# Apoptosis/Neurogenesis Favorably Informs Memory Development

**Willard L. Miranker**
**Department of Computer Science**
**Yale University**
**TR1284**

# Apoptosis/Neurogenesis Favorably Informs Memory Development

**Willard L. Miranker**[1]
**Department of Computer Science**
**Yale University**

**Abstract-** The functional significance of alternate forms of plasticity in brain (such as apoptosis/neurogenesis) is not easily observable with biological methods. Employing Hebbian dynamics for synaptic weight development, a three-layer neural network model of the hippocampus is used to simulate non-supervised learning. This learning, implemented by means of the encoding of memory traces, is applied to the characters of a pair of related alphabets, first the Roman and then the Greek. These encodings are endogenously developed by the network. A U-shaped performance curve, resulting from a tradeoff in the dentate gyrus, the model's middle layer (namely, forgetting induced by apoptosis of neurons versus enhancement of learning induced by neurogenesis), shows that apoptosis/neurogenesis in hippocampus favorably informs memory development.

## 1. INTRODUCTION

Adult neurogenesis occurs in man as well as other mammalian species (Gould et al, 2000; Makakis and Gage, 1999). This phenomenon appears most robustly in certain brain regions, particularly the dentate gyrus (DG) of the hippocampus and in the olfactory system (Erikson et al, 1998; Kornack and Rakic, 2001). The functional significance of neuronal plasticity (such as apoptosis/neurogenesis) is not easily observable with biological methods (Aakerlund and Hemmingsen, 1998). So neural network simulations provide a salient procedure for direct analysis of learning and memory properties of plasticity in neural systems.

In Chambers et al, 2004, the possibility that replacement neurons could favorably impact cognition as well as a variety of other brain function informed by hippocampal activity (short and long term memory formation, adaptations to sex and stress hormones, various forms of mental illness...) was studied by computer simulation. That simulation modeled learning tasks employing a three layer neural net model of hippocampus, those layers, modeling in turn, the entorhinal cortex, the dentate gyrus, and CA3. The network was made to learn a representation of the Roman alphabet, the first task. Upon completion of this task, the network was made to learn a representation of the similar but not identical Greek alphabet, the second task. The postulate that neurogenesis favorably influences the second task was demonstrated.

---

[1] willard.miranker@yale.edu

Information in brain (memory traces...) is taken to be recorded in a distributed manner in the synapses of the relevant neuronal assemblies. It is adjustments to the strength of these many synaptic connections that are taken to be the recording mechanism. This synaptic adjustment proceeds by means of a brain learning dynamics that must be simulated as part of the model. In the previous study by Chambers et al, the learning dynamics chosen was the back-propagation algorithm, a method of learning (of the so-called supervised type) commonly employed in neural net simulations (Haykin, 2000). Here we shall invoke Hebbian learning, an unsupervised form of neural net learning dynamics that is used in models of brain circuitry. We shall show that neurogenesis favorably informs learning (i.e., memory trace formation) in the more realistic modeling context of Hebbian learning.

The use of an unsupervised form of learning requires development of an intrinsic representation (an endogenous encoding) of the memory traces. That is, in place of guiding the model's output to take on an extrinsically (and arbitrarily) specified encoding of the information to be recorded (as with the supervised learning protocol of back-propagation), we take as a critical aspect of the learning ability, the capacity of the model to implement an intrinsic and autonomous method for encoding of the information being presented. That a neural system has the functionality to do this is a key and novel feature of the present approach to the study of neurogenesis, and one that we believe accurately models memory establishment in the brain.

## 2. THE MODEL

### 2.1 The Neural Net Architecture, I-O Dynamics, Learning Dynamics

**The neural circuit**: The hippocampus is comprised of three layers. The first layer, the entorhinal cortex is connected by the perforant path to the second layer, the dentate gyrus. The latter in turn is connected by the mossy fibers to CA3, the third layer. In the model, we shall refer to these as layer $k$ ($k = 1,2,3$, respectively). The $k$-th layer shall have a number, $N_k$ of neurons. The perforant path and the mossy fibers are simulated by forward excitatory (synaptic) connections between the model's layers. The model also includes lateral inhibitory connections within layers 2 and 3. (See Figure 2.1.) Each synaptic connection is characterized by an associated weight $w_{ij}^{kl}$, a real valued parameter, where the indices denote a connection from neuron $j$ in layer $l$ to neuron $i$ in layer $k$. Of course only the superscript pairs 12 and 23 (for forward connections) and 22 and 33 (for lateral connections) are relevant, and a synaptic weight corresponding to any others indices that may appear is taken to be zero. We suppose that when a neuron is connected to another neuron that it is connected to all of the neurons in the latter's layer. We shall describe such an arrangement as being a fully connected one. Setting an associated synaptic weight to zero accommodates missing connections.

**Input/output dynamics**: Layer 1 is an input layer to the neural net and the output of layer 3 is its output. The neuronal input/output dynamics are defined as follows. Let $y_j^l$

denote the output of neuron $j$ ($j = 1,\cdots,N_l$) in layer $l$ ($l = 1,2,3$). Then $v_i^k$, the total weighted input to neuron $i$ in layer $k$ ($k = 2,3$) is specified as follows.

2.1) $$v_i^k = \sum_{j=1}^{N_l} w_{ij}^{k,k-1} y_j^{k-1} + \sum_{j=1}^{N_k} w_{ij}^{kk} y_j^k .$$

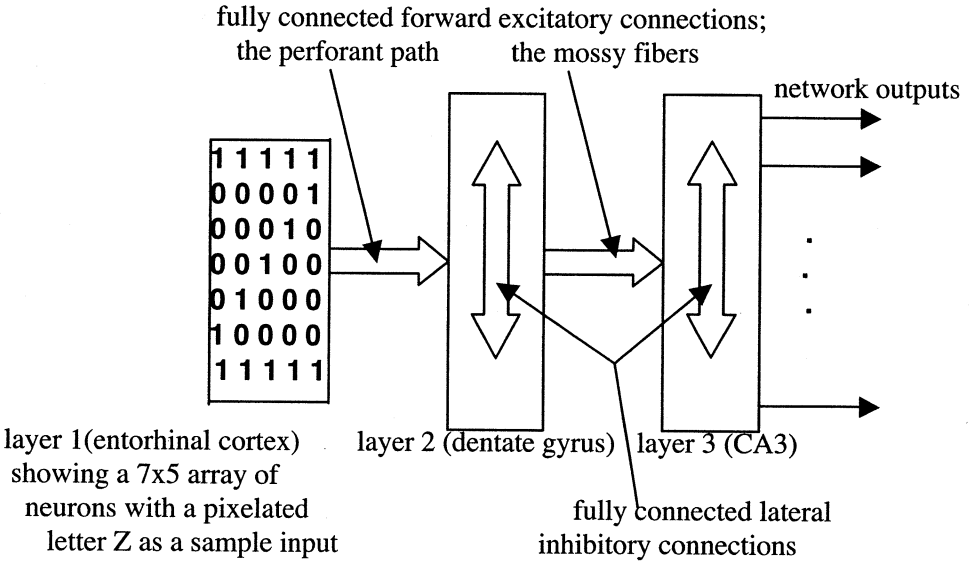


**Figure 2.1:** Schematic of the neural network. Block arrows indicate a fully connected set of connections, inter- or intra-layer, as the case may be.

The neurons are taken to be McCulloch-Pitts neurons[2]. This implies that for $k = 2,3$, the output $y_j^k$ is

2.2) $$y_j^k = \begin{cases} 1, & v_i^k \geq \theta^k \\ 0, & v_i^k < \theta^k, \end{cases}$$

where $\theta^k$ is the neuronal firing threshold.

**Input/output sequencing**: Each exogenous input (an alphabetic character) is presented at layer 1. (See Figure 2.1.) Then the outputs of all neurons in layers 2 and 3 are specified using (2.1) and (2.2). These neurons are taken to fire in the following order.

---

[2] McCulloch-Pitts neurons are among the most basic model neurons. Since we are able to demonstrate the relevant apoptosis/neurogenesis effects of interest with these neurons, the use of a more complex model neuron is not called for.

2.3) $$y_1^2, y_2^2, \cdots, y_{N_2}^2, y_1^3, y_2^3, \cdots, y_{N_3}^3 .$$

So layer 1 fires first and then layer 2 fires, and as indicated in (2.3), the neurons fire in sequence within each layer as well. The numbering of the neurons within a layer is chosen arbitrarily[3].

**Learning**: Synaptic weights are initialized randomly, and they change according to *Hebb's law*, that is, according to correlation between input at a synapse and subsequent firing/not-firing of that synapse's neuron. This law is implemented by the following formula.

2.4) $$\Delta w_{ij}^{kl} = \tau\left(a_0^{kl} y_i^k y_j^l + a_1^{kl} y_i^k + a_2^{kl} y_j^l\right),$$

where $\tau$ is the learning rate[4]. The coefficients $a_j^{kl}$ appearing in (2.4) are chosen so that the computed weight change is consistent with the correlations (i.e., with Hebb's law) that arise during the learning stage. A neuron's weights are updated immediately upon that neuron's firing.

**Clock cycle**: The learning algorithm proceeds with a clock timing that we shall index with $n$, say. To indicate that a variable changes with clock cycle, this time index $n$ will be appended to that variable accordingly. A tick (an advance) of the clock is specified in the Section 2.2.

## 2.2 The Learning Tasks

**The alphabets and their representation**: The neural net is to learn the upper case characters of two different alphabets, the Roman and the Greek. These are represented in pixelated form shown in Figure 2.1. Note that these two alphabets have 14 character symbols in common.

**The internal encoding**: As a first task, the neural net (neural circuit) is to learn the characters of the Roman alphabet, and to accomplish this task, the net is repeatedly presented with representations of those characters lexicographically. Referring to Figure 2.1, we see that layer 1 is schematized as a pixelated retina. The characters are presented in pixilated form on that retina, and so, this representation defines the successive inputs $y^1(n), n = 1,2,\cdots$ as binary vectors. (Note that this assignment of the time index implies that the clock ticks once after the last neuron in layer 3 fires and the latter's weights are updated.) The net creates an *evolving* binary encoding of each character as the latter is inputted. This encoding is defined as the corresponding vector of network output

---

[3] The averaging of many randomly initiated runs, characteristic of our approach to the simulation, accommodates our fixing of a single ordering choice for all of the neurons within a layer; a choice made for reasons of simplicity, causing no loss of generality.

[4] Use of this basic choice of learning rule is validated for reasons analogous to those cited in footnote 1.

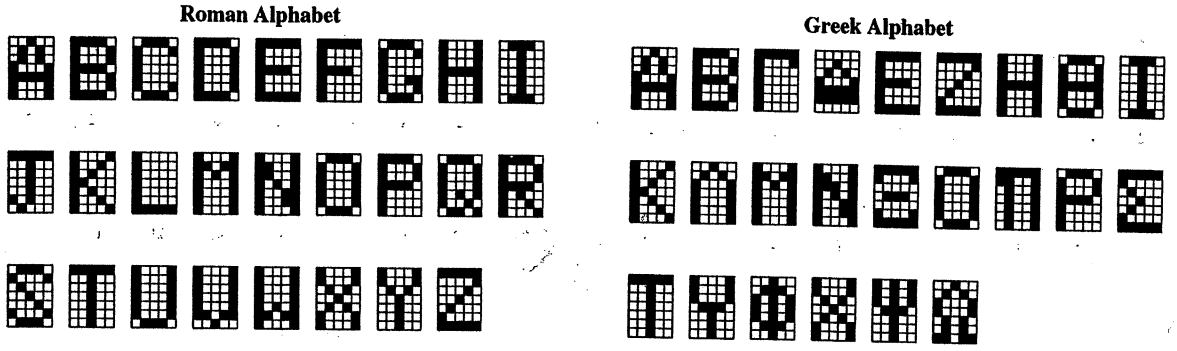


**Figure 2.1:** Pixelated character representations of the Roman and Greek alphabets

$y^3(n), n = 1,2,\cdots$, and so, a character's encoding is a vertex of the unit binary $2^{N_3}$-cube. The alphabet itself is encoded by a collection of such vertices. We take the net to have learned the alphabet (ceasing thereby the learning presentations) if the following two conditions are met.

(i) The output encoding the alphabet is a collection of $M$ (where $M$ is the alphabet size) vertices, that is, each character corresponds to a unique vertex,

(ii) The encoding in (i) is repeated exactly without exception during presentation of the entire alphabet an agreed upon number, say $R > 0$ of times.

**The learning task change**: After the net has learned the Roman alphabet, the task is switched to learning the Greek alphabet. The learning of the Greek alphabet proceeds in the earlier manner, as for the Roman.

### 2.3 Modeling Apoptosis and Neurogenisis

The learning of the Greek is re-implemented a number of times. Each time the neural net (the collection of its synaptic weights) is first restored to its precise state at the completion of the Roman learning task. Next, one or more neurons in layer 2 (the dentate gyrus) are chosen to be replaced by nascent ones whose weights are randomly initialized. So apoptosis followed by neurogenesis is simulated in this manner. With this preparation, the learning of the Greek is restarted. Appealing to a process of cyto-toxicity, we replace those neurons with the largest input weights[5]. To implement this, let $w_i^2,\ i = 1,\cdots,N_2$ denote the vector of synaptic weights corresponding to forward

[5] Those neurons with largest weights will tend to work hardest and age the fastest, and so, would seem to have a biological need to be replaced.

connections into neuron $i$ in layer 2, and let $w_i^{22}$ be the analogous vector corresponding to lateral connections within layer 2 into that neuron. Then we compute

2.5) $$T_i = \left\| w_i^2 \right\| + \left\| w_i^{22} \right\|.$$

Here $\| z \|$ denotes the Euclidean norm of a vector $z$. $T_i$ is taken as a measure of the cyto-toxicity of the corresponding neuron. (See footnote 4.) Now when it is decided to replace $r$ neurons (apoptosis followed by neurogenesis), those with the $r$ largest values of $T_i$ are chosen.

## 3. THE SIMULATION

### 3.1 Simulation Protocols and Parameter Values

**Layer sizes**: We take $N_1 = 35$, and this corresponds to a $7 \times 5$ input retina. $N_2$ and $N_3$ are taken to vary over collections of different values. In particular, $N_2 \in \{16,20,24,28,32\}$ and $N_3 \in \{11,12,13,14\}$.

**Synaptic weights, initial values, floor and ceiling**: The magnitudes of the initial values of synaptic weights are chosen randomly from a specified interval $I$, with the forward excitatory weights positive and the lateral inhibitory weights negative. The weights develop according to (2.4), the learning formula, but they are not allowed to change sign nor are their magnitudes permitted to exceed a ceiling $C$. Specifically, we take $I =$ [0, 0.1] and $C = 0.125$. If a computed weight change would cause the value of that weight to exit the interval $[0,C]$ on the left/right, its actual value is truncated and taken to be the floor/ceiling value $0/C$.

**Learning rate adjustment**: The displacements calculated by iterative systems, such as the dynamical systems ((2.1)-(2.2) and (2.4)) in our simulation, change as the learning progresses, varying typically trending smaller as convergence is approached. Then it is useful to vary the learning rate $\tau$, decreasing/increasing it, thereby stimulating the displacements taken to trend smaller/larger more responsively[6]. Among the many ways to install such a feature, the following autonomous choice was taken. Specifically the learning rate $\tau$ is varied according to the following rule.

2.7) $$\tau_{n+1} = \tau_n \frac{\left\| y^3(n) - y^3(n-1) \right\|_H + 1}{\left\| y^3(n-1) - y^3(n-2) \right\|_H + 1}.$$

---

[6] Global convergence of iterative dynamical systems is accelerated when the displacements they compute, usually trending larger-to-smaller are exogenously exaggerated through correlated variations in the learning rate. This commonly used algorithmic feature is sometimes referred to as over/under relaxation.

Here $\| y^3 \|_H$ denotes the Hamming norm of $y^3 = (y_1^3, \cdots, y_{N_3}^3)$, the binary valued vector of layer 3 outputs, that is, the net's output vector.

**Hebb rule parameters**: For the forward connections from layer $l$ to layer $k$ (i.e., for $(k,l) = (2,1)$ and $(3,2)$), we take $a_0^{kl} = 1.5$, $a_1^{kl} = -0.5$, and $a_2^{kl} = -0.5$. For the lateral connections (i.e., for $(k,l) = (2,2)$ and $(3,3)$), we take $a_0^{kl} = -1.5$, $a_1^{kl} = 0.5$, and $a_2^{kl} = 0.5$. These choices may be seen to accommodate the correlation requirements of Hebb's rule.

**Threshold parameters**: The specific choices of thresholds are $\theta^2 = \theta^3 = 0.1$.

**Learning epoch, repeat parameter, and learning remission factor**: The process of displaying the characters of an entire alphabet in lexicographical order on the input retina (each character display followed by the specified neural firings and weight updates associated with that display) is called a learning epoch[7]. The number of such epochs allowed in a training run is limited arbitrarily to 400. The value of the repeat number (for encodings) is set arbitrarily to $R = 2$. We don't expect learning always to be perfect (complete) in a reasonable number of epochs, and so, we also introduce a learning remission factor denoted by $f$. That is, we specify a fraction $f$ of the alphabet that, if learned, is considered adequate for purposes of learning. Values for $f$ are chosen from $\{0.8, 0.85, 0.9\}$.

## 3.2 Results of the Simulation, Plots

A particular simulation case corresponds to a triplet $(f, N_2, N_3)$. ($60 = 3 \times 5 \times 4$ cases in all.) Each case was run 20 times with newly chosen random starting weights each time. So there were 1200 runs in all. So any particular one of these runs is indexed by the symbol[8] $(f, N_2, N_3)(r), r = 1,\ldots,20$. The results are presented in 4 plots. These are 3 plots corresponding to $(f, \overline{N}_2, \overline{N}_3)(\bar{r})$, where the upper bars represent averaging over all values of the bar-ed symbol, and one plot corresponding to $(\bar{f}, \overline{N}_2, \overline{N}_3)(\bar{r})$. So the first 3 of these plots represents an average over 400 runs each, while the last represents an average over 1200 runs. The plots, each containing three curves, are given in Figure 3.1-3 and 3.4, respectively. The three curves correspond respectively to the three choices of $f$, namely $\{0.8, 0.85, 0.9\}$. These are denoted respectively (i) with diamond tic marks, (ii) with square tic marks, and (iii) with triangular tic marks

**Description of the plots**: The abscissas of the plots are the different learning tasks. The first of these marked Roman is the task to learn the Roman alphabet. All of the

---

[7] Since there are $M$=24/26 characters in the Greek/Roman alphabets, a learning epoch takes 24/26 clock ticks.

[8] The symbol of other parameters (e.g., the $a$'s, the $\theta$'s, etc.) associated with a run is not displayed as an index, since those parameters are fixed in value.

succeeding ones, marked Post $g$, where $g = 0,1,2,4,5,12,16$ successively, denote the task of learning the Greek alphabet with $g$ neurons replaced[9] in layer 2.

Figure 3.1 is a plot of the convergence time in epochs needed for the learning of an alphabet to occur. If learning in a run does not occur before the limit of 400 epochs is reached, then 400 is taken as a default value of the number of learning epochs required for that run.

Figure 3.2 is a plot of the fraction of runs (out of 20) that complete the learning before reaching the 400-epoch limit. (Recall that completion of learning means the invariance of the intrinsic encoding achieves the repetition requirement (R=3) subject to the remission factor $f$.)

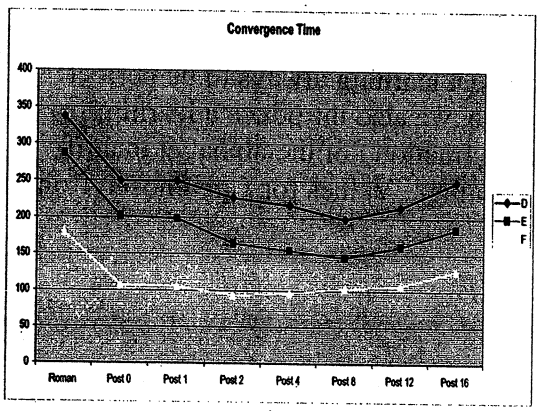


**Fig 3.1:** Plot for $(0.8,\overline{N}_2,\overline{N}_3)(\bar{r})$

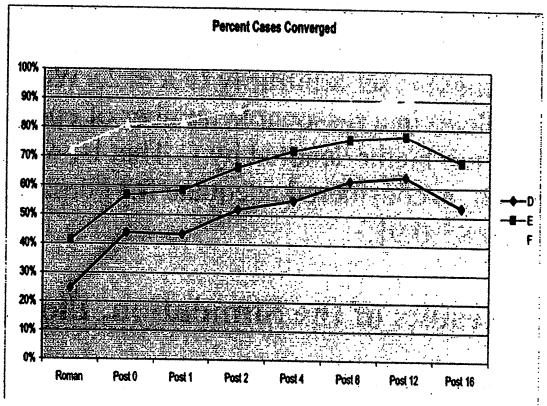


**Fig 3.2:** Plot for $(0.85,\overline{N}_2,\overline{N}_3)(\bar{r})$

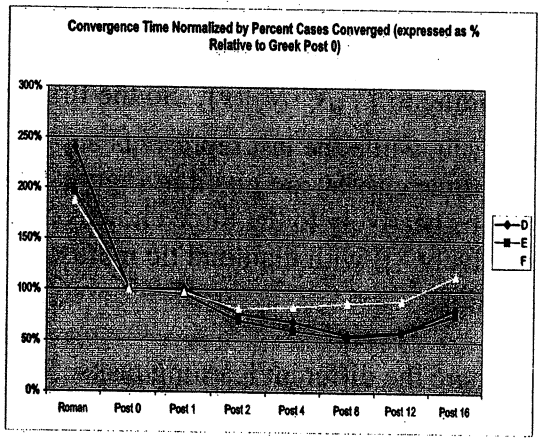


**Fig 3.3:** Plot for $(0.9,\overline{N}_2,\overline{N}_3)(\bar{r})$

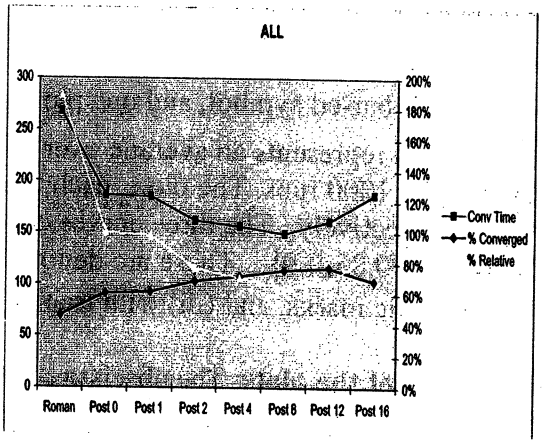


**Fig 3.4:** Plot for $(\bar{f},\overline{N}_2,\overline{N}_3)(\bar{r})$

[9] Replaced, meaning as customary, the apoptosis of $g$ neurons followed by their replacement through neurogenesis, moreover with randomly selected synaptic weights.

Figure 3.3 is a plot of a normalized fraction of cases that converged. In particular, we first compute the ratio of the corresponding curves in Figures 3.1 by those in Figure 3.2. This curve in turn is normalized by its own value at the learning of Greek point without neurogenesis.

Figure 3.4 is a plot corresponding to $\left(\bar{f},\overline{N}_2,\overline{N}_3\right)(\bar{r})$. Namely it is a plot of the average of the three curves in each of the Figures 3.1-3.3. These averages (over 1200 runs each) are denoted respectively (i) with square tic marks, (ii) with diamond tic marks, and (iii) with triangular tic marks.

## 4. DISCUSSION

We employed an autonomous (i.e., unsupervised) form of learning to model and simulate the recording of information in the hippocampus. This required that the model be capable of developing memory traces that are intrinsic representations (intrinsic endogenous encodings) of the information to be learned. Demonstrating that a neural system has the functionality to do this is a key and novel feature of the present approach.

Our simulation models learning the Roman and Greek alphabets. The ability to learn the Greek alphabet after the Roman alphabet has been learned is favorably informed by neurogenesis. We see this by examining the red curves with square tic marks in Figures3.1, 3.3 and 3.4. The leftmost tic mark (the learning of Roman) of this curve corresponds to a considerably higher value (approximately 30-40% more learning epochs required) than the curve's second left most tic mark (learning of the Greek following the Roman without apoptosis and neurogenesis). We interpret this, noting that the Roman learning places the network in a favorable posture for that subsequent learning of the Greek. By favorable posture, we mean that the synaptic weights developed by the learning of the Roman characters already reflect information about the 14 upper case characters that the alphabets have in common.

Next we see that neurogenesis increases this efficaciousness of the learning of Greek. The red curve descends with each increase of the number $g$ of new neurons that are made available by the neurogenesis to the Greek learning, up to a point. When too many neurons are replaced the improvement that we see levels off and then begins to reverse. We interpret this by noting that while the new neurons are aiding the learning of the Greek by replacing older neurons whose synaptic weights have become saturated[10], the apoptosis of the old neurons trained on the Roman causes a progressive loss of that Roman alphabet information (the commonality of the alphabets) that was responsible for the initial gain of the 30-40% cited above. So the U-shape of the red learning curve comes from a tradeoff of a learning effect and a forgetting effect

We are aware that our observations may inform issues of natural selection.

---

[10] A weight is saturated when its value has been driven by the dynamics to be near the ceiling $C$.

## REFERENCES

Aakerlund L, Hemmingsen R (1998) Neural networks as models of psychopathology. *Biol Psychiatry* **43**: 471-482.

Chambers RA, Potenza MN, Hoffman RE, Miranker WL, "Simulated Apoptosis/Neurogenesis Regulates Learning and Memory Capabilities of Adaptive Neural Networks", *Neuropsychopharmacology*, **29.** 2004.

Erikson PS, Perfilieva E, Bjork-Eriksson TB, Alborn A, Nordberg C, Peterson DA, Gage FH (1998) Neurogenesis in the adult human hippocampus. *Nat Medicine* 4: 1313-1317.

Gould E, Beylin A, Tanapat P, Reeves A, Shors T (1999) Learning enhances adult neurogenesis in the hippocampal formation. *Nat Neurosci* 2: 260-265.

Haykin S (1999) *Neural Networks a Comprehensive Foundation,* Prentice Hall, Upper Saddle River.

Kornack DR, Rakic P (2001) The generation, migration, and differentiation of olfactory neurons in the adult primate brain. *PNAS*, USA 98: 4753-4757.

Makakis EA, Gage FH (1999) Adult-generated neurons in the dentate gyrus send axonal projections to field CA3 and are surrounded by synaptic vesicles. *J Comp Neuro* 406: 449-460.

**Acknowledgements:** The author is grateful to K. Azimi for programming assistance and to R.A Chambers for helpful suggestions.