# The Gamma Model- A New Neural Model for Temporal Processing

**Bert De Vries and Jose C. Principe**
Electrical Engineering Department, University of Florida
Gainesville, Florida 32611
principe@brain.ee.ufl.edu

**Keywords:** temporal processing, dynamic neural nets, short term memory, ARMA model, gamma model, focused backpropagation

**Abstract**

*In this paper we develop the gamma neural model, a new neural net architecture for processing of temporal patterns. Time varying patterns are normally segmented into a sequence of static patterns that are successively presented to a neural net. In the approach presented here segmentation is avoided. Only current signal values are presented to the neural net, that adapts its own internal memory to store the past. Thus, in the gamma neural net, an adaptive short term mechanism obviates a priori signal segmentation. We evaluate the relation between the gamma net and competing dynamic neural models. Interestingly, the gamma model brings many popular dynamic net architectures, such as the time-delay-neural-net and the concentration-in-time-neural-net, into a unifying framework. In fact, the gamma memory structure appears as general as a temporal convolution memory structure with arbitrary time varying weight kernel w(t). Yet, the gamma model remains mathematically equivalent to the additive (Grossberg) model with constant weights. We present a backpropagation procedure to adapt the weights in a particular feedforward structure, the focused gamma net.*

## 1 INTRODUCTION TO THE TEMPORAL PROCESSING PROBLEM

This paper addresses the general problem of processing time varying signals with a neural network. We consider signals for which the temporal evolution encodes valuable information regarding the source. A prototype of this class of problems is speech recognition, but a lot of problems in image processing, prediction and system identification fit this framework as well[1]. The central difficulty is how to represent time in a neural net structure. The conventional approach is the “snap shot representation” where the input signal undergoes segmentation in a pre-preprocessor (a frame in imagery or a segment in speech). The continuous flow of information is then reconstructed by concatenating fixed size signal segments. The ideas are further illustrated in Figure 1.

Let the signal $x(s,t)$ be defined for $t_0 \le t \le t_f$ in the temporal dimension and for $s_0 \le s \le s_f$ in the spatial domain. We refer to the signal representation in Figure 1 as a <u>space-</u>

1. Although most of the work presented here is general enough to apply to various kinds of processing, in the back of our mind we have kept real-time classification of speech as our ultimate goal. Most of our ideas are therefore presented in a speech recognition context.

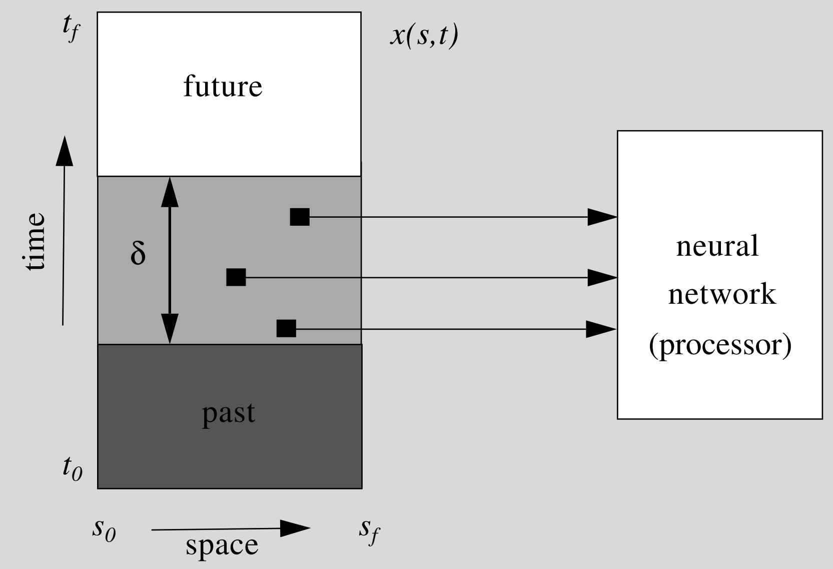

Figure 1 Schematic picture of signal presentation to a neural net. d is the width of a sliding window. For tasks such as signal classification, the neural net needs a short term memory mechanism to store information regarding the past of the signal.

ogram, or spectrogram when $s$ is the frequency. At any time, only a part of the space-ogram is available for input to the neural net. The selection mask is called a window. Three possibilities for the size of the window in the temporal domain arise. First, the width δ of the window equals the duration $t_f - t_0$ of the signal. This case effectively transforms the temporal dimension to an extra spatial dimension. No temporal processing capabilities of the neural net are required since the problem is transferred to a static processing problem. The second possibility concerns $0 < \delta < t_f - t_0$, which is called the sliding window technique. In this case, the selection mask moves in some fashion over the space-ogram so as to cover the input signal space as time evolves. The extreme case of the sliding window technique occurs when $\delta \to 0$, that is, only the current signal values are available for neural net input. We will refer to this choice as current-time processing, and effectively when $\delta \to 0$ there is no segmentation.

The choice of the sliding window width influences the neural network performance. We identify two problems associated with the selection of δ. First, for large δ, the dimensionality of the neural net also increases. This fact minimally complicates the neural net training. It has been shown that neural net learning time unfortunately scales worse than proportional with the dimension of the weight vectors (Perugini and Engeler, 1989). There are more problems associated with large networks, such as the required increased dimension of the training set. For smaller windows the neural network may not have enough information to appropriately learn the signal dynamics because only a fraction of the decision space is available. However, the network dimension gets progressively smaller which eases the learning requirements in terms of training set size and number of learning steps.

A second issue that complicates the choice of δ involves segmentation of non-stationary signals. Normally the length of the stationarity interval is not known a-priori, and can very well change with time. A large δ tends to average the time-varying statistics of non-stationary signals before the signals enter the neural net. A smaller δ makes the classification very sensitive to the actual signal segment utilized, and tends to make the classification less robust. The balance is very difficult to achieve, and in general varies with time. It follows that adaptive segmentation appears as the appropriate solution. However due to the computational burden of adaptive segmentation, it is hardly ever utilized (Rabiner and Schafer, 1978). The common practice in speech is to bias the selection to fixed, relative small, segments (10 milliseconds).

One way of coping with the time varying nature of the input data is to incorporate an adaptive short term memory mechanism in the neural network. The system that we propose stores the signal history in a short term memory structure. Both memory depth and resolution can be adapted by a common neural net training procedure such as backpropagation. Thus, effectively the segmentation dilemma is solved by incorporating this stage as part of the adaptive neural net processing system. Both the temporal weights (resolution, memory depth) and the spatial weights are unified in the same optimization problem, leaving behind the need for artificial and suboptimal a priori selection of the samples relevant for the classification. In our method the non-linear dynamics of neural nets learn and deal with non-stationarities in a computational manner. Indeed, exploitation of nonlinear processing capabilities is the main motivation for the choice of a neural net processor in the first place. The challenge is to develop a neural net model with an adequate memory mechanism to discriminate a given set of space-time patterns.

In the next section, the basic concepts of short-term memory in neural nets are reviewed. This section is followed by the introduction of a new neural model, the gamma model, which in our opinion provides a unifying framework for the reviewed models. We derive a backpropagation procedure for the focused gamma model architecture, inspired by Mozer's work (Mozer, 1989).

## 2 A REVIEW OF SHORT-TERM MEMORY IN NEURAL NETWORKS

### 2.1 introduction

The basic neural network model for classification of static patterns is the multi-layer perceptron. The activation of the units are computed by -

$$x_i = \sigma\left(\sum_{j<i} w_{ij} x_j\right) + I_i, \qquad \text{Eq.1}$$

where $x_i$ is the activation of neuron (unit, node) $i$. The weight factor $w_{ij}$ connects node $j$ to node $i$; σ() is a (non)-linear squashing function and $I_i$ represents the external input. We assume a system dimensionality of $N$. Sometimes we will use the shorthand notation $net_i = \sum_{j<i} w_{ij} x_j$.

This model is static in the sense that the activations are a function of the current window only. The simplest way to add dynamics (memory) to the model is to add a capacitive term $\tau \frac{dx_i}{dt}$ to the left-hand side of Eq.1. After rearrangement of terms, the so-called (dynamic) additive model is obtained -

$$\tau \frac{dx_i}{dt} = -x_i + \sigma\left(\sum_j w_{ij} x_j\right) + I_i . \qquad \text{Eq.2}$$

The term additive model is used to discriminate from multiplicative (or shunting) models where $w_{ij} = w_{ij}(x_i)$ (Grossberg, 1982).The additive model is used in the great majority of practical applications of neural networks today. Sejnowski provides a biological motivation for the additive model (Sejnowski, 1981). Biologically, the neural time constants $\tau$ are fixed and equal approximately 5 msecs (Shen, 1989). However, recognition of a spoken word requires the ability to remember the contents of a passage for approximately 1 second. To accomplish this, neural temporal resolution decreases while the "temporal window of susceptibility" increases toward the cortex. Apparently, the brain is able to modulate temporal resolution and depth of short-term memory using processing units with fixed small time constants. There are two main strategies to extend the short-term memory capacities of the additive model. The first strategy involves positive feedback, the second incorporates explicit delays in the model. Next, we will analyze the characteristics of both approaches.

## 2.2 short term memory by positive feedback

The additive model can be extended with a positive state feedback term, yielding -

$$\tau \frac{dx_i}{dt} = -x_i + \sigma(net_i) + I_i + kx_i , \qquad \text{Eq.3}$$

where $k$ is a positive constant. In the biological literature, such local positive feedback is often named reverberation, while neural net researchers speak of self-excitation. Eq.3 can be rewritten as -

$$\left(\frac{\tau}{1-k}\right) \frac{dx_i}{dt} = -x_i + \hat{\sigma}(net_i) + \hat{I}_i , \qquad \text{Eq.4}$$

where $\hat{\sigma}(net_i) = \frac{\sigma(net_i)}{1-k}$, and $\hat{I}_i = \frac{I_i}{1-k}$. For $\tau = 5$ msec and $k = 0.995$, we get the new time constant $\hat{\tau} = \frac{\tau}{1-k} = 1$ sec. Units that self-excite over a time span that is relevant with respect to the processing problem are referred to in the neural net literature as context units. Several investigators have explored the temporal computational properties of additive feedforward nets, extended by context units (Jordan, 1986; Elman, 1990; Mozer, 1989, Stornetta et al., 1988). In Hertz et al. (1991), neural models of this kind are collectively referred to as sequential nets. While the positive feedback mechanism is simple and used in biological information processing, there are two computational problems associated with this method. First, the new time constant is very sensitive to $k$. For our example, an increase of 0.5% in $k$ to $k = 1$ makes the model unstable. The time-varying nature of biological parameters makes it therefore unlikely that reverberation is the predominant mechanism for short term memory over long periods. The second handicap of Eq.4 is that the new model is still governed by first-order dynamics. As a result, weighting in the temporal domain is limited to a recency gradient (exponential for linear feedback), that is, the most recent items carry a larger weight than previous inputs. Note that for a neural net composed of *N* neurons, the number of weights in the spatial domain is $O(N^2)$, while the temporal domain is governed only by $\tau$. The restriction to recency weighting obviously implies a limitation on the representation structure of the past in the net. As an example, optimal temporal weighting for the discrimination of the words "cat" and "mat" will not be a recency but rather a primacy gradient.

In conclusion, short term memory by local positive feedback is simple and has been applied successfully in artificial neural nets. However, reverberation may lead to instability. Secondly, this mechanism restricts computational flexibility in the temporal domain. In the next section, short term memory by delays will be reviewed.

## 2.3 short term memory by delays

A general delay mechanism can be represented by temporal convolutions instead of (instantaneous) multiplicative interactions. Consider the following extension of the additive model -

$$\tau \frac{dx_i}{dt} = -x_i + \sigma\left(\sum_j \int_0^t w_{ij}(t-s)\, x_j(s)\, ds\right) + I_i . \qquad \text{Eq.5}$$

We will call this model the (additive) convolution model. In the convolution model the net input is given by -

$$net_i(t) = \sum_j \int_0^t w_{ij}(t-s)\, x_j(s)\, ds. \qquad \text{Eq.6}$$

In a discrete time model, this translates to the following signal -

$$net_i(t) = \sum_j \sum_{n=0}^{t} w_{ij}(t-n)\, x_j(n) . \qquad \text{Eq.7}$$

There is ample biological support for the substitution of weight constants *w* by time varying weights *w*(*t*). Miller has reviewed experimental evidence that "... cortico-cortical axonal connections impose a range of conduction delays sufficient to permit temporal convergence at the single neuron level between signals originating up to 100-200 msec apart" (Miller, 1987). Several artificial neural net researchers have also experimented with additive delay models of type Eq.5. However, due to the complexity of general convolution models, only strong simplifications of the weight kernels have been proposed. For instance, Lang et al. used the discrete delay kernels $w(t) = \sum_k w_k \delta(t - t_k)$ in the time delay neural network (TDNN) (Lang et al., 1990). The TDNN, considered the state-of-the-art, is a multilayer feedforward net that is trained by error backpropagation. The authors reported excellent results on a phoneme recognition task. Recently, the CMU-group introduced the TEMPO 2 model, where adaptive gaussian distributed delay kernels store the past (Bodenhausen and Waibel, 1991). Distributed delay kernels such as used in the TEMPO 2 model improve on the TDNN with respect to the capture of temporal context. Tank and Hopfield also prewired $w(t)$ as a linear combination of dispersive kernels, in particular $w(t) = \sum_k w_k f_k(t) = \sum_k w_k \left(\frac{t}{k}\right)^{\alpha} e^{\alpha\left(1-\frac{t}{k}\right)}$. This technique was utilized as a preprocessor to a Hopfield net for classification of temporal patterns (Tank and Hopfield, 1987). The weight factors $w_k$ were non-adaptive and determined a priori. They successfully built such a system in hardware for an isolated word recognition task. In a later publication successful experiments were reported with adaptive gaussian distributed delay kernels (Unnikrishnan et al., 1991).

The convolution model in its general formulation is more flexible in the temporal domain than the first-order additive model, since the weighting of the past is not restricted to a recency gradient. However, a high price has to be paid for the increased flexibility. We identify three complications for the convolution model when compared to the additive model -

- Analysis. The convolution model is described by a set of functional differential equations (FDE) instead of ordinary differential equations (ODE) for the additive model. Such equations are in general harder to analyze - a handicap when we need to check (or design for) certain system characteristics such as stability and convergence.

- Numerical Simulation. For an $N$-dimensional convolution model, the required number of operations to compute the next state for the FDE set scales with $O(N^2T)$, where $T$ is the number of time steps necessary to evaluate the convolution integral (using Euler method: $x(t+h) = x(t) + h\frac{dx}{dt}$). An $N$-dimensional additive model scales by $O(N^2)$.

- Learning. The weights in the convolution model are the time-varying parameters $w(t)$. Thus, the dimensionality of the free parameters grows linearly with time. For a long temporal segment, the large weight vector dimensionality impairs the ability to train the network.

The difficulty with the convolution model is that the number of parameters of freedom linearly increase with time. This is obvious from Eq.7 where the number of weights $w_{ij}$ equals t. This presents a severe modeling problem since the number of parameters of a system (to be modeled) do not always increase with the memory depth of the system. Although a convolution model is interesting as a biological model and powerful as a computational model, the previous arguments make this model not very attractive as a model for engineering applications. It makes sense to investigate under what conditions the time varying kernel $w(t)$ can be adequately approximated by a fixed dimensional set of constant weights. This topic is discussed in the next section.

## 2.4 the convolution model versus the ARMA model

We have seen in section 2.3 that a general delay mechanism can be written as -

$$net(t) = \int_0^t w(t-s)\, x(s)\, ds \qquad \text{Eq.8}$$

for the continuous time domain and -

$$net(t) = \sum_{n=0}^{t} w(t-n)\,x(n) \qquad \text{Eq.9}$$

in the discrete time domain. We argued that the problem associated with time-dependent weight functions is that the number of parameters grows linearly with *t*. In this section we investigate under what conditions Eq.8 can be reduced to an equivalent memory model with a fixed number (not dependent on *t*) of parameters. This problem was studied by Fargue (1973) and the answer is provided by the following theorem -

Theorem 1 *The (scalar) integral equation -*

$$net(t) = \int_0^t w(t-s)\,x(s)\,ds \qquad \text{Eq.10}$$

*can be reduced to a K- dimensional system of ordinary differential equations with constant coefficients if (and only if) w(t) is a solution of -*

$$\frac{d^K w}{dt^K}(t) = \sum_{k=0}^{K-1} a_k \frac{d^k w}{dt^k}(t)\,, \qquad \text{Eq.11}$$

*where* $a_0, a_1, \ldots, a_{K-1}$ *are constants.*

Proof. We provide a constructive proof for sufficiency. Let us write the initial conditions for Eq.11 as $\hat{w}_k \equiv \frac{d^k w}{dt^k}(0)$, for $k = 0,\ldots,K\text{-}1$, and define the variables $w_k(t) = \frac{d^k w}{dt^k}(t)$, which allows to rewrite Eq.11 as the following set of *K* first-order differential equations -

$$\frac{dw_k}{dt}(t) = w_{k+1}(t)\,, for\ k = 0,\ldots,K\text{-}2,$$

$$\frac{dw_{K-1}}{dt}(t) = \sum_{k=0}^{K-1} a_k w_k(t)\,. \qquad \text{Eq.12}$$

Next, we introduce the auxiliary variables -

$$x_k(t) \equiv \int_0^t w_k(t-s)\,x(s)\,ds, for\ k = 0,\ldots,K\text{-}1. \qquad \text{Eq.13}$$

Note that the system output is given by -

$$net(t) = \int_0^t w_0(t-s)\,x(s)\,ds = x_0(t)\,. \qquad \text{Eq.14}$$

The variables $x_k(t)$ can be recursively computed. Differentiating Eq.13 with respect to $t$ using Leibniz' rule gives -

$$\frac{dx_k}{dt}(t) = \int_0^t \frac{\partial}{\partial t} w_k(t-s)\, x(s)\, ds + w_k(0)\, x(t)\,, \qquad \text{Eq.15}$$

which using the recurrence relations from Eq.12 evaluates to -

$$\frac{dx_k}{dt}(t) = x_{k+1}(t) + \hat{w}_k x(t)\,, for\ k = 0,\ldots,K\text{-}2,$$

$$\frac{dx_{K-1}}{dt}(t) = \sum_{k=0}^{K-1} a_k x_k(t) + \hat{w}_{K-1} x(t)\,. \qquad \text{Eq.16}$$

Thus, the integral equation Eq.10 can be reduced to a system of differential equations with constant coefficients ( Eq.16) if the weight kernel $w(t)$ is a solution of the recurrence relation Eq.11. □ (end proof).

In the signal processing community, the obtained system Eq.14 and Eq.16 is called an autoregressive moving average (ARMA) model ( Figure 2). The memory of an ARMA system is represented in the state variables $x_k(t)$. It is interesting to observe the relation between the ARMA model parameters and the convolution model. The autoregressive parameters $a_k$ are the coefficients of the recurrence relation Eq.11 for $w(t)$. The moving average parameters $\hat{w}_k$ equal the initial conditions of Eq.11.

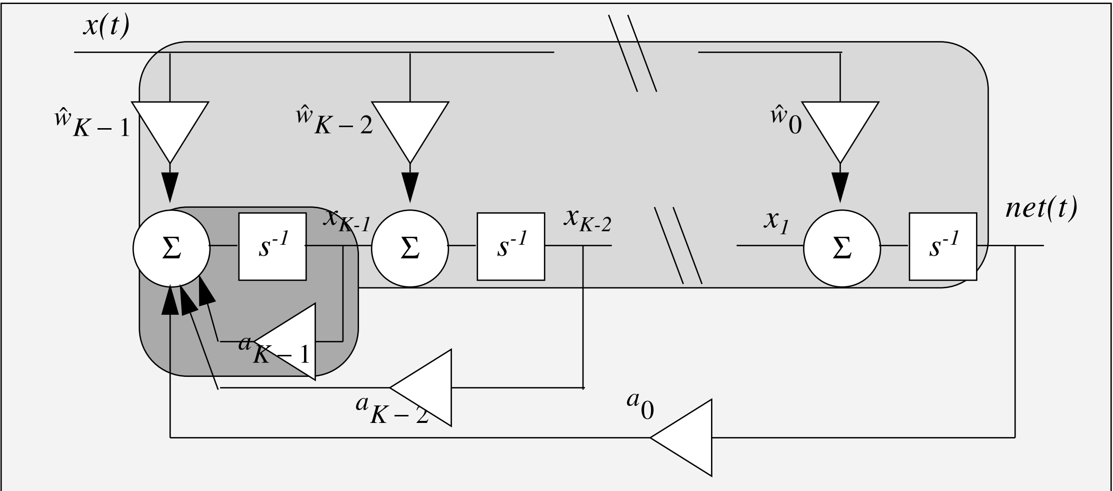

Figure 2 An ARMA model implementation of a convolution model with recursive $w(t)$.

The following theorem reveals what is meant by imposing the condition Eq.11 on $w(t)$.

Theorem 2 *Solutions of the system* $\frac{d^K w}{dt^K}(t) = \sum_{k=0}^{K-1} a_k \frac{d^k w}{dt^k}(t)$ *can be written as a linear combination of the functions* $t^{k_i} e^{\mu_i t}$, *where* $1 \le i \le m$, $0 \le k_i < K_i$ *and* $\sum_{i=1}^{m} K_i = K$.

The proof of Theorem 2 is provided in most textbooks on ordinary differential equations (e.g. Braun, 1983). The $\mu_i$'s are the eigenvalues of the system. In particular, the $\mu_i$'s are the solutions of the characteristic equation of Eq.11 -

$$\mu^K - \sum_{k=0}^{K-1} a_k \mu^k = 0. \qquad \text{Eq.17}$$

$m$ is the number of different eigenvalues and $K_i$ the multiplicity of eigenvalue $\mu_i$. The functions $t^{k_i} e^{\mu_i t}$ are the eigenfunctions of Eq.11, and $i$ enumerates the various eigenmodes of the system.

At this point, let us reflect for a moment on the ARMA structure as displayed in Figure 2. In the context of this exposition, we like to think of an ARMA model as a dynamic model for a memory system.We just proved that this configuration is equivalent to a convolution memory model if the condition described by Eq.11 is obeyed. Yet, we do not know of neural network models that utilize the full ARMA model to store the past of $x(t)$. The reason is that the global recurrent loops in the ARMA model make it difficult to control stability in this configuration. This is particularly true when the autoregressive parameters $a_k$ are adaptive. There are some substructures of the ARMA model for which stability is easily controlled. Examples are the feedforward tapped delay line and the first order autoregressive model (or context unit). These two structures, which are shaded differently for clarity in Figure 2, have been used extensively in neural networks as a memory mechanism. We have already discussed the virtues and shortcomings of either approach. In this paper, we introduce a different approximation to the ARMA memory model. A look ahead to Figure 4 shows that our memory model utilizes a cascade of locally recursive structures in contrast to the global loops in the full ARMA model. As a result, the memory model that we introduce will provide a more flexible approximation to the ARMA or convolution model. Yet, the stability conditions will prove to be trivial. In the next chapter, we introduce the mathematical framework for this new memory model.

# 3 THE GAMMA NEURAL MODEL

## 3.1 gamma model definitions

We have looked at a specific subset of the class of functions that admit Eq.11. Let us consider the following case[1] -

$$w(t) = \sum_{k=1}^{K} w_k g_k^{\mu}(t), \qquad \text{Eq.18}$$

where -

$$g_k^{\mu}(t) \equiv \frac{\mu^k}{(k-1)!} t^{k-1} e^{-\mu t}, \; k = 1,...K, \; (\mu > 0). \qquad \text{Eq.19}$$

It is easily checked that the kernels $g_k(t)$ (we drop the superscript $\mu$) are a solution to Eq.11. Since the kernels $g_k(t)$ are the integrands of the (normalized) $\Gamma$-function ($\Gamma(x) \equiv \int_0^{\infty} t^{x-1} e^{-t} dt$), we refer to them as gamma kernels. In view of the solutions of Eq.11, the gamma kernels are characterized by the fact that all eigenvalues are the same, that is, $\mu_i = \mu$. Thus, the gamma kernels are the eigenfunctions of the following recurrence relation -

$$\left(\frac{d}{dt} + \mu\right)^K g(t) = 0. \qquad \text{Eq.20}$$

The factor $\frac{\mu^k}{(k-1)!}$ normalizes the area such that -

$$\int_0^t g_k(s)\, ds = 1, \; k = 1,2,..,K \qquad \text{Eq.21}$$

The shape of the gamma kernels $g_k(t)$ is pictured in Figure 3 for $\mu = 0.7$.

It is straightforward to derive an equivalent ARMA model for $net(t)$ when $w(t)$ is constrained by Eq.18. The procedure is similar to the proof of Theorem 1. First, we write the $g_k(t)$'s as a set of first-order differential equations -

$$\frac{dg_1}{dt} = -\mu g_1$$

$$\frac{dg_k}{dt} = -\mu g_k + \mu g_{k-1}, \; k = 2,..K. \qquad \text{Eq.22}$$

Substitution of Eq.18 into Eq.8 yields -

$$net(t) = \sum_{k=1}^{K} w_k x_k, \qquad \text{Eq.23}$$

where we defined the gamma state variables -

1. The upper bound $K$ for $k$ in Eq.18 is not meant to be the same $K$ as in Eq.11.

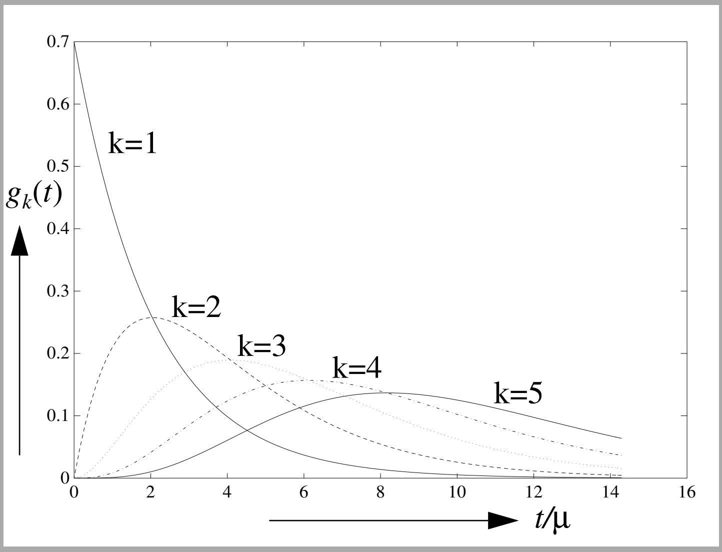

Figure 3 the gamma kernels $g_k(t)$ for $\mu$=0.7

$$x_k(t) = \int_0^t g_k(t-s)\,x(s)\,ds,\ k = 1,..,K. \qquad \text{Eq.24}$$

The gamma state variables hold memory traces of the neural states $x(t)$. How are the variables $x_k(t)$ computed? Differentiating Eq.24 leads to -

$$\frac{dx_k}{dt}(t) = \int_0^t \frac{\partial}{\partial t} g_k(t-s)\,x(s)\,ds + g_k(0)\,x(t)\,, \qquad \text{Eq.25}$$

which, since $g_k(0) = 0$ for $k \geq 2$ and $g_1(0) = \mu$, evaluates to -

$$\frac{dx_k}{dt}(t) = -\mu x_k(t) + \mu x_{k-1}(t) \text{ for } k = 1,..,K, \qquad \text{Eq.26}$$

where we defined $x_0(t) \equiv x(t)$. The initial conditions for Eq.26 can be obtained from evaluating $x_k(0) = g_k(0)\,x(0)$, which reduces to -

$$x_0(0) = x(0)\,, \qquad x_1(0) = \mu x(0)$$

$$x_k(0) = 0, \text{ for } k = 2,..,K. \qquad \text{Eq.27}$$

Thus, when $w$(t) admits Eq.18, $net(t)$ can be computed by a $K$-dimensional system of ordinary differential equations Eq.26.

We would like to know if and how the selection of gamma kernels (constraint Eq.18) affects the generality of $w(t)$. The following theorem states that the approximation of arbitrary $w(t)$ by a linear combination of gamma kernels can be made as close as desired.

Theorem 3 *The system $g_k(t)$, for $k = 1,2,..,K$ is closed in* $L_2\,[0, \infty]$.

Theorem 3 is equivalent to the statement that for all $w(t)$ in $L_2\,[0, \infty]$ (that is, any $w(t)$ for which $\int_0^\infty |w(t)|^2 dt$ exists), for every $\varepsilon > 0$, there exists a set of parameters $w_k$, $k = 1,..,K$, such that -

$$\int_0^\infty \left| w(t) - \sum_{k=1}^{K} w_k g_k(t) \right|^2 dt < \varepsilon.$$

The proof for this theorem is based upon the completeness of the Laguerre polynomials and can be found in Szego (1939, pg. 108, theorem 5.7.1). The foregoing discussion is summarized by Theorem 4.

Theorem 4 *The convolution memory model described by -*

$$net(t) = \int_0^t w(t-s)\, x(s)\, ds, \qquad \text{Eq.28}$$

*is equivalent (w.r.t. $L_2$-norm) to the following system -*

$$net(t) = \sum_{k=1}^{K} w_k x_k(t)$$

*where* $x_0(t) = x(t)$, *and*

$$\frac{dx_k}{dt}(t) = -\mu x_k(t) + \mu x_{k-1}(t),\ k=1,..,K,\ \mu > 0. \qquad \text{Eq.29}$$

*(initial conditions as specified by Eq.27).*

We reserve the term gamma memory to indicate the delay structure described by Eq.29.The recursive nature of the gamma memory computation is illustrated in Figure 4. It is easily checked that the gamma memory structure is stable when $\mu > 0$.

For the discrete time case, we have chosen to approximate the derivative in Eq.29 by the first-order forward difference, that is, -

$$\frac{dx_k}{dt}(t) \approx x_k(t+1) - x_k(t). \qquad \text{Eq.30}$$

Applying Eq.30 leads to the following recurrence relations for the discrete (time) gamma memory -

$$x_0(t) = x(t)$$

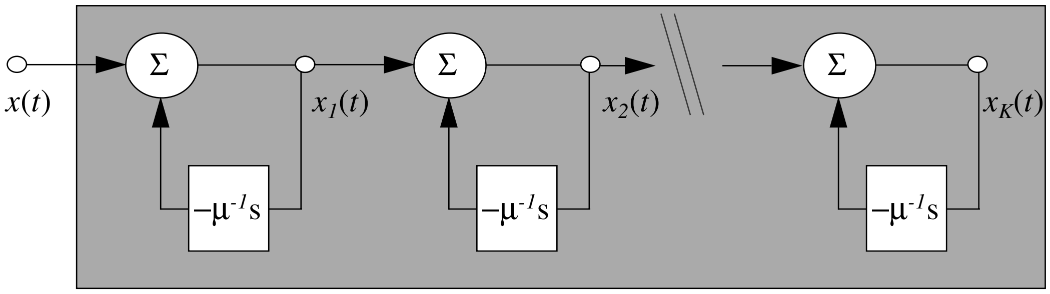

Figure 4 the gamma memory structure. ($s = d/dt$)

$$x_k(t) = (1-\mu)\,x_k(t-1) + \mu x_{k-1}(t-1)\,,\; k = 1,..K. \qquad \text{Eq.31}$$

The time index $t$ now runs through the iteration numbers 0,1,2,...The discrete gamma memory structure is displayed in Figure 5, and can be recognized as a dispersive delay line.

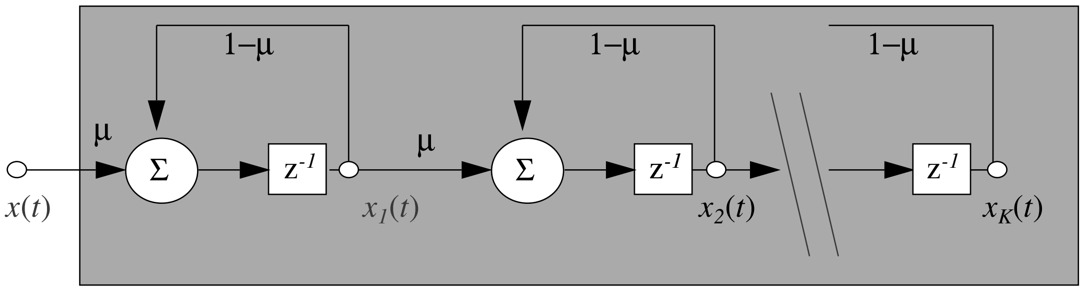

Figure 5 The discrete gamma memory structure.

In Principe and De Vries (1991), the properties of the gamma memory structure are analyzed in detail. The discrete gamma memory is stable for $0 < \mu < 2$. We show that for a $K$-th order structure the memory depth $D$ is well approximated by $D = \frac{K}{\mu}$ and the resolution is $R = \mu$. Thus, for a $K$-th order gamma memory the following important relation characterizes the balance between depth and resolution of the memory -

$$K = D \times R \qquad \text{Eq.32}$$

Since $\mu$ is an adaptive parameter, the memory characteristics can be optimized with respect to a performance index.

We define the gamma neural model as an additive model where the units have the capacity to store their activation history in an adaptive gamma memory structure of (maximal) order $K$ and resolution $\mu_i$. The activation of the $k$th tap of neuron $i$ is written as $x_{ik}(t)$. The weight $w_{ijk}$ connects the $k$th tap of neuron $j$ to neuron $i$. The system equations for the gamma model are given by -

$$\frac{dx_i}{dt}(t) = -a_i x_i(t) + \sigma\left(\sum_j \sum_k w_{ijk} x_{jk}(t)\right) + I_i(t) \qquad \text{Eq.33}$$

for the activations $x_i(t)$ and for the taps $x_{ik}(t)$ we compute -

$$\frac{dx_{ik}}{dt}(t) = -\mu_i x_{ik}(t) + \mu_i x_{i,k-1}(t),\ k = 1,..,K. \qquad \text{Eq.34}$$

In Eq.33 the time constant is processed in the decay parameter $a_i$. Also, for notational convenience we defined $x_i(t) \equiv x_{i0}(t)$ .The structure of the gamma neural model is displayed in Figure 6. Note that as a consequence of Theorem 4, for appropriate memory order *K* the gamma model is equivalent to the convolution model as described by Eq.5.

The following equations apply to the discrete gamma model-

$$x_i(t) = \sigma\left(\sum_{j<i} \sum_k w_{ijk} x_{jk}(t)\right) + I_i(t) \qquad \text{Eq.35}$$

$$x_{ik}(t) = (1-\mu_i) x_{ik}(t-1) + \mu_i x_{i,k-1}(t-1),\ k = 1,..,K \qquad \text{Eq.36}$$

Next, we compare the gamma model to other relevant models for temporal processing.

## 3.2 The gamma model versus the additive neural model

Additive neural models are characterized by the fact that the adaptive net parameters are constants, that is, not dependent on the neural activation nor time dependent. The free parameters in the gamma neural model are $\mu_i$ and $w_{ijk}$, hence the gamma net is an additive model. This is an important result, since the substantial theory on additive neural nets applies directly to the gamma model. Also, existing learning procedures such as Hebbian learning and backpropagation apply without restriction to the gamma net. Since the gamma model is additive, it is possible to express the system equations Eq.33 and Eq.34 as a Grossberg additive model.We have shown this conversion in De Vries and Principe (1991). The result in insightful - the gamma model can be recognized as a prewired additive model. The prewiring of the gamma model for processing of temporal patterns has rendered this model with less free weights than a general additive model. This property is important, since both the learning time and the number of required training patterns grows with the number of free weights in a neural net.

## 3.3 the gamma model versus the convolution model

In section 2.3, we mentioned theoretical analysis, numerical simulation and learning as problem areas for the convolution model. We have seen that the gamma model is mathematically equivalent to the convolution model. We will now re-evaluate these problems when the convolution model is expressed as a gamma model.

Analysis - As was discussed in section 3.2, the gamma model can be represented as a “prewired” additive model. Consequently, theoretical results for the additive model are entirely applicable to the gamma model.

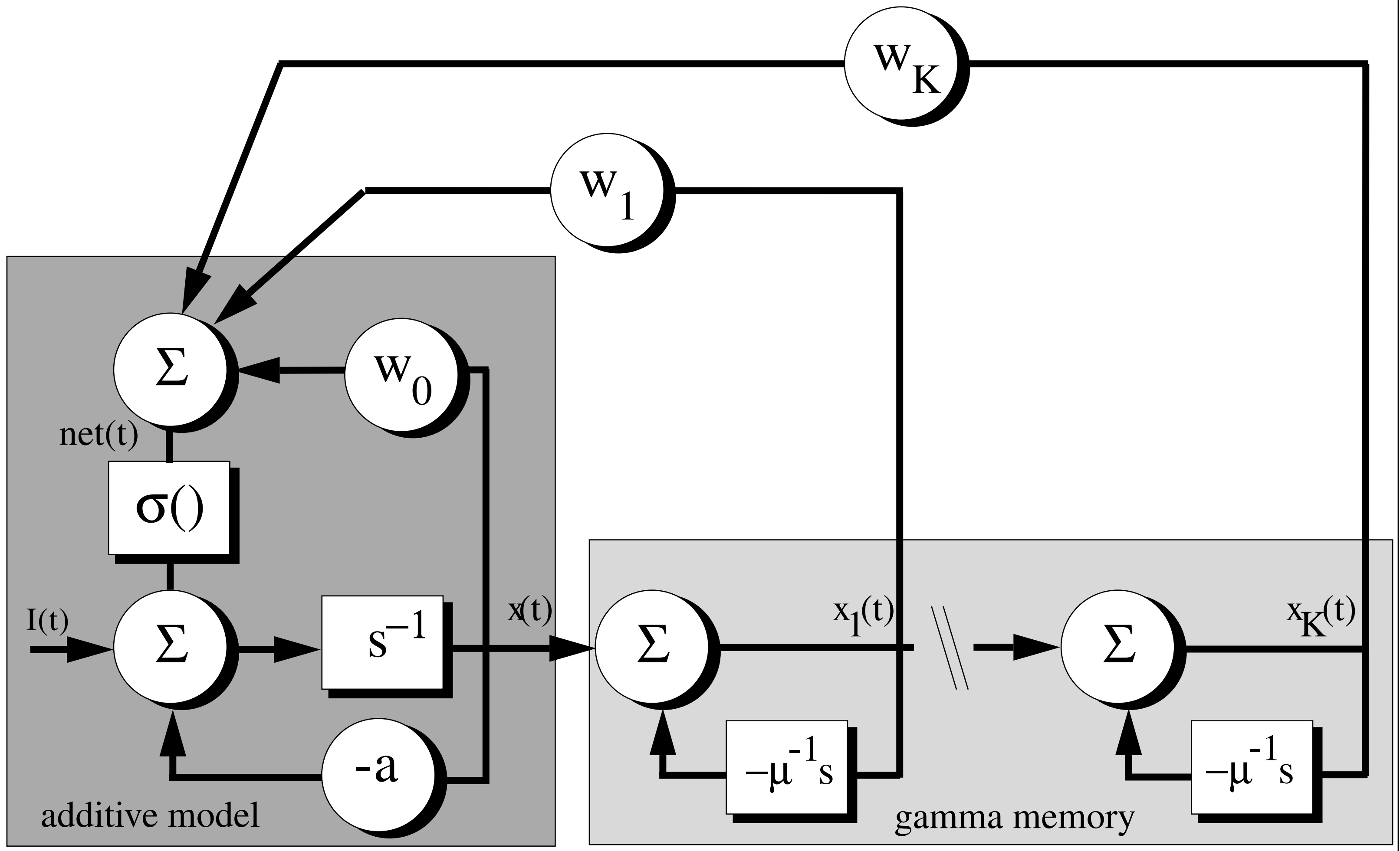

Figure 6 the additive gamma neural model.

Numerical simulation - Whereas the complexity of numerical integration of the convolution model scales as $O(N^2T)$, the gamma model scales as $O(N^2K)$. As an example, consider a 1 sec word sampled at 10 msec intervals. For this case, $T$=100, while the dynamic order $K$ for speech is approximately 12 (Tishby, 1990). Thus, the simulation of the gamma model promises approximately a factor 8 efficiency increase.

Learning - While the weights in the convolution model are time-varying, the gamma weights are constants. Thus, the dimensionality of the weight vector is independent of time. This is a very important distinction from an engineering standpoint, since it allows a direct generalization of conventional learning algorithms to the gamma model.

## 3.4 The gamma memory versus the concentration-in-time net

In 1987, Tank and Hopfield presented an analog neural net with dispersive delay kernels for temporal processing (Tank and Hopfield, 1987). The memory taps $x_k(t)$ for this concentration-in-time net (CITN) are obtained by convolving the input signal with the following kernels -

$$f_k(t) = \left(\frac{t}{k}\right)^{\alpha} e^{\alpha\left(1-\frac{t}{k}\right)}, k = 1,..,K, \qquad \text{Eq.37}$$

where $\alpha$ is a positive integer. $f_k(t)$ is normalized to have maximal value 1 for $t = k$. The degree of dispersion is regulated by parameter $\alpha$. In Figure 7, the kernels $f_k(t)$ are displayed for $k$=1 to 5 when $\alpha = 5$.

Compare this figure to Figure 3. The kernels $f_k(t)$ visually resemble peak-normalized gamma kernels. Yet it is not possible to generate the kernels $f_k(t)$ by a recursive set of ordinary differential equations with constant coefficients, as is the case for the gamma kernels. In fact, differentiating Eq.37 leads to the following time-varying differential equation for $f_k(t)$ -

$$\frac{df_k}{dt}(t) = \alpha\left(\frac{1}{t} - \frac{1}{k}\right) f_k(t) . \qquad \text{Eq.38}$$

While Tank and Hopfield's model shares with the gamma model the capability of regulating temporal dispersion, only the gamma model provides an additive neural mechanism for this capacity. As a result, in the gamma memory the dispersion control parameter $\mu$ can be treated as an adaptive weight. In Tank and Hopfield's model, $\alpha$ is fixed. We have not investigated the relative merits of peak- versus area-normalization of dispersive delay kernels.

Similar arguments hold when we compare the gamma memory to the adaptive gaussian distributed delay models such as the TEMPO 2 model (Bodenhausen and Waibel, 1991) and the gaussian version of the concentration-in-time neural net (Unnikrishnan et al., 1991). These memory models do offer the advantage of adaptive dispersion, yet only the gamma memory offers an additive neural mechanism to create dispersive delays. The other models require evaluation of a convolution integral with respect to the delay kernels in order to compute the memory traces. Although not a priority for engineering applications, the gamma memory is biologically plausible, since there is no (non-neural) external mechanism required to generate delay kernels.

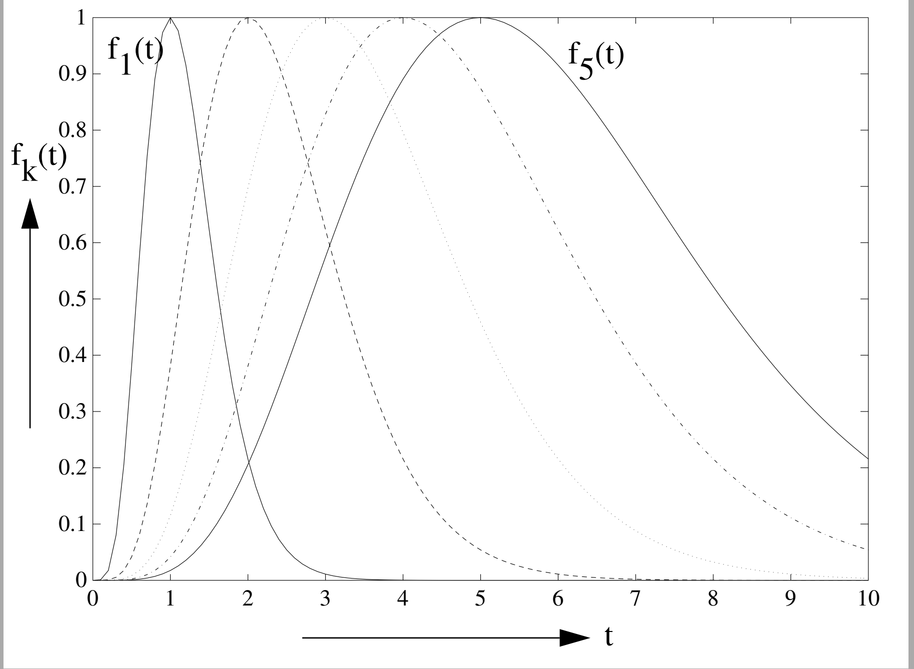

Figure 7 Tank and Hopfield's delay kernels; $\alpha$=5, $k$=1,..,5

## 3.5 the gamma memory versus the time delay neural net

The memory structure in the time delay neural net (TDNN) is a tapped delay line. In fact, TDNN structures can be created in the gamma memory by fixing $\mu = 1$ in the discrete gamma model. Thus, the TDNN is a special case of the discrete gamma model. When $0 < \mu < 1$, the discrete gamma memory implements a tapped dispersive delay line. The amount of dispersion is regulated by the adaptive memory parameter $\mu$. We discussed that the memory depth of the gamma memory can be estimated by $K/\mu$. Hence, the memory depth can be adapted independently from the number of taps (and weights!) in the structure. In the TDNN, the memory depth and the memory order both equal $K$. As a result, increasing the memory depth in the TDNN is always coupled with an increase in the number of weights in the net, which is sometimes not desirable.

# 4 A TOPOLOGICAL EXAMPLE: THE FOCUSED GAMMA NET

## 4.1 introduction

The gamma model equations (Eq.33 and 34) depicted in the block diagram of Figure 6 describe several possible neural network topologies. Here we will select one possible gamma neural network topology, using arguments based on the constraints imposed by learning network weights in a temporal environment. Since the gamma model is an additive model, the gradient descent algorithms that have been designed for these networks can be directly applied to the gamma model. In a temporal environment, the real-time-recurrent-learning (RTRL) procedure (Williams and Zipser, 1989) and backpropagation-through-time (BPTT) (Rumelhart et al., 1986; Werbos, 1990) are applicable to general additive models. Thus, these algorithms can be derived for the gamma net in a straightforward manner. In practice however, learning in a temporal environment entails more than generalizing a set of static equations to handle time. Indeed, for static networks the backpropagation method provides an algorithmic approach to solve a large area of problems previously unreachable due to the amount of computation involved. This advantage is not so obvious when we generalize backpropagation to dynamic networks such as the ones derived from the gamma model. A straightforward application of backpropagation to dynamic networks leads to unfolding in time of the network layers. As a result for this method, the storage requirements grow linearly with time. The alternative method, real-time-recurrent-learning, imposes a constant (in time) load on the computational resources. Yet, since RTRL effectively is a "brute force" method of gradient evaluation the computational requirements per time step are large. Thus, application of RTRL is restricted to small networks.

One way to speed up error gradient computation is to prewire the network architecture in order to reduce the complexity of the learning algorithm (Williams, 1991). A particular interesting example in this matter is the focused backpropagation architecture, introduced by Mozer (1989). Comparable structures were used by Stornetta et al. (1987) and Gori et al. (1989). In the focused backpropagation net, context-unit memory elements are restricted to the first layer. Furthermore, the input layer representation of the past is fed to a strictly static feedforward net only. The architectural restrictions on the focused net allow for very efficient error gradient computation. Also, the error estimates in the dynamic input layer do not disperse during training since the context units do not have lateral connections in the input layer. Thus the error estimates do not disperse in the focused net, which explains the adjective "focused".

Application of the focused backpropagation architectural principle to the gamma model leads to an interesting structure which we call the focused gamma net. All beneficial properties of the focused backpropagation net carry over. In fact, the focused gamma net topology was implicitly assumed in the comparisons with some of the other neural networks that were previously discussed in this paper. For $\mu = 1$, the focused gamma net reduces to a time-delay-neural-net where the memory bank is restricted to the input layer. When $K = 1$, the focused gamma net reduces to the first-order context-unit models as described by Mozer (1989) and Stornetta et al. (1987). Note that a linear one-layer focused gamma net generalizes Widrow's adaline structure (Principe and De Vries, 1991).

Next we introduce the architecture of the focused gamma net, followed by a derivation of the error gradients for learning.

## 4.2 architecture

The focused gamma net is schematically drawn in Figure 8. Assume the $N_I$-dimensional input signal $I(t)$. The past of this signal is represented in a gamma memory structure as described by the following relations -

$$x_i(t) = I_i(t) \qquad \text{Eq.39}$$

$$x_{ik}(t) = (1-\mu_i)x_{ik}(t-1) + \mu_i x_{i,k-1}(t-1) \qquad \text{Eq.40}$$

where $t = 0,1,2...$, $i = 1,..,N_i$ and $k = 1,..,K$. This layer, the input layer, has $N_i$ memory parameters $\mu_i$. The activations in the input layer are mapped onto a set of output nodes by way of a (non-linear) static strictly feedforward net. The nodes in the feedforward net are indexed $N_i$+1 through $N$. The activations in the static net are computed by -

$$x_i(t) = \sigma\left(\sum_{j<i}\sum_k w_{ijk}x_{jk}(t)\right), \qquad \text{Eq.41}$$

where we have utilized the notation $x_{i0}(t) \equiv x_i(t)$ and $w_{ij0} \equiv w_{ij}$.

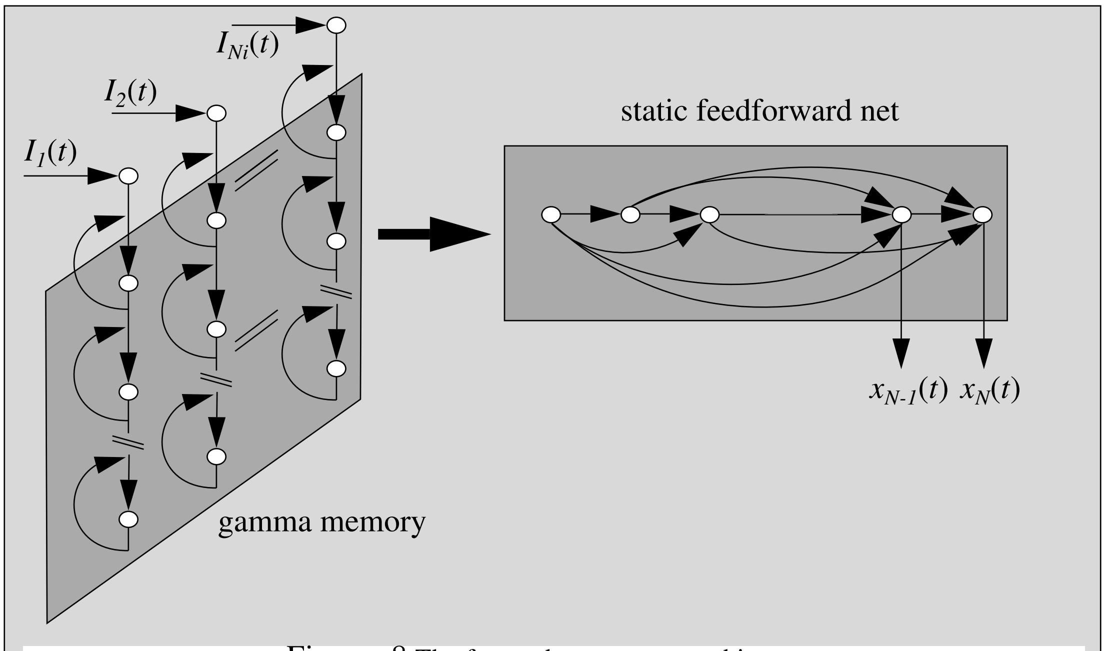

Figure 8 The focused gamma net architecture

## 4.3 learning equations

Let us first derive the update equations for the weights $w_{ijk}$, using the backpropagation method. We assume a quadratic performance index described by -

$$E = \frac{1}{2}\sum_t\sum_i [e_i(t)]^2 = \frac{1}{2}\sum_t\sum_i [d_i(t) - x_i(t)]^2, \qquad \text{Eq.42}$$

where $d_i(t)$ is a target signal and $e_i(t)$ the (instantaneous) error signal. The derivation of the error gradients in static nets has been discussed in many publications (e.g. Rumelhart et al., 1985). Application of the backpropagation technique to the feedforward net leads to the following backpropagation system -

$$\varepsilon_i(t) = -e_i(t) + \sum_{j>i} w_{ji}\delta_j(t) \qquad \text{Eq.43}$$

$$\delta_i(t) = \sigma_i'(net_i(t))\,\varepsilon_i(t) \qquad \text{Eq.44}$$

where we defined $\varepsilon_i(t) \equiv \frac{\partial E}{\partial x_i(t)}$ and $\delta_i(t) \equiv \frac{\partial E}{\partial net_i(t)}$. The error gradients $\frac{\partial E}{\partial w_{ijk}}$ can be computed by -

$$\frac{\partial E}{\partial w_{ijk}} = \sum_t \delta_i(t)\,x_{jk}(t) \qquad \text{Eq.45}$$

Note that since the mapping network is static and feedforward there is no need to backpropagate through time in order to find the backprop errors $\delta_i(t)$. In fact, since the $\delta_i(t)$ 's are computed in real-time, it is easy to convert Eq.45 into a real-time procedure by defining -

$$\frac{\partial E}{\partial w_{ijk}}(t) \equiv \delta_i(t)\,x_{jk}(t) \qquad \text{Eq.46}$$

Application of backpropagation to compute the error gradients with respect to the parameters $\mu_i$ leads to a backpropagation-through-time procedure, since the input layer is recurrent in nature. It is expected that in most gamma networks the number of memory parameters is relatively small so it is efficient to use the RTRL method here. Thus, the error gradients $\frac{\partial E}{\partial \mu_i}$ can be evaluated as follows -

$$\frac{\partial E}{\partial \mu_i}(t) = \sum_m \frac{\partial E}{x_m(t)} \times \frac{\partial x_m(t)}{\partial x_{ik}(t)} \times \frac{\partial x_{ik}(t)}{\partial \mu_i}$$

$$= -\sum_m e_m(t)\,\sigma_m'(net_m(t)) \sum_k w_{mik}\alpha_i^k(t) \qquad \text{Eq.47}$$

where $\alpha_i^k(t) \equiv \frac{\partial x_{ik}(t)}{\partial \mu_i}$. $\alpha_i^k(t)$ can be computed by differentiation of Eq.40, yielding -

$$\alpha_i^0(t) = 0$$

$$\alpha_i^k(t) = (1-\mu_i)\,\alpha_i^k(t-1) + \mu_i\alpha_i^{k-1}(t-1) + \mu_i[x_{i,k-1}(t-1) - x_{ik}(t-1)]\,. \qquad \text{Eq.48}$$

In this architecture we have taken advantage of the particular characteristics of both the BPTT and RTRL adaptive procedures. Since the feedforward net is static, we use the very efficient backpropagation procedure to update the weights $w_{ijk}$. The input layer of the focused gamma net is dynamic however and as a result, application of backpropagation would introduce the burden of time-dependent storage requirements during the backward pass. RTRL on the other hand is a real-time procedure that is tailored to application in small dynamic networks. Thus we propose RTRL in the recurrent input layer to compute the error gradients to the memory parameters $\mu_i$.

## 5 CONCLUSIONS

In a historical perspective, our goal was to develop a neural model for processing of temporal patterns that avoid the segmentation problem. The gamma model utilizes an adaptive short term memory mechanism that obviates a priori signal segmentation. In a sense, while the temporal structure of the memory in discrete-delay neural models is fixed by the sampling rate, the gamma model allows adaptation of its internal time constants so as to match the temporal structure of the short-term memory to the input signal space. We showed that the gamma memory structure is closely related to state-of-the-art models such as the time-delay-neural-net, the concentration-in-time-neural-net and the context-unit-memory models. In fact, the gamma memory configuration seems to provide an interesting medium between these models.

The crux of this exposition is centered in the equivalence between convolution and ARMA models (Theorem 1), interpreted as dynamical models for memory. Unfortunately, adaptive ARMA models are very difficult to work in practice due to stability constraints. The gamma neural model appears as a new approximation to the convolution model (closed in $L_2$) that can be formulated as a dynamic additive model ( Eq.2). The theory (stability, convergence) and learning procedures for the additive model are far more developed than for the convolution model.

On the practical side, the choice of the gamma kernels implies a particular choice of basis functions of time. Apart from the properties that we have mentioned before (Theorem 4), the usefulness of such a basis is reflected in how well large temporal vector spaces can be approximated by a reduced number of gamma functions. As a particular example, how well do the gamma kernels compare to, for instance, the delta functions in representing electroencephalogram (EEG) signals, known to be very complex time signals?

In order to test this problem, the following experiment was conducted. We selected an EEG data segment from sleep stage two ( Figure 9a). The processing system was a two-layer linear focused gamma network with five hidden units. The processing goal was to predict the EEG signal by five samples. The performance index, defined as $E = \frac{var[e^2(t)]}{var[d^2(t)]}$, is shown after convergence as a function of $\mu$ in Figure 9b. The system was trained by the backpropagation method as described in section 4.3. Apparently, the performance of the second memory order net for $\mu = 0.6$ is comparable to the fourth order net with a tapped delay line ($\mu = 1$). The system with second order memory is to be preferred, since the number of weights is less for this structure than for the fourth order memory model.

We have experimented with several signals (sinusoids in noise, Feigenbaum map, electroencephalogram (EEG)) for various processing protocols (prediction, system identification, classification). Invariably the optimal memory structure[1] was obtained for $\mu < 1$. These data will be reported in a forthcoming publication.

1. The optimal memory structure is defined as the structure of lowest dimensionality that minimizes the performance index E.

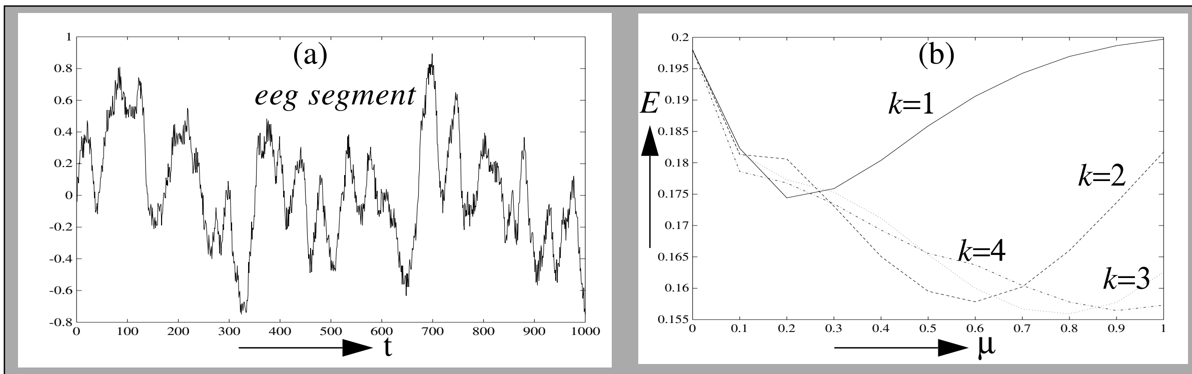

Figure 9 eeg prediction results for a two-layer linear gamma model. Experimental parameters: 1 input unit, K gamma memory taps, 5 hidden units, 1 ouput unit; 5 prediction steps . (a) the eeg segment was taken from sleep stage two. (b) normalized performance index after convergence as a function of $\mu$.

Our current research concentrates on two aspects. We have shown that learning rules for dynamic nets are easily generalized to the focused gamma net. We are investigating how the gamma net compares with TDNN in a speech recognition problem (the E set). In another project, we are developing a multi-layer gamma net for real-time isolated word recognition. In this model, both layer-to-layer weights and memory decay parameters $\mu$ are adaptive. Thus we are able to create successive layers with decreasing resolution (and increasing depth of memory), very much like a temporal analog of the neocognitron (Fukushima, 1980).

It is worth mentioning that the focused gamma net is only one example of a variety of gamma model implementations. The leading discriminant feature of the gamma model is the special way the input signal past is stored in the neural structure, which we called the gamma memory. Therefore, any of the additive static neural network implementations such as the Hopfield net or feedfoward nets can utilize the gamma memory. Of particular theoretical interest are feedforward implementations utilizing the gamma memory in the hidden layer, which do not have counterpart in traditional system theory.

## ACKNOWLEDGMENTS

This work has been partially supported by NSF grant ECS-8915218 and DDM-8914084. The clarity of this paper substantially improved after the comments on a first submission by two anonymous reviewers, which we gratefully acknowledge.

## REFERENCES

Bodenhausen U. and Waibel A., Learning the Architecture of Neural Networks for Speech Recognition, IEEE Proc. of the ICASSP, May 1991.

Braun M., *Ordinary Differential Equations and their Applications*, 3rd ed., Springer-Verlag, Berlin, 1983.

De Vries B. and Principe J.C., A Theory for Neural Nets with Time Delays, *NIPS90 processings*, Lippmann R., Moody J., and Touretzky D. (eds.), San Mateo, CA, Morgan Kaufmann, 1991.

Elman J.L., Finding structure in time. *Cognitive Science 14,* pp. 179-211, 1990.

Farque M.D., Reductibilite des systemes hereditaires a des systemes dynamiques (regis par des equations differentielles ou aux derivees partielles). *C.R. Acad. Sc. Paris*, Serie B, pp. 471-473, 1973.

Fukushima K., Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition by Shift in Position. *Biological Cybernetics* 36, pp. 193-202, 1980.

Gori M., Bengio Y. and De Mori R., BPS: A Learning Algorithm for Capturing the Dynamic Nature of Speech, in Proc. of the Int. Conf. on Neural Networks, vol. 2, pp. 417-413, 1989.

Grossberg S., *Studies of Mind and Brain*. Reidel, Boston, MA, 1982.

Jordan M.I., Atractor dynamics and parallelism in a connectionist sequential machine. *Proc. 8th annual conf. on Cognitive Science* Society, pp. 531-546, 1986.

Hertz J., Krogh A. and Palmer R.G., *Introduction to the Theory of Neural Computation*, Addison-Wesley, Reading, MA, 1991.

Lang K., Waibel A.H. and Hinton G.E., A Time-Delay Neural Network Architecture for Isolated Word Recognition. *Neural networks*, vol. 3. (1), pp. 23-44, 1990.

Miller R., Representation of brief temporal patterns, Hebbian synapses, and the left-hemisphere dominance for phoneme recognition. in *Psychobiology, vol. 15 (3)*, pp. 241-247, 1987.

Mozer M.C., A Focused Backpropagation Algorithm for Temporal Pattern Recognition, *Complex Systems* 3, pp. 349-381, 1989.

Principe J.C. and De Vries B., The Gamma Filter - A New Class of Adaptive IIR Filters with Restricted Feedback, subm. to *IEEE Transactions on Signal Processing*, 1991.

Rabiner L.R. and Schafer R.W., *Digital Processing of Speech Signals*, Prentice-Hall, Englewood Cliffs, NJ, 1978.

Perugini N.K. and Engeler W.E. , Neural Network Learning Time: Effects of Network and Traning size. IJCNN, II-395, 1989.

Rumelhart D.E., Hinton G.E. and Williams R.J., Learning Internal Representations by Error Back-propagation, in *Parallel Distributed Processing*, vol. 1, ch. 8, Rumelhart D.E., McClelland J.L. (eds.), MIT Press, 1986.

Sejnowski T.J., Skeleton Filters in the Brain. in *Parallel model of Associative memory*, pg. 189-212, Hinton and Anderson (eds.), LEA Inc., 1981.

Shen L., Neural Integration by Short Term Potentiation. *Biological Cybernetics 61*, pp. 319-325, 1989.

Stornetta W.S., Hogg T. and Huberman B.A., A Dynamical Approach to Temporal Pattern Processing, in *Neural Information Processing Systems*, Anderson D.Z. (ed.), pp. 750-759, 1988.

Szego G., *Orthogonal Polynomials*, Colloquium Publications, vol. 23, American Mathematical Society, Providence, RI, 1939.

Tank D.W. and Hopfield J.J., Concentrating information in time: analog neural networks with applications to speech recognition problems. in *IEEE 1st international conference on neural networks,* vol. IV, pp. 45-468, June 1987.

Tishby N., A Dynamical Systems approach to Speech Processing. *Proc. ICASSP,* pp. 365-368, 1990.

Unnikrishnan K.P., Hopfield J.J. and Tank D.W., Connected-Digit Speaker-Dependent Speech Recognition using a Neural Network with Time-Delayed Connections, *IEEE Transactions on Signal Processing*, vol. 39, no. 3, pp. 698-713, March 1991.

Werbos P.J., Backpropagation Trough Time: What It Does and How to Do It. *Proc. of the IEEE*, vol. 78, no. 10, pp. 1550-1560, Oct. 1990.

Williams R.J. and Zipser D., A Learning Algorithm for Continually Running Fully Connected Networks, *Neural Computation* 1, pp. 270-280, 1989.

Williams R.J., Adaptive state representation and estimation using recurrent connectionist networks, in Neural Networks for Control, Ed. Miller, Sutton, Werbos, pp. 97-114, MIT Press 1990.