

Exact Gradient Calculation in Gamma Neural Networks

Tomasz J. Cholewo*, Jacek M. Zurada*, and Andrzej Cichocki†

**Department of Electrical Engineering, University of Louisville
Louisville, Kentucky 40292, USA
e-mail: t.cholewo@ieee.org*

†*Laboratory for Artificial Brain Systems
The Institute of Chemical and Physical Research (RIKEN)
Hirosawa 2-1, Saitama 351-01, Wako-shi, Japan*

Abstract—In this paper we introduce a novel algorithm for computation of the exact error gradient for a gamma network with an arbitrary feedforward topology. An approximate algorithm with reduced computational complexity referred to as Truncated Temporal Backpropagation is also developed. Time series prediction experiments using Mackey-Glass chaotic time series show improved training convergence for our algorithm. The importance of including additional terms in the gradient calculation is demonstrated by illustrating averaged synapse impulse responses for different degrees of truncation and calculating the gradient error in the estimated model introduced by the approximate method.

I. INTRODUCTION

Gamma neural networks are feedforward neural networks with commonly used scalar synapses replaced by linear filters. This gives them the capability of performing dynamic mappings which depend on past input values, making them suitable for time series prediction, nonlinear system identification, and signal processing applications. As of yet no algorithms for calculating the exact gradient in gamma networks have been published. In this paper the algorithm which enables the exact calculation of gradient for a gamma neural network is introduced. Our method is based on an algorithm developed by Wan [1] referred to as Temporal Backpropagation.

A general network architecture and the notation used in the paper are introduced in Section II. Section III presents the new algorithm. Section IV describes the results of time series prediction experiments using Mackey-Glass chaotic time series. Finally, conclusions and future work directions are presented.

II. NETWORK ARCHITECTURE

We use a general feedforward neural network (FNN) as our network architecture. Neurons in such a network form an oriented acyclic graph. The general net-

work structure is static (memoryless) since there are no global recursive connections. As a result neurons can be arranged so that each neuron's inputs originate only at the outputs of previous neurons. This generalizes a multilayer feedforward neural network (MFNN) which has neurons arranged in layers with connections only between neurons in consecutive layers.

Each neuron performs two calculations: first, it sums the output values $y_{ji}(t)$ of the incoming synapses to calculate the activation value (“net” value)

$$a_j(t) = \sum_{i \in \text{IN}_j} y_{ji}(t).$$

of a neuron. IN_j denotes a set of neurons connected to the input of neuron j . Unit bias is realized by means of a “bias synapse” which supplies a constant value $y_{ji}(t) = b_{ji}$ to the summation node of a neuron. Second, a nonlinear function $f_j(\cdot)$ is applied to $a_j(t)$ yielding an output value:

$$z_j(t) = f_j(a_j(t)).$$

In a gamma network each synapse is a gamma filter. Gamma filters are a particular instance of a generalized tapped delay line filters in which the usual unit delay z^{-1} is replaced by a transfer function $G(z)$. For gamma filters a first-order autoregressive operator

$$G(z) = \frac{\mu}{z - (1 - \mu)}$$

is used where μ is a memory depth parameter. The resulting gamma synapse is shown in Figure 1. It is easy to verify that for $\mu = 1$ operator $G(z) = z^{-1}$ and the synapse is equivalent to a FIR filter. Memory depth can be adapted independently for each synapse, allowing the network to use input information on varying time scales and reducing the number of parameters required for long term memory. Our formulation allows ignoring unnecessary time lags which can lead to a further reduction in the number of model parameters.

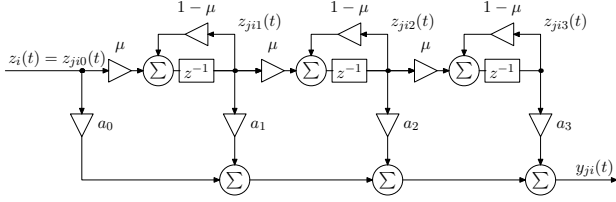


Figure 1: A third-order gamma filter.

An output of a gamma synapse can be expressed in the following recursive formulation

$$y_{ji}(t) = \sum_{l \in \text{MA}_{ji}} a_{jil} z_{jil}(t) \quad (1)$$

where $z_{jil}(t)$ is a value at lag l . When $l = 0$ we obtain that $z_{ji0}(t) = z_i(t)$ and when $l > 0$

$$z_{jil}(t) = (1 - \mu_{ji})z_{jil}(t-1) + \mu_{ji}z_{ji,l-1}(t-1).$$

Gamma synapse combines attractive stability properties of a FIR synapse with some of the general power of an IIR filter [2]. As in an IIR filter the effective depth of the memory is uncoupled from the number of time lags, yet the stability conditions are trivial ($0 < \mu < 2$) and the error surface is quadratic in the function of parameters a_{jil} . The error surface is potentially non-convex in the function of parameter μ_{ji} but practice shows that it usually poses no problems [2].

An architecture which employs a separate gamma filter for each input to create a set of inputs for a regular MFNN network is known as a focused gamma network [3]. It has an advantage of simple, low cost training but was shown to be less powerful than a network in which all synapses are gamma filters [4].

III. GRADIENT CALCULATION

Dynamic neural networks can approximate a wide class of nonlinear dynamic mappings. Adaptive numerical optimization (training) algorithms are used to find the appropriate values of network parameters by searching for a minimum on a multidimensional surface of some error function. The values of the derivatives of the error function with respect to each parameter (error gradient) are usually required in such approaches.

In the batch (also known as “off-line”) approach the criterion to minimize is the error E over all samples of a time series (total error):

$$E \triangleq \sum_t E(t)$$

where the instantaneous error $E(t)$ is usually defined as a sum of squares of all units’ output errors:

$$E(t) \triangleq \frac{1}{2} \sum_i [e_i^o(t)]^2.$$

The unit’s output error is defined as

$$e_i^o(t) \triangleq \begin{cases} d_i(t) - z_i(t) & \text{if } i \text{ is an output neuron,} \\ 0 & \text{otherwise.} \end{cases}$$

The proposed algorithm follows the Temporal Back-propagation method developed by Wan [1] for networks with FIR filter synapses. The derived formulas for the gradient of the total error are exact under the assumption that model parameters remain constant during the training epoch. This assumption is satisfied by the batch (off-line) training method which is appropriate for most time series prediction tasks but not for control problems or large data sets requiring on-line adaptation.

Wan [1] proposed a new way of applying the chain rule of derivation by considering derivatives of the total error with respect to units’ activations at different moments of time. The derivative of the total error E with respect to any parameter p_{ji} from synapse connecting neuron i to neuron j can be obtained as:

$$\frac{\partial E}{\partial p_{ji}} = \sum_t \frac{\partial E}{\partial a_j(t)} \frac{\partial a_j(t)}{\partial p_{ji}} = \sum_t \delta_j(t) \frac{\partial y_{ji}(t)}{\partial p_{ji}}$$

where

$$\delta_j(t) \triangleq \frac{\partial E}{\partial a_j(t)} = \frac{\partial E}{\partial z_j(t)} \frac{\partial z_j(t)}{\partial a_j(t)} = e_j(t) f'_j(a_j(t)).$$

The quantity $f'_j(a_j(t))$ is a derivative of the activation function. The local error $e_i(t)$ is a generalization of the training error at the output neurons to all neurons. For output neurons this error is simply equal to the output error $e_i^o(t)$. For hidden neurons $e_i(t)$ is obtained using the error backpropagation formula introduced by Werbos [5]:

$$\begin{aligned} e_i(t) \triangleq \frac{\partial E}{\partial z_i(t)} &= \sum_{j \in \text{OUT}_i} \sum_{t'} \frac{\partial E}{\partial a_j(t')} \frac{\partial a_j(t')}{\partial z_i(t)} \\ &= \sum_{j \in \text{OUT}_i} \sum_{t'} \delta_j(t') h_{ji}(t, t') \end{aligned} \quad (2)$$

where

$$h_{ji}(t, t') = \frac{\partial y_{ji}(t')}{\partial z_i(t)}$$

is an impulse response of a synapse s_{ji} at time t and indices t and t' run over the duration of the whole training set. The Equation (2) can be interpreted as back-filtering of error terms through a reversed dynamic synapse. The result is a non-causal dependence of $e_i(t)$ on terms $\delta_j(t')$ for $t' > t$. This problem can be solved for finite h_{ji} (e.g., for a FIR synapse) by delaying computation of $e_i(t)$ until all $\delta_j(t')$ values are

known. For infinite impulse response synapses the exact computation has to stretch up to the last input sample.

For a gamma filter the impulse response is:

$$h_{ji}(t) = \sum_{l \in \text{MA}_{ji}} a_{jil} \beta_{jil}(t)$$

where $\beta_{jil}(t) \triangleq \frac{\partial z_{jil}(t)}{\partial z_i(0)}$ is the impulse response at a lag l . For $l = 0$ the output is obviously the same as the synapse's input, hence $\beta_{ji0}(t) = \delta(t)$ where $\delta(t)$ is the unit impulse. When $l > 0$ the following recursive formula can be used:

$$\beta_{jil}(t) = (1 - \mu_{ji})\beta_{jil}(t-1) + \mu_{ji}\beta_{ji,l-1}(t-1).$$

The quantities $\frac{\partial y_{ji}(t)}{\partial p_{ji}}$ are obtained by taking a derivative of Equation (1). For weights a_{jil} we get:

$$\frac{\partial y_{ji}(t)}{\partial a_{jil}} = z_{jil}(t).$$

For memory depth parameters μ_{ji} the derivatives are:

$$\frac{\partial y_{ji}(t)}{\partial \mu_{ji}} = \sum_{l \in \text{LAGS}_{ji}} a_{jil} \alpha_{jil}(t)$$

where

$$\alpha_{jil}(t) \triangleq \frac{\partial z_{jil}(t)}{\partial \mu_{ji}} = (1 - \mu_{ji})\alpha_{jil}(t-1) + \mu_{ji}\alpha_{ji,l-1}(t-1) + z_{ji,l-1}(t-1) - z_{jil}(t-1)$$

if $t > 0$ and $l > 0$, and $\alpha_{jil}(t) = 0$ otherwise.

For synapses with an infinite impulse response the cost of exact gradient calculation is proportional to the number of samples in the time series which for long data sets can lead to low computational efficiency. In such a case one may employ an approximation which we call Truncated Temporal Backpropagation TTBP(n) for which the local error is calculated as

$$e_i(t) = \begin{cases} e_i^o(t) & \text{if } i \text{ is an output,} \\ \sum_{j \in \text{OUT}_i} \sum_{t' \leq t \leq T_{ji}} \delta_j(t') h_{ji}(t' - t) & \text{otherwise.} \end{cases}$$

This formula utilizes only a limited number T_{ji} of terms of the synapse's impulse response $h_{ji}(l)$ in the process of backward error filtering and uses the fact that the gamma filter is causal and time invariant. Computational and space complexities for this method can be balanced against the desired precision of the gradient calculation. The number of used terms can be specified independently for each synapse. As a special case, TTBP(n) is reduced for $n = 0$ to the instant cost/instant gradient approximation presented

by Lawrence et al. [4]. Only gradient in the hidden layer is influenced by this approximation. Output layer gradient calculation does not depend on the backward error propagation and therefore is always calculated exactly. During training, however, the inexact parameter changes in hidden layer synapses influence the output layer gradient indirectly through the forward propagation step.

A similar algorithm limited to layered IIR networks with synapses having weights at all time lags was developed in [6].

IV. SIMULATIONS

A simple network with one input, two hidden neurons and one output having four gamma synapses of a third order was used for the task of prediction of Mackey-Glass chaotic time series. This data set is a solution of a delay-differential equation

$$\dot{x}(t) = -bx(t) + \frac{ax(t - \Delta)}{1 + x(t - \Delta)^c}$$

where $\Delta = 30$, $a = 0.2$, $b = 0.1$, $c = 10$, initial conditions $x(t) = 0.9$ for $0 \leq t \leq \Delta$, and sampling rate $\tau = 6$. The training set consisted of the first 100 samples and the test set of the next 100 samples.

Parameter adaptation in simulations was performed with a scaled conjugate gradient method which significantly improves the speed and convergence of training as compared to the usual steepest descent method.

The importance of including additional terms in the gradient calculation is confirmed by Figure 2 which shows the final training error averaged for five different initial conditions. The error becomes smaller that is the training convergence is improving when the number of used impulse terms is increased. Figure 3 shows that the weights for which the exact gradient is zero and which hence are a fixed point of the training, result in non-zero approximate gradient which means that the exact solution can not be obtained when too few terms are used.

On the other hand, these two figures show also that it is possible to use only several impulse response terms and still obtain good results. It is a consequence of the fact that in most cases, only a finite part of synapse responses is significantly different from zero. Impulse responses of synapses of one of the networks trained on the Mackey-Glass data are shown as an example in Figure 4.

V. CONCLUSIONS

The algorithms presented herein can be used for a wide variety of tasks involving dynamic neural networks. Exact gradient calculation allows for better training convergence. Potentially more accurate predictions can also be obtained though the representational capabilities of the network remain unchanged.

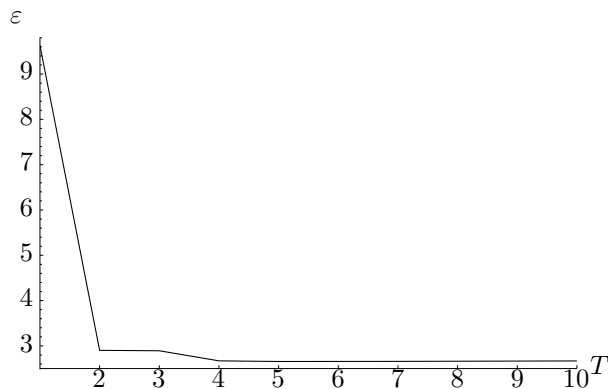


Figure 2: The averaged final training error in function of the number of used impulse response terms.

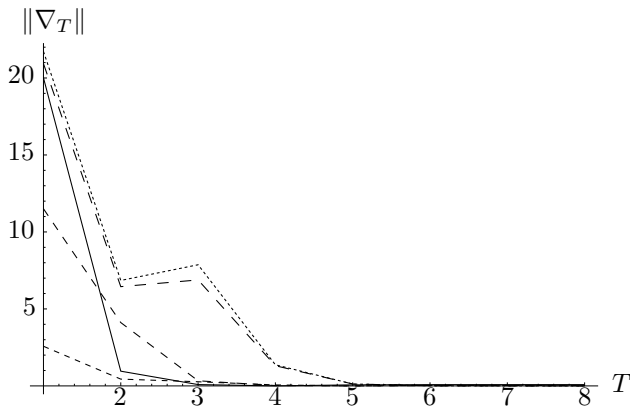


Figure 3: Norm of approximate gradient at a fixed point of training with an exact gradient in function of the number of used impulse response terms.

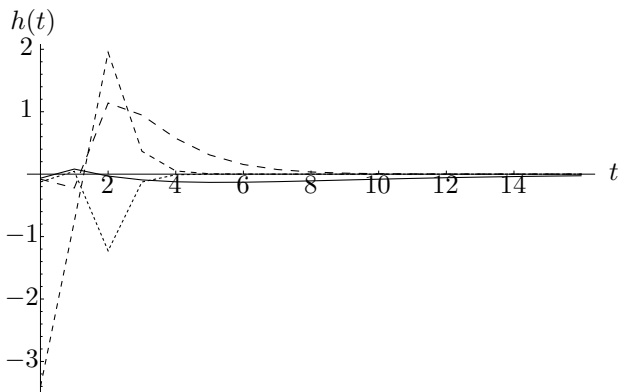


Figure 4: Impulse responses of third order synapses in a gamma network trained on the Mackey-Glass time series.

For small problems and/or data sets the algorithm can be used in its exact form without simplifications. For larger data sets an approximate version of our algorithm can be used to find an optimal balance between algorithm's speed and accuracy.

Future research directions include calculation of second order derivatives in dynamic and locally recurrent globally feedforward (LRGF) networks for model complexity control and use of an output error equation approach in time series prediction.

References

- [1] Eric A. Wan. *Finite Impulse Response Neural Networks with Applications in Time Series Prediction*. PhD thesis, Stanford University, 1993. [I](#), [III](#), [III](#).
- [2] Jose C. Principe, Bert de Vries, and Pedro Guedes de Oliveira. The gamma filter—a new class of adaptive IIR filters with restricted feedback. *IEEE Transactions on Signal Processing*, 41(2):649–656, 1993. [II](#), [II](#).
- [3] Bert de Vries and Jose Principe. The gamma model—a new neural network for temporal processing. *Neural Networks*, 5(4):565–576, 1992. [II](#).
- [4] Steve Lawrence, Ah Chung Tsoi, and Andrew D. Back. The Gamma MLP for speech phoneme recognition. In D. Touretzky, M. Mozer, and M. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 785–791. MIT Press, 1996. [II](#), [III](#).
- [5] Paul Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Harvard University, Cambridge, Mass., 1974. [III](#).
- [6] Paolo Campolucci, Aurelio Uncini, and Francesco Piazza. A unifying view of gradient calculations and learning for locally recurrent neural networks. In *Proceedings of Italian Workshop on Neural Networks (WIRN'97)*, Vietri Sul Mare (Salerno), Italy, May 1997. Springer-Verlag Ed. [III](#).