\|A\|_\infty = \max i \sum |w_{ij}|$, and $\|A\|_F = [\sum_{i,j} |w_{ij}|^2]^{1/2}$, we obtain three easily interpreted sufficient conditions to ensure training convergence:

$$
\begin{align*}
\mu &\sum_{d=0}^D \max \left( \sum_j |w_{dj}| \right) < 1 \\
\mu &\sum_{d=0}^D \max \left( \sum_i |w_{id}| \right) < 1 \\
\mu &\sum_{d=0}^D \left[ \sum_{i,j} |w_{ij}|^2 \right]^{1/2} < 1.
\end{align*}
$$

These conditions are satisfied, for instance, if all weights are in the range $\pm \frac{1}{\mu N(D+1)}$. Note that neither condition implies the others.

4.1 Related Results. The upper bound is in nice agreement with the results of Hochreiter and Schmidhuber (1997) for standard recurrent networks (Williams & Zipser, 1989). For example, if $N$ is the matrix size, they find that setting $\forall i,j, |w_{ij}| < w_{\text{max}} < 4/N$ and $\beta = 1$ will result in the exponential gradient decay with a decay rate $\tau = Nw_{\text{max}}/4 < 1$. Clearly, setting $W_0^0 = 0$ and $D = 1$ transforms the DRNN into a standard recurrent network, with a single delay fixed to 1. Notice that for sigmoid transfer functions, $G_k^\prime$ is by definition an $N \times N$ diagonal matrix constructed from $g^\prime(h) = \beta g(h)(1 - g(h))$ so that $g^\prime(h) < \beta/4$. Hence, $\mu = 1/4$. Consequently, $\rho < 1$ reduces to a somewhat weaker condition $\|W_k^0\| < 4$ for all $k (\beta = 1)$, since one or several weights may exceed $w_{\text{max}}$. Conversely, it is readily seen that $\|W_k^0\| < 4$ is satisfied if $|w_{ij}| < 4/N$ since $\|W\| < Nw_{\text{max}}$. Similarly, Frasconi et al.’s (1992) forgetting behavior condition $-|w_{jj}| < 1/d$ where $d = \max_i (g^\prime(h)) = 4$, for local feedback networks—is also a particular case of theorem 2.

Interestingly, the existence of $[I - G_k^0 W_0^0]^{-1}$, and thereby the stability of the fixed points, is also guaranteed. It suffices to set $D = 0$; theorem 2 yields $\|W_0^0\| \cdot \|G_k^0\| < 1$, which clearly is a sufficient condition for the existence of $[I - G_k^0 W_0^0]^{-1}$.

4.2 Link to Regularization. Interestingly, the term $\sum_{d=0}^D \|W_0^d\|$ in corollary 1 reminds us of a specific regularizing term usually added to the cost function to penalize nonsmooth functions. The links between regularization and Bayesian inference are well established. It can easily be shown that in the gaussian case, minimizing the regularized cost is equivalent to maximizing
the posterior solution, assuming the following prior on the weights,

\[ P(w | \alpha) \propto \prod_{d=0}^{D} \exp(-\alpha \|W^d\|), \tag{4.5} \]

where the hyperparameter, \(\alpha\), parameterizes the distribution. Therefore, the training convergence and the statistical analysis of recurrent neural networks concur—by very different considerations—in the observation that the small weights should be privileged. Formal regularization stabilizes the training procedure by indirectly enforcing the error backflow decay whatever prior is used (e.g., gaussian prior, Laplace prior; Goutte, 1997).

5 Tailoring the Number of Backpropagations

In this section, we derive an upper bound for the BPTT truncation length that guarantees a given precision on the weight updates in order to reduce the cost due to the duplication of the hardware. As \(D\) is the maximum synaptic delay, we assume that the error propagation network, at iteration \(n\), is unfolded \(sD\) times, where \(s\) is some positive integer. Therefore, \(\hat{\Delta}W^d_n(s) = \eta \sum_{k=n-sD}^n \delta_{k}^n v_k^T \) is the estimate of the true weight matrix update \(\Delta W^d_n(s) = \eta \sum_{k=sD}^n \delta_{k}^n v_k^T\) at time \(n > sD + d\). We are seeking a condition on the network parameters such that \(\|\Delta W^d_n(s) - \Delta W^d_n\| < \epsilon\), for all \(\epsilon > 0\). The following theorem expresses \(s\) as a function of \(\epsilon\), the required precision on the weight updates:

**Theorem 3.** In batch mode, it suffices, at iteration \(n\), to allocate memory to accommodate a maximum of \(sD\) backpropagations where \(s\) is the smallest integer satisfying

\[ s > \frac{1}{\ln \rho} \ln \left[ \frac{\epsilon(1 - \rho)(1 - \mu \|W^0\|)}{\eta \mu ND \|e_n\|} \right] \tag{5.1} \]

to guarantee that the weights are updated with precision \(\epsilon\).

**Proof.** Let \(n\) be the current iteration. Let \(\hat{\Delta}W^d_n(s) = \eta \sum_{k=n-sD}^n \delta_{k}^n v_k^T\) for \(n > sD + d\), denote the estimate of the weight matrix update at time \(n\) obtained after \(sD\) duplications of the network, so that \(\Delta W^d_{sD+d}(s) = \Delta W^d_{sD+d}\).

According to equation 4.1, we have \(\forall d \leq D, \forall s > 0\),

\[ \|\Delta W^d_n(s) - \Delta W^d_n\| = \eta \sum_{k=sD}^{n-sD-1} \delta_{k}^n v_k^T. \]
For \( n > sD + d \), we may write

\[
\sum_{k=d}^{n-sD-1} \delta_k^T \nabla_{k-d} \leq \sum_{k=d}^{n-sD-1} \|G_k^T\| \|y_k^T\| \|v_{k-d}\|
\]

\[
\leq \mu \sum_{k=d}^{n-sD-1} \|y_k^T\| \|v_{k-d}\|
\]

\[
\leq \mu N \sum_{k=d}^{n-sD-1} \|y_k^T\|
\]

\[
\leq \mu ND \|y_n^T\| \frac{\rho^{s+1}}{1 - \rho}, \tag{5.2}
\]

where the inequalities stem from theorem 1 and from \( \|v_{k-d}\|_p < N \) for \( p = 1, 2, \infty \). Theorem 3 is readily obtained from the above inequality and from the inequality

\[
\|y_n^T\| \leq \|I - W_0^T G_k^T\|^{-1} \|e_n\| \leq \frac{\|e_n\|}{1 - \mu \|W_0^T\|}, \tag{5.3}
\]

completing the proof.

Note furthermore that \( \|W_0^T\| < Nw_{\max} \); the dependence on \( n \) (i.e., the instant at which the gradient is backpropagated) can be removed by considering that

\[
\|e_n\| \leq N_{out} \cdot \max_j (e_n(j)) \leq N_{out}, \tag{5.4}
\]

where \( N_{out} \) is the number of output units.

Formula 5.1 provides an implementation-friendly stopping criterion. As far as the memory allocation is concerned, the upper bound for \( s \) ensures that the bookkeeping always fits the time interval spanned by the error backflow.

**Remark.** It is worth remarking that \( s \) varies as a function of \( \ln(1 - \rho) / \ln \rho \) (see Figure 2) and thus diverges swiftly for values of \( \rho > 0.9, s \to \infty \) when \( \rho \to 1 \) and \( s \to 1 \) when \( \rho \to 0 \). Note also that the number of backpropagations to guarantee a given precision on the weights updates varies as a function of \( \ln ND \), provided \( \rho \) and \( \|W_0^T\| \) are constant.

**5.1 Numerical Applications.** For purposes of illustration, numerical applications are given for standard recurrent neural networks that are special cases of DRNN.
Figure 2: Typical increase of the number of backpropagations in time (i.e., $sD$), given by corollary 2, to guarantee a given precision error on the weight updates, as a function of $\rho$. See equation 5.1.

5.1.1 Williams and Zipser Architecture. Consider first the recurrent network studied by Williams and Zipser (1989) obtained by setting $W^0 = 0$ and $D = 1$ (all synapses are delayed by one unit of time). Usual values such as $N = 10$, $D = 1$, $\mu = 1/4$, $\eta = 10^{-2}$, $\epsilon = 10^{-6}$, and $\rho = 10^{-1}$, with a single output, yield a sensible value for $s$ equal to 5 by formula 5.1. In other words, the absolute error on the weight update at each time step during the trajectory is guaranteed to remain below $10^{-6}$ (i.e., $\|\Delta W^d_n(s) - \Delta W^d_n\| < 10^{-6}$) after only five replications of the network backward in time. If we increase $\rho$’s value to 0.5, we obtain $s = 17$, and for $\rho = 0.9$, we obtain $s = 128$.

5.1.2 NARMAX Networks. These feedback networks consist of a feed-forward (static) network with a single output $z(t)$ and $M$ state additional inputs that are delayed values of the output, $z(t-1), \ldots, z(t-D)$. NARMAX networks can be seen as particular DRNN where delayed synapses ($d = 1, \ldots, D$) connect the output unit to the hidden units. Values such as $N = 10$, $D = 5$, $\mu = 1/4$, $\eta = 10^{-2}$, $\epsilon = 10^{-6}$, $\|W^0\| = 1$, and $\rho = 10^{-1}$ yield a value for $s$ equal to 6. For $\rho = 0.5$ and $\rho = 0.9$, we obtain $s = 19$ and $s = 144$, respectively.
5.1.3 DRNN. Consider now a fully recurrent DRNN of size $N = 100, D = 1$ with 10 outputs ($N_{out} = 10$) with a (higher) precision of $\epsilon = 10^{-8}$ and $\mu = 1/4, \eta = 10^{-2}, \|W_0\| = 1$. For $\rho = 10^{-1}$, we obtain $s = 10$. For $\rho = 0.5$ and $\rho = 0.9$, we obtain $s = 32$ and $s = 233$, respectively.

As expected, an increase of $\rho$ entails a dramatic increase of $s$ in all cases, making corollary 2 useless in some cases. Therefore, the validity and the usefulness of the error backflow exponential decay rate, $\rho$, will be experimentally confirmed in section 6.

5.2 Complexity. In the light of the discussion above, one may naturally take advantage of the exponential decay of the gradient to stop the gradient backpropagation after $sD$ relaxations of the error propagation network. This leads to an efficient parallel implementation requiring reasonable computational resources and obeying predefined precision requirements on the weights.

Provided $\rho$ is constant, $s$ is of order $\ln(DN)$. Storage of order $N \ln(DN)$ is needed to keep track of the network previous states and $DN^2$ to store the filter parameters. The time required at each backward pass to compute the elementary weight update scales as $O(DN^2)$ so that $O(DN^2 \ln(DN))$ operations are needed at each time step to adjust the network parameters. In contrast, a forward propagation algorithm entails storage capacity of $O(DN^3)$ and the computational requirements scale\(^1\) as $O(D^2N^4)$. Finally, the latter requires a matrix inversion to compute the gradient, with the result that locality is lost. It is essential to note, however, that this procedure essentially gives up on the the long time lag problem by optimally cutting off the gradient, thus reducing the computational effort. Therefore, this training procedure is sure to fail if the time gap between an event and the occurrence of the corresponding error signal spans a length of time significantly greater than the maximum delay order present in the DRNN structure. In this case, one has to resort to more sophisticated architectures or data processing techniques, as discussed in section 7.

6 Experimental Simulation

The analytical findings we arrived at provide valuable theoretical insight into the error backflow behavior and the forgetting behavior of recurrent networks with delays. As expected, extensive simulations carried out in Aussem (1999) with a family of DRNN models clearly confirm that the

\(^1\) Note that BPTT and RTRL can be merged together to reduce the average time complexity per time step of the forward propagation algorithm to $O(D^2N^3)$ instead of $O(D^2N^4)$ while still calculating the same gradient (Schmidhuber, 1992).
Figure 3: Truncation length error rate definition. Let $Y(t - k) = \|y^n_{t-k}\|, k_1 = s_1 D$ is the smallest number of backpropagations in time such that $Y(t - k_1) < \epsilon$, where $\epsilon$ is some small value. $k_2 = s_2 D$, where $s_2$ is the smallest integer satisfying inequality 5.1. A truncation length error rate of 2 means that the error backflow was propagated twice as far as necessary into the unfolded error network.

6.1 Experimental Set-up. For simplicity, a simple $1 \times N \times 1$ DRNN model was used where $N$, the number of hidden units, was varied. The single
input is connected to all hidden units with nondelayed links \((D = 0)\). The \(N\) hidden units are fully connected with connections of delay \(d = 1, \ldots, D\), where \(D\), the filter order, was varied. All hidden units were connected to the single output with nondelayed links. As \(W^0\) is triangular, a fixed point exists at each iteration. Such architecture forces the DRNN model to encode the input sequence internally as no spatial memory (i.e., time-lagged window) is provided.

Initial weight values of the networks were chosen randomly between \(-\xi\) and \(+\xi\), with \(\xi = 10\epsilon - 2\) such that \(\sum_{d=0}^{D} \|W^d\| < 1/\mu\) to ensure convergence of the training (see corollary 1). We have chosen the Frobenius matrix norm, which is compatible with the Euclidean vector norm. One hundred random input-target pairs were generated. Each time, the DRNN output was calculated for that particular input. Given the target, the error gradient was backpropagated into the unfolded network until it was found neglectable \((<\epsilon)\) and the observed decay rate was estimated.

In practice, the larger the weights, the larger the decay rate. Also, in order to cover all possible decay rates, the procedure was repeated with, at each time, a value for \(+\xi\) incremented by \(10\epsilon - 2\). Therefore, the decay rate averaged over 100 trials was observed to be larger each time. The procedure was stopped once the observed exponential rate became almost equal to 1.

### 6.2 Results

Several experiments were conducted with \(N = 1, 5,\) and 10 and \(D = 1, 5,\) and 10 (see Figure 4). The error rates are shown to be very similar to the theoretical truncation length plotted in Figure 2. It appears implicitly that the analytical convergence rate, \(\rho\), is an accurate approximation of the true observed rate as long as \(\rho\) is small. The situation, however, degrades as the size of the network increases. As expected, the error rate swiftly increases in all cases as \(\rho\) approaches 1. Still, acceptable errors are obtained for values of \(\rho\) below 0.5: the error rate on the truncation length is less than 5.

For completeness, we have plotted in Figure 5 the truncation length error rate for (1) a typical Williams and Zipser architecture with \(N = 10\) fully connected units, a single input, and a single output, and (2) a typical (feedforward) NARMAX network with \(N = 10\) hidden units, a single exogenous input, a single output \(z(t)\), and \(D = 5\) additional inputs that are delayed values of the output, \(z(t - 1), \ldots, z(t - 5)\). Here again, the error rate swiftly increases as \(\rho\) approaches 1. It seems that as the product \(ND\) is increased, the error rate tends to increase more rapidly.

### 7 Discussion

So far, we have focused on the analysis of the vanishing gradient. We now briefly point out a few solutions proposed in the literature to alleviate this problem.
Figure 4: Truncation length error rate versus the observed decay rate plotted for different DRNN architectures. $N$ is the number of hidden neurons, and $D$ is the maximum synaptic delay. The hidden layer is fully connected to itself with connections of delay $d = 1, \ldots, D$. 
Williams & Zipser architecture
N=10

NARMAX
N=10,D=5

Figure 5: Truncation length error rate versus the observed decay rate. (Top) A fully connected Williams and Zipser architecture with 10 units. (Bottom) A NARMAX (feedforward) architecture with 10 hidden units and 5 additional inputs that are delayed values of the output.

DRNN-like networks fail to learn anything within reasonable time in the presence of long time lags between relevant inputs and target events because the gradient vanishes at an exponential rate. This is not new; various experiments that validate claims regarding the limitations in learning long-term dependencies can be found—for instance, in Aussem et al. (1995) and Hochreiter and Schmidhuber (1997)—and a further understanding of this problem is developed in Bengio et al. (1994) and Lin et al. (1996).

While large time delays also lead to vanishing gradient in the case of long-term dependencies, the effect gets weakened by a factor of $1/D$. This property is exploited in networks with a single adaptable time delay associated with each link, as discussed in Duro and Santos Reyes (1999),
which extends the traditional backprop algorithm to this purpose. Constructive heuristic for carefully adding new delayed connections to a given architecture are proposed in Boné, Cruciani, Verley, & Asselin de Beauville (2000).

Another possibility is to decompose the input sequence into varying scales of temporal resolution via a wavelet transform, each of which holds information on the specific timescale, so that faint temporal structures may be detectable at different resolution levels (see, e.g., Aussem & Murtagh, 1997). A specific architecture that operates on multiple scales was also proposed by El Hihi and Bengio (1996). Nevertheless, these approaches generally fail if precise real values are to be stored precisely over periods potentially spanning hundreds of time steps.

To remedy this difficulty, Hochreiter and Schmidhuber (1997) and Gers, Schmidhuber, and Cummins (2000) have proposed a promising architecture, called long short-term memory (LSTM), based on high-order gating units capable of bridging arbitrarily long time intervals. This desirable feature arises from the inclusion of specific so-called memory cells, aimed at maintaining constant error flows within the cell. In contrast to DRNN-like networks, error signals trapped within a memory cell do not change until they are released by the input gate. This is achieved by using input and output multiplicative gate units in order to open and close access to error flow through internal states of memory cells. Besides, a local self-recurrent connection with a nontrainable weight of 1.0 ensures that the error flow remains constant within the cell. In other words, this can be seen as a way to enforce $\rho = 1$ and stay at the cutting edge between exponential explosion and exponential decay. As a result, LSTM can extract and store information over potentially unbounded periods.

On the other hand, the appealing storing capability may come at the expense of a loss of sensitivity to short-term information since the additional multiplicative gate units—and thus the additional connections—exclusively devoted to maintaining constant gradient flow become unnecessary. Yet this is also true for DRNN-like networks, when the synaptic delay orders overshoot the typical time length spanned by the useful information. In either case, the use of sophisticated units and/or extra delayed connections have a side effect arising from having to use relatively large nets, which can undermine the success of the model.

There is still much room for work to be performed to overcome the problem of vanishing gradient. While several promising approaches have been proposed to overcome this limitation, whether by enforcing constant error flow within the sophisticated units, using adaptable delays, or decomposing the input sequence into varying temporal scales, no method is a priori superior to another in all situations. As many authors have pointed out, it is essential to experiment with a variety of techniques to determine which is best suited to the specific task at hand.
8 Conclusion

In this article, we have derived an upper bound on the norm of the weight matrix to ensure the convergence of the error backflow in a fully recurrent architecture that encompasses a large variety of synaptic delay–based discrete-time neural models that have been proposed in the literature. An upper bound for the BPTT truncation length that guarantees a given precision on the weight updates was also derived. We have experimentally shown that the truncation length is acceptable as long as the error gradient decay is small.

Appendix: Asymptotic Behavior of the Backpropagated Error Terms

In this appendix, theorem 1 is demonstrated.

The behavior of $y_{k_f}^{k_f}$ can be inferred from the backward recursive equations 3.2. It suffices to set $e_j = 0$ for $j = 0, 1, \ldots, k_f - 1$. This yields the relation

$$y_{k_f}^{k_f} = [I - W_0^0 G_k']^{-1} \sum_{d=1}^{D} W_d^{k_f} G_{k_f}^d y_{k_f}^d$$

(A.1)

for $k < k_f$, with the following boundary condition: $y_{k_f}^{k_f} = [I - W_0^0 G_k']^{-1} e_{k_f}$. The vectorial equation, A.1, governs the dynamic of the error terms. Let $\mu = \max \{g'(h)\}$ so that $\|G_k'\| < \mu$, $\forall k$. Let us assume that $\|W_0^0\| < 1/\mu$ for $k = 0, \ldots, k_f$ so that the convergence of $I + W_0^0 G_k' + (W_0^0 G_k')^2 + \ldots$ is assured (Notice at this point that $\|\|$ refers to any matrix of vector norm compatible with each other, i.e., $\|A\|_p = \sup_{x \neq 0} \|Ax\|_p/\|x\|_p$, $p = 1, 2, \infty$). Hence, for any iteration $k = 0, \ldots, k_f - 1$, we have

$$\|y_{k_f}^{k_f}\| \leq \|I - W_0^0 G_k'\|^{-1} \sum_{d=1}^{D} \|W_d^{k_f}\| \|G_{k_f}^d\| \|y_{k_f}^d\|$$

$$\leq \frac{\mu}{1 - \mu \|W_0^0\|} \sum_{d=1}^{D} \|W_d^{k_f}\| \|y_{k_f}^d\|.$$  

(A.2)

Now define $a_d = \max_k \left( \frac{\mu \|W_d^0\|}{1 - \mu \|W_0^0\|} \right), 1 \leq d \leq D$. From equations A.1 and A.2, we deduce the following inequality:

$$\|y_{k_f}^{k_f}\| \leq a_1 \|y_{k_f}^{k_f}\| + a_2 \|y_{k_f}^{k_f}\| + \cdots + a_D \|y_{k_f}^{k_f}\|.$$  

(A.3)

The time dependency of the coefficients has been removed. This allows the analysis of a univariate series, $\{u_n\}$, defined by $u_n = \sum_{j=1}^D a_j u_{n-j}$, to be
carried out. \( \{u_n\} \) is of interest since
\[ u_n \geq \|y_{k_f-n}^k\| \text{ for } n = 0, \ldots, k_f, \]
with
\[ u_n = \|y_{k_f-n}^k\| \text{ for } n = 0, \ldots, D - 1. \]
Since
\[ u_n < \max_{j=1,\ldots,D}(u_{n-j}) \cdot \sum_{j=1}^{D} a_j, \]
and considering that
\[ u_j < (\sum_{j=1}^{D} a_j) \cdot u_0 \text{ for } j > 0 \]
and
\[ u_0 = \|y_{k_f}^k\|, \]
it is readily proven by induction that
\[
\|y_{k_f-(D+j+1)}^i\| \leq u_{D+j+1} < \left( \sum_{j=1}^{D} a_j \right)^{k+1} \|y_{k_f}^k\|.
\]
\[ \forall k \geq 0, \forall j = 0, \ldots, D - 1. \quad (A.4) \]

Therefore, \( \sum_{j=1}^{D} a_j < 1 \) ensures the convergence of the backpropagation
dynamic, completing the proof.

References

Aussem, A. (2000). Sufficient conditions for error back flow convergence in
Conference on Neural Networks (pp. 577–582).
Aussem, A., & Murtagh, F. (1997). Combining neural network forecasts on
wavelet-transformed time series. Connection Science, 9, 113–121.
networks—towards environmental time series prediction. International Jour-
nal of Neural Systems, 6, 145–170.
Baldi, P. (1995). Gradient descent learning algorithm overview: A general dy-
namical systems perspective. IEEE Transactions on Neural Networks, 6, 182–
195.
with gradient descent is difficult. IEEE Transactions on Neural Networks, 5,
157–166.
exploration approach to constructive algorithms for recurrent neural net-
works. In Proceedings of the International Joint Conference on Neural Networks
(pp. 27–32).
algorithms for locally recurrent neural networks. IEEE Transactions on Neural
Networks, 10, 253–271.
Duro, R. J., & Santos Reyes, J. (1999). Discrete-time backpropagation for training
synaptic delay-based artificial neural networks. IEEE Transactions on Neural
Networks, 10, 779–789.


Received March 30, 2001; accepted November 20, 2001.