ESANN'2005 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 27-29 April 2005, d-side publi., ISBN 2-930307-05-6.

Stability of backpropagation-decorrelation efficient O(N) recurrent learning

Jochen J. Steil

Neuroinformatics Group, University of Bielefeld, Germany
jsteil@techfak.uni-bielefeld.de, www.jsteil.de

Abstract. We provide a stability analysis based on nonlinear feedback theory for the recently introduced backpropagation-decorrelation (BPDC) recurrent learning algorithm. For one output neuron BPDC adapts only the output weights of a possibly large network and therefore can learn in O(N). We derive a simple sufficient stability inequality which can easily be evaluated and monitored online to assure that the recurrent network remains stable while adapting. As byproduct we show that BPDC is highly competitive on the recently introduced CATS benchmark data [1].

1 Introduction

While recurrent neural networks have matured into a fundamental tool for trajectory learning, time-series prediction, and other time-dependent tasks, major difficulties for their more widespread application remain. These are the known high numerical complexity of training algorithms and the difficulties in assuring stability, which often is crucial in particular for adaptive control applications (see also the review [2]). Most of the efficient existing algorithms rely on backpropagation through time to compute error gradients and additionally require proper adjustment of learning rates and time-constants.

To advance in the direction of a simple online recurrent learning technique, which could attract an even wider audience to use recurrent networks, in [3] we have introduced the *backpropagation-decorrelation* rule (BPDC), which combines three principles: (i) one-step back propagation of errors; (ii) the usage of the temporal memory in the network dynamics which is adapted based on decorrelation of the activations, and (iii) the employment of a non-adaptive reservoir of inner neurons to reduce complexity. The output weights then implement a linear readout function while at the same time the output neuron provides full feedback into the reservoir. In its most efficient and useful form, the BPDC rule applied to learning one output is $O(N)$ and in [3] it has already been shown that BPDC performs well on a number of standard tasks.

The BPDC rule roots in a combination of recent ideas to differentiate the error function with respect to the states in order to obtain a "virtual teacher" target, with respect to which the weight changes are computed [4, 5]. Further, under the notion "echo state network" [6] and "liquid state machine" [7] non-adaptive recurrent networks as a kind of dynamic reservoir to store information about the temporal behavior of inputs have been proposed, which allow to effectively learn a linear readout function. In [3], the BPDC rule has been formally

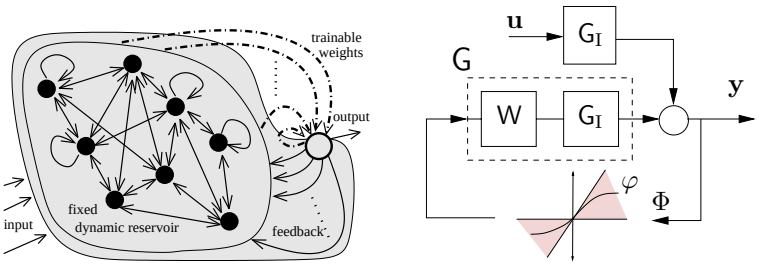

Fig. 1: Left: BPDC trains output weights between a dynamic reservoir and the output neuron, which gives feedback to the reservoir. Right: Network composed from linear feedforward operator G and nonlinear feedback Φ (see text).

derived and shown that it can be interpreted as a combination of these ideas leading to (i)-(iii).

In this contribution, we give a formal technique to analyze the stability of the "fixed dynamic reservoir + output neuron + feedback " network configuration shown in Fig. 1. It is based on the small gain theorem from nonlinear feedback theory and leads to a simple stability inequality to be monitored online while learning. We demonstrate this technique and give simulations for the recently introduced CATS benchmark which shed some light on the complex trade-offs between stability, learning, and network configuration.

2 The BPDC learning rule

We consider fully connected recurrent networks

$$\mathbf{x}(k+\Delta t) = (1-\Delta t)\mathbf{x}(k) + \Delta t \mathsf{W}\varphi(\mathbf{x}(k)) + \Delta t \mathsf{W}_u \mathbf{u}(k), \tag{1}$$

where $x_i, i = 1, \ldots, N$ are the states, $\mathsf{W} \in \mathbb{R}^{N\times N}$ is the weight matrix, W_u the input weight matrix and $k = \hat{k}\Delta t, \hat{k} \in N_+$ is a discretized time variable such that for small Δt we obtain an approximation of the continuous time dynamics $dx/dt = -x + W\varphi(x)$ and for $\Delta t = 1$ the standard discrete dynamics. We assume that φ is a standard sigmoidal differentiable activation function with $\varphi' \leq 1$ and is applied component wise to the vector $\mathbf{x}$. We further assume that W is initialized with small random values in a certain weight initialization interval $[-a, a]$. Denote by $O \subset \{1, .., N\}$ the set of indices s of N_O output neurons (i.e. x_s output $\Rightarrow s \in O$) and let for a single output neuron w.r. $O = \{1\}$ such that x_1 is the respective output of the network shown in Fig. 1.

If all but the output weights are fixed, we can regard the inner neurons as dynamical reservoir which is triggered by the input signal and provides a dynamical memory. The output layer linearly combines these states to read out

the desired output. In [3] the *Backpropagation-Decorrelation* rule

$$\Delta w_{ij}(k+1) = \frac{\eta}{\Delta t}\frac{\varphi(x_j(k))}{\sum_s \varphi(x_s(k))^2+\varepsilon}\gamma_i(k+1), \tag{2}$$
$$\text{where} \quad \gamma_i(k+1) = \sum_{s\in 0}\Big((1-\Delta t)\delta_{is} + \Delta t w_{is}\varphi'(x_s(k))\Big)e_s(k) - e_i(k+1),$$

has been introduced, where η is the learning rate, ε a regularization constant ($\varepsilon = 0.002$ throughout), and $e_s(k)$ are the non-zero error components for $s \in O$ at time $k : e_s(k) = x_s(k) - y_s(k)$ with respect to the teaching signal $y_s(k)$. In [3] it has also been shown that the term $\varphi(x_j(k))/(\sum_s \varphi(x_s(k))^2 + \varepsilon)$ enforces an approximative decorrelation of the neuron output vectors $\varphi(x_j(k))$ over time. The γ_i propagate a mixture of the current errors $e_i(k+1)$ and the errors in the last time step $e_s(k)$ weighted by a typical backpropagation term involving φ'.

3 The operator framework

Using the standard notation for nonlinear feedback systems [8, 9] [1] the network (1) is composed of a linear feedforward and a nonlinear feedback operator Φ:

$$\dot{\mathbf{x}} = -\mathbf{x} + \mathbf{e}, \qquad \mathbf{e} = \mathsf{W}_u\mathbf{u} + \mathsf{W}\varphi(\mathbf{y}), \quad \mathbf{y} = \mathbf{x}.$$

The Laplace transformation of the linear part yields the forward operator $\mathsf{G}_\mathrm{I}(s) = (\mathrm{I} + s\mathrm{I})^{-1}$ while the activation function φ defines the feedback operator Φ, see Fig. 1. Φ does not explicitely have to be stated in the frequency domain because it will be approximated by its gain which is defined by the maximum slope of φ. Denote this network interpretation as $((\mathsf{G}_\mathrm{I}, \mathsf{W}), \Phi)$ for the input-output equation

$$\mathbf{y} = \mathsf{G}_\mathrm{I}(\mathsf{W}_u\mathbf{u} + \mathsf{W}\Phi(\mathbf{y})) = \mathsf{G}_\mathrm{I}\mathsf{W}_u\mathbf{u} + \mathsf{G}\Phi(\mathbf{y}), \quad (\mathsf{G} = \mathsf{G}_\mathrm{I}\mathsf{W}).$$

The network acts as nonlinear feedback system implementing a loop operator $\mathsf{H} \doteq ((\mathsf{G}_\mathrm{I}, \mathsf{W}), \Phi)$ which transforms L_2^n signals[2] $\mathsf{W}_u\mathbf{u}$ into L_2 output signals $\mathbf{y}$. Using the small gain theorem, this system is input-output stable (and the origin is globally exponentially stable for the respective unforced dynamics (1) with $\mathbf{u} \equiv 0$), if the operator gains $\gamma(\mathsf{G}_\mathrm{I}), \gamma(\mathsf{G}) = \gamma(\mathsf{G}_\mathrm{I}\mathsf{W})$ and $\gamma(\Phi)$ are finite and the

$$\text{small gain condition:} \qquad \gamma(\mathsf{G})\gamma(\Phi) < 1 \tag{3}$$

holds. The small gain condition yields the loop gain estimate for $\mathbf{u}' = \mathsf{W}_u\mathbf{u}$

$$\gamma(\mathsf{H}) \le \frac{\gamma(\mathsf{G}_\mathrm{I})}{1-\gamma(\mathsf{G})\gamma(\Phi)} \quad \text{where} \quad \gamma(\mathsf{H}) = \sup_{\mathbf{u}'}\frac{\|\mathsf{H}(\mathbf{u}')\|_2}{\|\mathbf{u}'\|_2} = \|\mathsf{H}\|_2 \tag{4}$$

is the gain induced by the $L_2^p, p = n, 1$ norms for the operator $\mathsf{H} : L_2^n \to L_2$. Note that $\gamma(\mathsf{G}_\mathrm{I}) = \gamma(\Phi) = 1$ by definition. To derive the stability condition

[1] Here we give the framework only for continuous time, an analog derivation is possible for discrete time using the z-transform and sequence spaces, see ([8],chap. 6).

[2] Strictly speaking we can assume this only after assuring stability of the system, see [9].

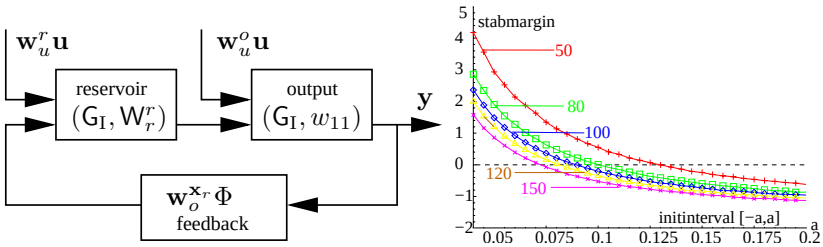

Fig. 2: Left: System composed of inner reservoir subsystem, output subsystem and feedback.Right: Stability margin vs. weight initialization intervals different network sizes (averaged over 50 network initializations.)

now decompose the network in reservoir and output neuron subsystems: let $\mathbf{x}^r = (x_2, ..., x_n)^T$ and W_r^r the submatrix connecting only these inner neurons. Then $((\mathsf{G}_\mathrm{I}, \mathsf{W}_r^r), \Phi)$ denotes the reservoir subsystem while $((\mathsf{G}_\mathrm{I}, w_{11}), \Phi)$ yields the 1-dimensional output subsystem. The dimensions of G_I and Φ have to be adjusted, respectively, but their gains remain equal to one. In Fig. 2 the network is shown as composition of the subsystems connected by the original feedback weights. The composite system is stable, if the subsystems are stable and because $\gamma(\mathsf{G}_\mathrm{I}) = \gamma(\Phi) = 1$ and thus $\gamma(\mathsf{G}) = \gamma(\mathsf{G}_\mathrm{I}\mathsf{W}) = \|\mathsf{W}\|$ we obtain for the subsystems the inequalities $\|\mathsf{W}_r^r\| < 1$ and $w_{11} < 1$ and, as proved in the Appendix,

$$\|\mathbf{w}_{\mathbf{x}_r}^o\| < (1 - \|\mathsf{W}_r^r\|)(1 - |w_{11}|)\|\mathbf{w}_o^{\mathbf{x}_r}\|^{-1} \tag{5}$$

for the overall network. Here $\mathbf{w}_{\mathbf{x}_r}^o$ is the vector of trainable weights and $\mathbf{w}_o^{\mathbf{x}_r}$ the vector of feedback weights from the output to the reservoir. This condition can be easily monitored online, because the matrix and vector norms on the right hand side can be precomputed at initialization time and while learning only the norm of the output weight vector $\|\mathbf{w}_{\mathbf{x}_r}^o\|$ has to be updated. Fig. 2 right shows the right hand side (stability margin) of (5) for different network sizes and initialization intervals.

4 Simulation results on CATS benchmark

The recently introduced CATS benchmark [1] provided data for a time-series competition held at IJCNN 2004. The data consist of 5000 points, with 100 points missing at positions 980-1000, 1980-2000, ... Error criteria are the MSQE for all 100 points where a main difficulty is the tailing last 20 data points 4981-5000 because only on-sided information is available there. As the BPDC algorithm is essentially an online method, it is useful to provide it with estimated targets also for the unknown data. The following strategy is adopted: first the last 4096 data points are fast Fourier transformed (fft), where the first four gaps are filled with suitable random noise around a linear interpolation. This gives a good estimation in the first four gaps (MSQE$\approx$ 390). For the last 20 points we use the prediction from the network states as input for the Fourier transform

	avg / stddev /best network		
network	0.02	0.1-0.055	0.2
50/25	494.56/9.93/478	484.69/15.61/454	485.06/25.67/441
80/30	499.19/7.50/481	499.08/17.14/468	502.24/35.41/443
100/20	521.61/2.03/517	512.69/12.41/488	497.05/33.90/432
100/30	479.92/7.78/464	478.64/13.99/451	488.18/31.62/426
120/40	480.13/7.17/467	498.47/25.78/448	488.51/37.79/412
150/50	487.81/33.27/447	485.18/30.05/439	480.39/76.69/367

Table 1: Average errors over at least 100 networks for the CATS benchmark for different network size/inputs 50/25-150/50 and different initialization ranges. The middle column uses initialization close to the border of the stability range 0.1, 0.08,0.07,0.07,0.065,0.055 (for increasing size of networks.)

such that with increasing accuracy of the network, the Fourier transform in turn becomes more accurate as well (in all gaps). Before each epoch the fft is computed, the upper 75% of the frequencies are pruned and the result is back-transformed to serve as teaching signal for the network.

Table 1 shows results for networks ($\eta = 0.03, \varepsilon = 0.002$) of different sizes and taking a different number of immediate past values as inputs in three initialization conditions: provable stable ($a = 0.02$); at the edge of the stability obtained from Fig. 2, right; and not provably stable ($a = 0.2$) with the presented method. Because small gain stability conditions are always only sufficient and known to be conservative in the last case the networks also may be stable and we did not encounter stability problems in practice. On average all but the 100/20 networks are ranked third with respect to the results of the times series competition at IJCANN behind ([10],408) and ([11], 446). The results show the robustness of BPDC to the initialization which obviously is of crucial importance because the reservoir is not adapted. On the other hand, the stddev increases with the initialization range because the reservoir naturally provides more dynamics with larger weights. These networks are harder to adapt, but the larger variance in initialization can as well lead to better results, note that the best network with an error of 367 outperforms [10].

5 Conclusion

In this contribution we have presented a method to prove and monitor stability for large networks where only the output layer is adapted. Though in principle the stability method is independent of the learning method used, we use it to access stability for the BPDC algorithm, which is a new and highly efficient O(N) learning paradigm for such output weights. The encouraging results on the CATS benchmark, which has proven to be a very hard task for time-series prediction, show that stability, efficient online learning, and accuracy can simultaneously be achieved with the BPDC learning.

Appendix

Consider the composite loop in operator notation and take norms:

$$\begin{aligned}
\|x_o(s)\| &= \|\mathsf{G}_{\mathrm{I}o}(s)e_o(s)\| \quad (e_o(s) = \mathbf{w}_{\mathbf{u}}^o\mathbf{u}(s) + w_{oo}\varphi(x_o(s)) + \mathbf{w}_r^o\Phi_r(\mathbf{x}_r(s))) \\
&\leq \|\mathbf{w}_{\mathbf{u}}^o\|\|\mathbf{u}(s)\| + |w_{oo}|\|x_o(s)\| + \|\mathbf{w}_r^o\|\|\mathbf{x}_r(s)\| \\
&\leq \|\mathbf{w}_{\mathbf{u}}^o\|\|\mathbf{u}(s)\| + |w_{oo}|\|x_o(s)\| + \|\mathbf{w}_r^o\|\frac{1}{1-\|\mathsf{W}_r^r\|}\|\mathbf{u}_r\| \\
&= \|\mathbf{w}_{\mathbf{u}}^o\|\|\mathbf{u}(s)\| + |w_{oo}|\|x_o(s)\| + \frac{\|\mathbf{w}_r^o\|}{1-\|\mathsf{W}_r^r\|}\|\mathsf{W}_{\mathbf{u}}^r\mathbf{u}(s) + \mathbf{w}_o^{\mathbf{x}_r}x_o(s)\| \\
&\leq \|\mathbf{w}_{\mathbf{u}}^o\|\|\mathbf{u}(s)\| + |w_{oo}|\|x_o(s)\| + \frac{\|\mathbf{w}_r^o\|}{1-\|\mathsf{W}_r^r\|}\left(\|\mathsf{W}_{\mathbf{u}}^r\|\|\mathbf{u}(s)\| + \|\mathbf{w}_o^{\mathbf{x}_r}\|\|x_o(s)\|\right),
\end{aligned}$$

where we used that $\gamma(\mathsf{G}_{\mathrm{I}}) = \|\mathsf{G}_{\mathrm{I}}\| = \gamma(\Phi) = \|\Phi\| = 1$ and the loop gain estimation from the small gain theorem for the reservoir subsystem to replace the output of the reservoir $\|\mathbf{x}_r(s)\|$ by its the scaled input $\|\mathbf{u}_r\|/(1-\|\mathsf{W}_r^r\|)$, $\mathbf{u}_r = \mathsf{W}_{\mathbf{u}}^r\mathbf{u} + \mathbf{w}_o^{\mathbf{x}_r}x_o$. Solving for $\|x_o\|$ yields

$$\|x_o(s)\| \leq \left(1 - |w_{oo}| - \frac{\|\mathbf{w}_r^o\|}{1-\|\mathsf{W}_r^r\|}\|\mathbf{w}_o^{\mathbf{x}_r}\|\right)^{-1}\left(\|\mathbf{w}_{\mathbf{u}}^o\| + \frac{\|\mathbf{w}_r^o\|}{1-\|\mathsf{W}_r^r\|}\|\mathsf{W}_{\mathbf{u}}^r\|\right)\|\mathbf{u}(s)\|.$$

Stability requires the denominator of the left hand side be larger zero and we can solve for the vector of the adapted output weights $\mathbf{w}_r^o$ as

$$1 - |w_{oo}| - \frac{\|\mathbf{w}_r^o\|}{1-\|\mathsf{W}_r^r\|}\|\mathbf{w}_o^{\mathbf{x}_r}\| > 0 \Leftrightarrow \|\mathbf{w}_r^o\| \leq \frac{(1-|w_{oo}|)(1-\|\mathsf{W}_r^r\|)}{\|\mathbf{w}_o^{\mathbf{x}_r}\|}.$$

References

[1] CATS benchmark. URL: www.cis.hut.fi/lendasse/competition/competition.html.

[2] B. Hammer and J. J. Steil. Tutorial: Perspectives on learning with recurrent neural networks. In *Proc. of ESANN*, pages 357–368, 2002.

[3] Jochen J. Steil. Backpropagation-decorrelation: Recurrent learning with O(N) complexity. In *Proc. IJCNN*, volume 1, pages 843–848, 2004.

[4] A. B. Atiya and A. G. Parlos. New results on recurrent network training: Unifying the algorithms and accelerating convergence. *IEEE TNN*, 11(9):697–709, 2000.

[5] Ulf D. Schiller and Jochen J. Steil. Analyzing the weight dynamics of recurrent learning algorithms. *Neurocomputing*, 63C:5–23, 2005.

[6] H. Jaeger. Adaptive nonlinear system identification with echo state networks. In *NIPS*, pages 593–600, 2002.

[7] T. Natschläger, W. Maass, and H. Markram. The "liquid computer": A novel strategy for real-time computing on time series. *TELEMATIK*, 8(1):39–43, 2002.

[8] M. Vidyasagar. *Nonlinear Systems Analysis*. Prentice Hall, 2. edition, 1993.

[9] Jochen J. Steil. *Input-Output Stability of Recurrent Neural Networks*. Cuvillier Verlag, Göttingen, 1999. (Also: Phd.-Dissertation, Faculty of Technology, Bielefeld University).

[10] S. Sarkka, A. Vehtari, and J. Lampinen. Time series prediction by Kalman smoother with cross validated noise density. In *Proc. IJCNN*, pages 1653–1658, 2004.

[11] X. Cai, N. Zhang, G. Venayagamoorthy, and D. Wunsch. Time series prediction with recurrent neural networks using a hybrid pso-ea algorithm. In *IJCNN*, pages 1647–1653, 2004.