Supplementary Material

To accompany the article 'Multi-scale modeling of gene-behavior associations in an artificial neural network model of cognitive development' by M. S. C. Thomas, N. A. Forrester, and A. Ronald

Introduction

This document contains technical details to accompany computer simulations that explore a population of networks acquiring the past tense domain. This domain is here used as a representative abstract learning problem within cognition. Individual variability is included both in the parameters of the artificial neural networks which model the children's learning systems, and the learning environment to which they are exposed. The parameters of the artificial neural networks are encoded in an artificial genome. Population variability in parameters is created by generating populations of artificial genomes. Each genome is realized as a parameterized network. The network is exposed to an individualized learning environment, generating a trajectory of behavioral development. The inclusion of an artificial genome level in the simulations allows us to study the associations that can arise between values on the artificial genome and behavioral variability that is the product of an implemented developmental process.

In the following, we describe the computational parameters that varied in the artificial neural networks. We outline how the range of variation for each parameter in the population was established. We then describe the method for designing the artificial genome, and the assumptions that this method embodies. Finally, a set of lookup tables is

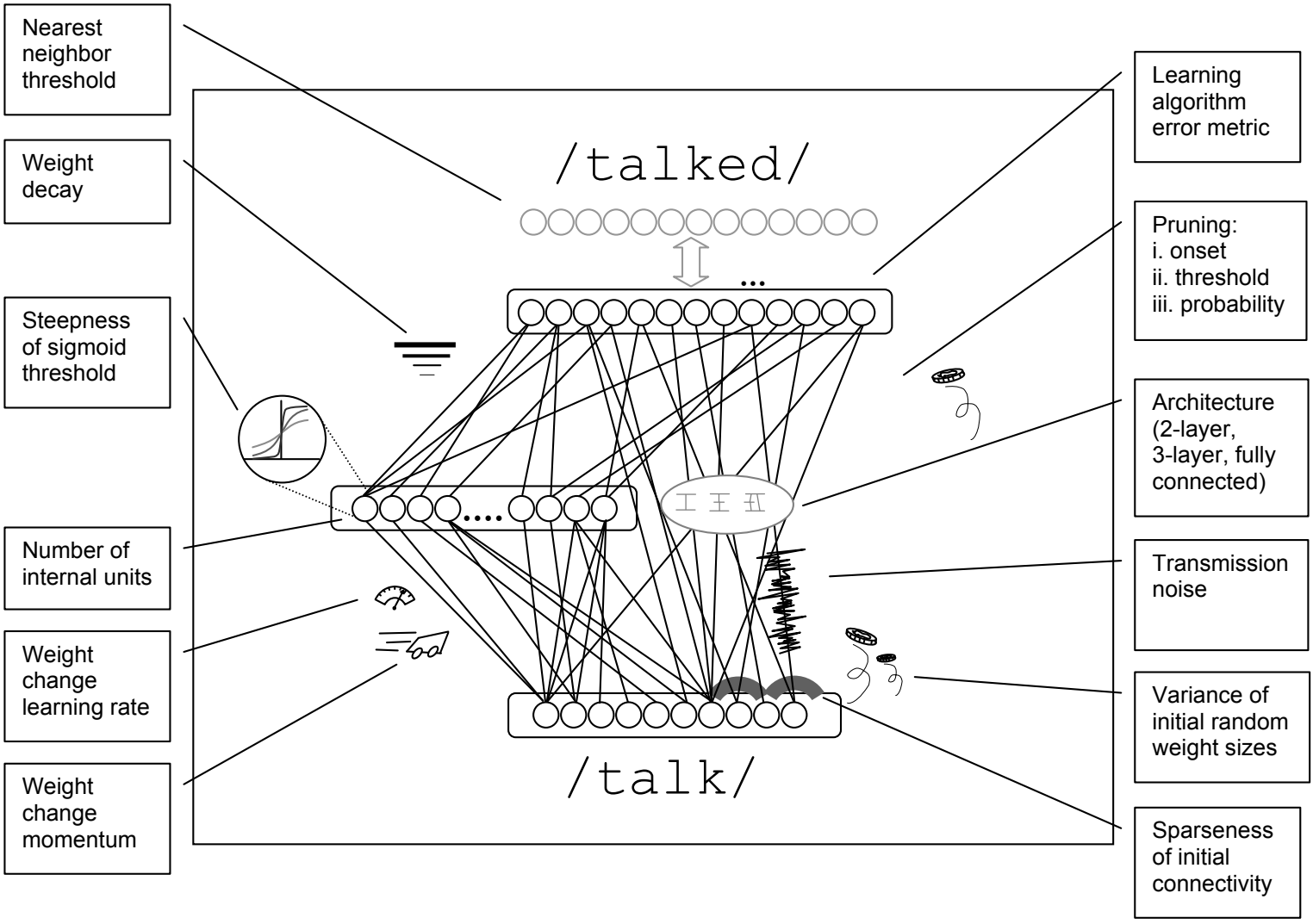
included detailing how values on the artificial genome were mapped to computational parameter values in the artificial neural networks.

The base past-tense model

Model architecture and parameters

The original connectionist model employed a three-layer artificial neural network, comprising an input layer, a layer of internal or 'hidden' units, and an output layer. It was trained using the backpropagation algorithm (Rumelhart, Hinton, & Williams, 1986), a type of supervised learning. The free parameters in the model were the number of hidden units, the learning rate, and the momentum (see below). An expanded set of 14 parameters was employed in the current simulations, in many cases to allow for additional analogues to known neurocomputational properties. However, backpropagation itself is not viewed as fully biologically plausible. We use it here in place of a more biologically plausible error-correction algorithm (see Thomas & McClelland, 2008, for discussion). An introduction to the idea that parameters in connectionist models can explain types of cognitive variability can be found in Thomas and Karmiloff-Smith, 2002a). The parameters and model architecture are depicted schematically in Figure 1.

Figure 1: Architecture of the connectionist model of English past-tense acquisition, showing the internal parameters that varied in the population.



The parameters were as follows:

Building the network:

- *Architecture*: In addition to the 3-layer network, a 2-layer network without a layer of hidden units, and a fully connected network were used. A 2-layer network has less computational power than a 3-layer network but learns more quickly. A fully connected network contains both direct connections from input to output and a hidden layer, and produces a computationally more powerful system. Networks could therefore have 1, 2, or 3 layers of connection weights. Previous connectionist models have proposed single or multiple pathways may be available to connect input and output (e.g., Westermann, 1998; Zorzi, Houghton & Butterworth, 1998), and that differential use of routes may explain individual differences in behavior (Harm & Seidenberg, 2004; Plaut, 1997; Thomas & Karmiloff-Smith, 2002b). Recent functional brain imaging of reading lend support to this proposal (e.g., Richardson et al., 2011; Seghier et al., 2008).
- *Hidden units*: For networks with a hidden unit layer, the number of hidden units could vary. Variations of the number of hidden units have been proposed to account for developmental deficits such as dyslexia (e.g., Harm & Seidenberg, 1999) and autism (e.g., Cohen, 1998), as well as individual differences (Richardson et al., 2006a, b). We did not vary the number of hidden layers. More hidden units within a layer increases computational power and the rate of learning, while more layers of hidden units increases computational power but slows down learning, since error must be

propagated from the output more deeply into the network to improve learning (see Richardson et al., 2006a,b, for a comparison of these conditions).

- *Sparseness*: The architecture determined how many layers of connection weights existed. Of the potential connections in a layer, only a certain proportion was created. The sparseness parameter set the probability that any given connection would be created. Greater connectivity increases computational power, but can lead to slower learning. Under some conditions, it can also lead to poorer generalization, since greater integration of information causes more item-specific and context-specific learning (see McClelland, 2000, for a proposal that conjunctive coding may cause autistic symptoms; and conversely, Beversdorf, Narayanan & Hughes, 2007, for a proposal that the symptoms arise from sparse connectivity).
- *Weight variance*: Connection weights were assigned an initial random value within a range depending on this parameter. E.g., if set to 0.5, weights would be randomized between +/- 0.5. Large initial weights take time to unlearn, which slows learning (an effect known as entrenchment; see Munakata & McClelland, 2003, for discussion).

Processing dynamics:

- *Processing noise*: The net activation a receiving unit receives from a given sending unit is a product of the sending unit's activation and the connection strength between them. Transmission noise was added to this net activation. Gaussian noise was used and the parameter specified the standard deviation of the noise distribution around zero. Noise has been used to simulate under-specified representations in development (e.g., to simulate Specific Language Impairment: Joanisse & Seidenberg, 2003; or as a

candidate explanation of autistic symptoms: Simmons et al., 2007), and has also been proposed as an essential primitive in neural processing (McClelland, 1993).

- *Unit threshold function*: A receiving unit sums the net activation from all sending units and uses an activation function to determine its consequent output. We used a common non-linear activation function, the sigmoid or logistic function, equivalent to a smoothed threshold. This function has a free parameter, the 'temperature', which makes the smoothed threshold either steeper or shallower. The activation function was:

$$Output = \frac{1}{1 + e^{-temperature \times (netinput + bias)}}$$

where *netinput* is the summed activation to a unit, *bias* is the negative of the unit's threshold, and *Output* is the unit's activation state in response to this input. A shallow function (low temperature) denies a unit the opportunity to make large output changes in response to small changes in net input, whereas a steep function (high temperature) approximates a non-smoothed threshold, thereby producing a unit with binary response characteristics. Variations in the slope of the sigmoid function have been proposed as candidate explanations of disorders such as specific language impairment (Thomas, 2005) and schizophrenia (Cohen & Servan-Schreiber, 1992), as well as ageing (Li & Lindenberger, 1999). Changes to the slope of the sigmoid have a number of effects on learning. A shallow slope means that processing units are less sensitive to small differences in their input. This poor discriminability means they will be slow to learn categorizations that rely on small distinctions in the input. Secondly, in the backpropagation algorithm, weight update for a given error signal is proportional to the slope on the sigmoid (the differential of the function). If the function resembles a

gentle S-shape, then the slope across the range of unit activations will be small. A shallow sigmoid will lead directly to slower learning. Conversely, if the temperature is very high, producing a sigmoid similar to a step function, for most inputs to a unit, it will be jammed on or off ('saturated') rather than in its dynamic range. When a unit is saturated, the slope on the sigmoid function is flatter (the regions below or above the step). When it is in its dynamic range it is steep (the step). If a unit is predominantly saturated due to a high temperature, the flat slope will again lead to small weight changes for a given error signal and therefore slow learning. Finally, units with high temperatures flip between being saturated on or off. They are therefore ill suited to learning mappings requiring graduations of activation states. In sum, temperatures that are either too high or too low can delay learning.

Network maintenance:

- *Connection weight decay*: each connection's magnitude was reduced by a small proportion on each presentation of a training pattern, according to the weight decay parameter. The approximate range of weight decay values was derived by estimating a percentage of weight value that could plausibly be lost overall all of training (e.g., 50%), and then dividing this proportion by the number of training epochs (e.g., 1000) and the number of training patterns presented on each epoch (e.g., 508), to give a proportional reduction in the connection weights to be applied on each pattern presentation (e.g., $0.5/1000/508=9.84 \times 10^{-7}$). To our knowledge, weight decay has not been used as a candidate mechanism to explain individual variability.

- We did not simulate the increase in synaptic density observed in human cortex during infancy and early childhood, instead simulating the outcome of this process through variations in the sparseness of connectivity; we did, however, implemented the pruning of spare resources from mid-childhood (Huttenlocher, 2002). The *pruning process* eliminated small connection weights. Variations in pruning have been proposed as an explanation of autistic symptoms, and specifically developmental regression (Thomas, Knowland & Karmiloff-Smith, 2011). The pruning process involved three parameters: onset, threshold, and probability:
- *Connection pruning – onset*: Connections that were not being used were probabilistically pruned away after a certain point in training. The onset parameter determined the point in training when pruning began (see Thomas & Johnson, 2006, for simulations of pruning applied to sensitive periods in plasticity).
- *Connection pruning – threshold*: Connections stood a chance of being pruned after onset only if their magnitude fell below a threshold determined by this parameter. The rationale is that small weights are assumed not to transmit strong activations and therefore not to be playing a key role in computations. They may therefore be removed to save on resources.
- *Connection pruning – probability*: If the magnitude of a connection fell below threshold after pruning had begun, it was eliminated probabilistically based on this parameter. High probability leads to faster loss of unused connections. Low probability leads to slower loss.

Network adaptation:

- *Learning algorithm error measure*: The backpropagation algorithm was used with two different metrics to determine the error signal marking the disparity between the network's current output and its intended target. These were Euclidean distance and cross-entropy (Hinton, 1989). The Euclidean distance metric produces less weight change for a unit when it is committed to an erroneous response than the cross-entropy measure. That is, when a unit is stuck on in a saturated state but the learning algorithm requires it to be off, or vice versa, cross-entropy will lead to faster changes to its weights to change its activation state than Euclidean distance. Under some conditions, cross-entropy can therefore be a more plastic learning algorithm, leading to faster learning and higher ceiling performance.
- *Learning rate*: This parameter determined how much the connection weights were altered in response to a certain disparity between output and target during supervised learning. A large learning rate produces a system that learns more quickly but that also may be unstable, flipping between good performance on different parts of the problem domain. Differences in learning rate have been proposed as explanations of individual differences in cognitive ability (Richardson et al., 2006a,b) and general intelligence (Garlick, 2002), as well as developmental deficits (e.g., dyslexia; Harm & Seidenberg, 1999).
- *Momentum*: This parameter allowed some proportion of the weight change on the previous learning trial to be carried over. It serves a smoothing function to prevent learning from getting stuck in local, sub-optimal solutions. While a parameter often

varied in connectionist models of development, it has not to my knowledge been used as a candidate explanation for individual differences in learning.

Network response:

- *Nearest neighbor threshold*: Network output comprised a vector of continuous activation values between 0 and 1, while legal responses of the network were binary vectors. An algorithm determined which legal phoneme was closest to the activation patterns at onset, nucleus, and coda. However, the phoneme was only recognized as a response if the activation was sufficiently close to the legal phoneme (using a root mean square or RMS measure). This was determined by the nearest neighbor threshold. (The legal phonemes could of course still be the incorrect ones for the target verb). The nearest neighbor computation may be viewed as equivalent to the settling of an unimplemented recurrent attractor network into a particular response state (see Plaut et al., 1996, for a model of reading development in which this attractor network was implemented). The nearest neighbor threshold parameter then indexes the efficiency of this attractor network to generate a response within some notional deadline. A high threshold allows an approximate output to be recognized as correct (i.e., larger error is tolerated); a low threshold requires a more exact initial output. The use of a nearest neighbor algorithm allowed the network to generate accuracy levels. Differences in the functioning of the attractor network (sometimes called 'clean-up' units) have been proposed as a candidate explanation of developmental deficits (e.g., dyslexia; Harm & & Seidenberg, 1999).

Calibrating parametric variation

Calibration was carried out to establish the full range of variation for each parameter over which the artificial neural network exhibited some degree of learning. In general, the network was fairly robust to variation in its parameters, as illustrated in Appendix A.

Two of the network parameters were categorical: the architecture and learning algorithm metric. The others were continuously valued. In order to produce variability in the population according to these remaining parameters, they were calibrated as follows. An initial 'normal' set of parameters was defined. These were estimated based on previous research. Each of the continuously valued parameters was then varied in turn, holding the all other parameters at their initial values. For each parameter, the range was derived that produced failure of learning up to highly successful learning. In some cases, parameters had a monotonic relationship to performance (e.g., hidden units, where more was better); in other cases, there was an optimal intermediate value (e.g., activation function). The functions linking a given parameter and behavioral outcomes, with all other parameters held constant, are included in Appendix A. The aim was to determine an average or adequate value for each parameter, which was defined heuristically as 'just enough to succeed and then a little bit more'. Values were then derived that would cause increasingly poorer or increasingly better performance around this value. We attempted to make poorer and better performance roughly symmetrical around average performance for each parameter. This caused some parameter ranges to be skewed. For example, 50 hidden units was determined as the average value in a 3-layer network. Values of 40 or 30 would cause poorer performance. However, to achieve equivalent differences above average level, 100 or 200 hidden units might be necessary. We chose to emphasize

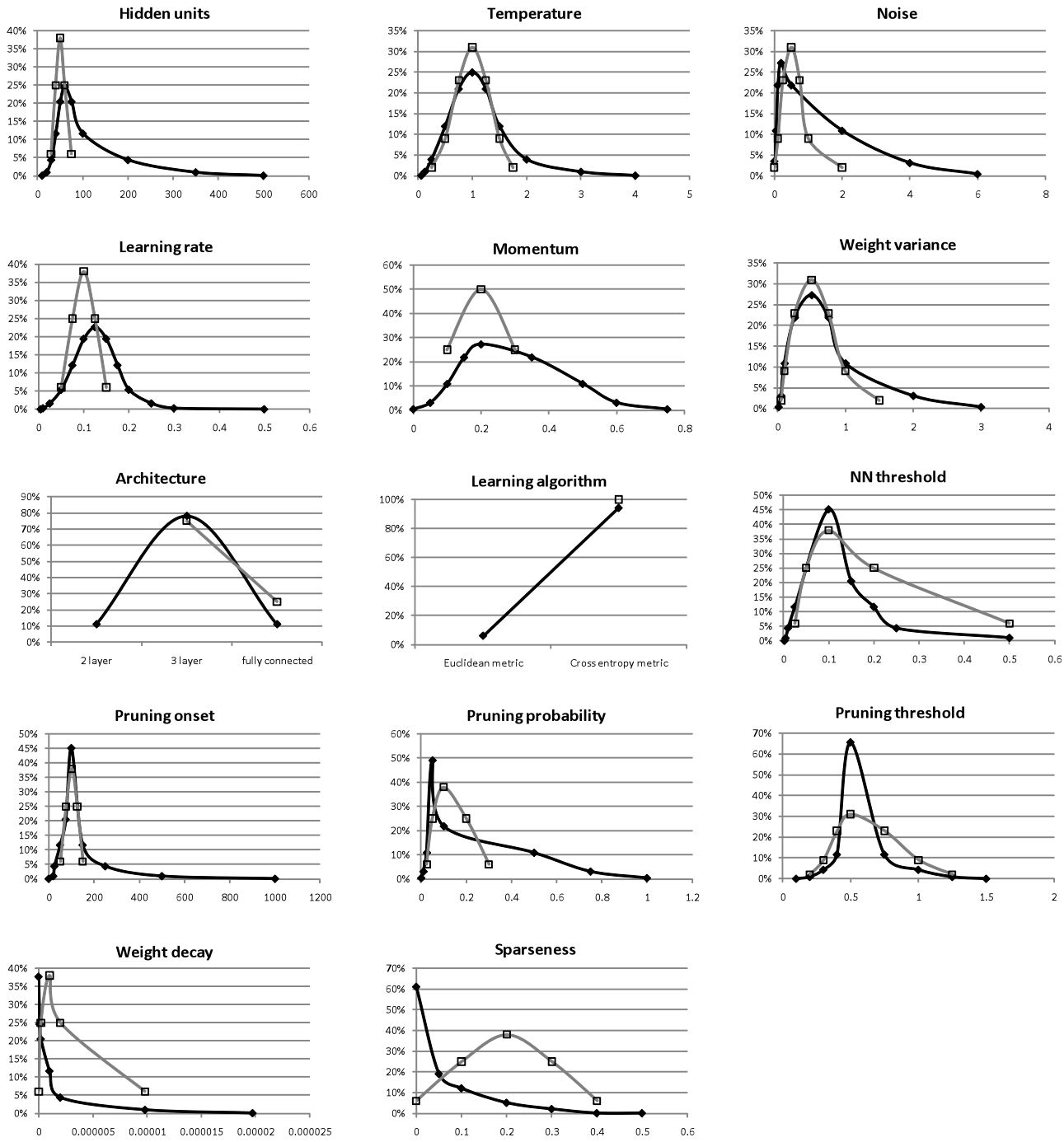
behavioral symmetry around the average parameter value rather than parametric symmetry, on the grounds that the symmetrical bell curve is a common pattern observed in human abilities. The ranges for each parameter for the phonology-to-phonology network are included in Figure 2.

We chose not to vary the input and output coding scheme. Our previous work suggests that, within certain limits, varying the problem encoding has similar effects on the developmental trajectory to altering computational parameters (Thomas & Karmiloff-Smith, 2003). However, recoding the problem domain can in principle have extreme effects on learnability, if key distinctions in the input or output are lost in the recoding. Some models of developmental language impairment and dyslexia propose that differences in the representation of phonology cause subsequent behavioral deficits in grammar and reading acquisition (e.g., Harm & Seidenberg, 1999; Hoeffner & McClelland, 1993; Joanisse, 2004).

Although only main effects of each parameter were considered as sources of variability during calibration, we expected interactions between these neurocomputational parameters in subsequent learning. To pick four examples: (i) large numbers of hidden units can partially compensate for a shallow sigmoid function in those processing units; (ii) having a more sparse initial connectivity is likely to reduce the amount of weights eliminated via pruning because their magnitudes will be larger; (iii) high weight decay can be countered by a higher learning rate; (iv) an over-aggressive pruning process (e.g., with a high threshold and high probability) can be alleviated if its onset occurs very late in training when weights have become large, but exacerbated if the onset is early. Large numbers of parameter combinations were possible within our

scheme: given the number of levels specified for each parameter, approximately two trillion unique parameter combinations were available.

Figure 2: Parameter values and target population frequencies (dark lines).



In Figure 2, dark lines show parameter values (x-axis) and their target frequencies in the population (y-axis) for each of the 14 computational parameters. Each gene had two alleles, coded as binary values. Several genes coded for each parameter value. Sets of binary values were summed and a look-up table used to derive each parameter value. The numbers of binary alleles for each parameter were as follows: hidden units: 10; temperature: 10; noise: 8; learning rate: 12; momentum: 8; weight variance: 8; architecture: 6; learning algorithm: 4; nearest neighbor threshold: 10; pruning onset epoch: 10; pruning probability: 8; pruning threshold: 10; weight decay: 10; sparseness: 12 (total 126 bits). The grey lines show the functions for a condition in which parameter variation was narrower. We do not consider this condition further here.

Specifying an artificial genome for the model

The use of genetic algorithms entails creation of an artificial genome to encode the neural network's parameter values, such that all possible genomes correspond to legal parameter sets. In creating the genome, we made the following assumptions:

- There were two copies of each gene, with genes residing on pairs of chromosomes.
- For simplicity, each gene had only two variants or alleles.
- The two alleles produced different outcomes in the functionality of the neurocomputational parameter which they encoded.
- The influence of genes was intended to be *additive*: we did not include dominant or recessive effects, and genes had the same effect in combination as in isolation.

This constraint was motivated by the finding within behavioral genetics that the effect of gene variants is predominantly additive on phenotypic outcomes (Plomin et al., 2008). Nevertheless, our method of implementing the mapping between gene variants and neurocomputational parameters did inadvertently produce some non-additive effects.

- All neurocomputational parameters were *polygenic*. That is, their value was determined by the additive action of a collection of genes.
- In the first instance, we assumed that the action of genes was not *pleiotropic*; that is, with respect to neurocomputational parameters, we assumed that no gene affected the value of more than one parameter at once. This simplification likely will not hold in many cases, and certainly the current theoretical view is that the relationship between genes and *cognitive processes* is pleiotropic (see, e.g., Kovas & Plomin, 2006).

The assumption of polygenicity was motivated by the fact that we are using computational models to capture cognitive-level phenomena, and is a point worth emphasizing. We expect many low-level neural variations to influence neurocomputational functions at the level of cognitive processes in neural circuits. We therefore view it as unlikely that a single gene would modulate a neurocomputational parameter responsible for normal cognitive variation.

We assumed, for reasons of simplification only, that the combination of alleles for each polygenic neurocomputational parameter had a deterministic relation to the value of that parameter in the instantiated network: that is, the allele set alone determined the

parameter value. Alternatively, this may be viewed as the assumption that the relation between alleles and parameter setting relied on an environment that did not vary across the individuals in the simulated population. We assumed (and did not instantiate) a much larger part of the genome that was species universal and was responsible for the basics of, for example, creating the processing units, the connections, the activation dynamics, the sensorium, the input-output connectivity pathways, and the mechanics of experience-dependent systems.

Parameter values and their link to the artificial genome for the past tense network

For the basic past tense network, the total of number of genes used to encode the value of the 14 computational parameters was 126 (or two copies of 63) as follows – hidden units: 10; temperature: 10; noise: 8; learning rate: 12; momentum: 8; weight variance: 8; architecture: 6; learning algorithm: 4; nearest neighbor threshold: 10; pruning onset epoch: 10; pruning probability: 8; pruning threshold: 10; weight decay: 10; sparseness: 12 (total 126 bits).

Figure 2 plots the range of values for each parameter against their target frequency of occurrence in the population. The translation of a genome into a parameter set was implemented by assigning alleles the value of 1 or 0, and then deriving the total for all the genes influencing the parameter. The parameter value was calculated from the total using a lookup table, created by hand for each parameter to reflect the range of values identified during the calibration stage. The lookup tables for the 14 parameters (in the Wide Genetic used) condition used in the association simulations are shown below.

Table 1. Lookup table linking the artificial genome to the Hidden Unit parameter, for the Wide Genetic Variation condition

	Hidden Unit Parameter Value										
Number of 1-valued alleles	0	1	2	3	4	5	6	7	8	9	10
Population probability	0.001	0.01	0.04	0.12	0.21	0.25	0.21	0.12	0.04	0.01	0.001
Parameter value	10	20	30	40	50	60	75	100	200	350	500

Table 2. Lookup table linking the artificial genome to the Temperature parameter, for the Wide Genetic Variation condition

	Temperature Parameter Value										
Number of 1-valued alleles	0	1	2	3	4	5	6	7	8	9	10
Population probability	0.001	0.01	0.04	0.12	0.21	0.25	0.21	0.12	0.04	0.01	0.001
Parameter value	0.0625	0.125	0.25	0.5	0.75	1	1.25	1.5	2	3	4

Table 3. Lookup table linking the artificial genome to the Noise parameter, for the Wide Genetic Variation condition

	Noise Parameter Value								
Number of 1-valued alleles	0	1	2	3	4	5	6	7	8
Population probability	-	0.04	0.11	0.22	0.27	0.22	0.11	0.03	0.00
Parameter value	0	0	0.05	0.1	0.2	0.5	2	4	6

Table 4. Lookup table linking the artificial genome to the Learning Rate parameter, for the Wide Genetic Variation condition

	Learning Rate Parameter Value												
Number of 1-valued	0	1	2	3	4	5	6	7	8	9	10	11	12

alleles													
Population probability	0.0002	0.0029	0.02	0.05	0.12	0.19	0.23	0.19	0.12	0.05	0.02	0.0029	0.0002
Parameter value	0.005	0.01	0.025	0.05	0.075	0.1	0.125	0.15	0.175	0.2	0.25	0.3	0.5

Table 5. Lookup table linking the artificial genome to the Momentum parameter, for the Wide Genetic Variation condition

	Momentum Parameter Value								
Number of 1-valued alleles	0	1	2	3	4	5	6	7	8
Population probability	0.004	0.03	0.11	0.22	0.27	0.22	0.11	0.03	0.004
Parameter value	0	0.05	0.1	0.15	0.2	0.35	0.5	0.6	0.75

Table 6. Lookup table linking the artificial genome to the Weight Variation parameter, for the Wide Genetic Variation condition

	Weight Variation Parameter Value								
Number of 1-valued alleles	0	1	2	3	4	5	6	7	8
Population probability	0.004	0.03	0.11	0.22	0.27	0.22	0.11	0.03	0.004
Parameter value	0.01	0.05	0.1	0.25	0.5	0.75	1	2	3

Table 7. Lookup table linking the artificial genome to the Architecture parameter, for the Wide Genetic Variation condition. (0 = 2-layer, 1 = 3-layer, 2 = fully-connected)

	Architecture Parameter Value						
Number of 1-valued alleles	0	1	2	3	4	5	6
Population probability	-	0.109	-	0.781	-	0.109	-
Parameter value	0	0	1	1	1	2	2

Table 8. Lookup table linking the artificial genome to the Learning Algorithm parameter, for the Wide Genetic Variation condition. (0 = Euclidean distance error metric, 1 = cross-entropy error metric)

	Learning Algorithm Parameter Value				
Number of 1-valued alleles	0	1	2	3	4
Population probability	0.063	0.938	-	-	-
Parameter value	0	1	1	1	1

Table 9. Lookup table linking the artificial genome to the Nearest Neighbor Threshold parameter, for the Wide Genetic Variation condition

	Nearest Neighbor Threshold Parameter Value										
Number of 1-valued alleles	0	1	2	3	4	5	6	7	8	9	10
Population probability	0.001	0.010	0.044	0.117	0.451	-	0.205	0.117	0.044	0.011	-
Parameter value	0.0025	0.005	0.01	0.025	0.1	0.1	0.15	0.2	0.25	0.5	0.5

Table 10. Lookup table linking the artificial genome to the Pruning Onset parameter, for the Wide Genetic Variation condition

	Pruning Onset Parameter Value										
Number of 1-valued alleles	0	1	2	3	4	5	6	7	8	9	10
Population probability	0.001	0.01	0.04	0.12	-	0.45	0.21	0.12	0.04	0.01	0.001
Parameter value	1000	500	250	150	100	100	75	50	25	20	0

Table 11. Lookup table linking the artificial genome to the Pruning Probability parameter, for the Wide Genetic Variation condition

	Pruning Probability Parameter Value								
Number of 1-	0	1	2	3	4	5	6	7	8

valued alleles									
Population probability	0.004	0.03	0.11	-	0.49	0.22	0.11	0.03	0.004
Parameter value	0	0.01	0.025	0.05	0.05	0.1	0.5	0.75	1

Table 12. Lookup table linking the artificial genome to the Pruning Threshold parameter, for the Wide Genetic Variation condition

	Pruning Threshold Parameter Value										
Number of 1-valued alleles	0	1	2	3	4	5	6	7	8	9	10
Population probability (%)	0.001	0.01	0.04	0.12	-	0.66	-	0.12	0.04	0.01	0.001
Parameter value	0.1	0.2	0.3	0.4	0.5	0.5	0.5	0.75	1	1.25	1.5

Table 13. Lookup table linking the artificial genome to the Weight Decay parameter, for the Wide Genetic Variation condition

	Weight Decay Parameter Value										
Number of 1-valued alleles	0	1	2	3	4	5	6	7	8	9	10
Population probability	-	-	-	-	0.38	0.25	0.21	0.12	0.04	0.01	0.001
Parameter value	0	0	0	0	0	$1x10^{-7}$	$2x10^{-7}$	$9.8x10^{-7}$	$19.7x10^{-7}$	$98.4x10^{-7}$	$196.9x10^{-7}$

Table 14. Lookup table linking the artificial genome to the Sparseness parameter, for the Wide Genetic Variation condition

	Sparseness Parameter Value												
Number of 1-valued alleles	0	1	2	3	4	5	6	7	8	9	10	11	12
Population probability	-	-	-	-	-	-	0.61	0.19	0.12	0.05	0.02	0.003	0.0002
Parameter value	0	0	0	0	0	0	0	0.05	0.1	0.2	0.3	0.4	0.5

References

Beversdorf, D. Q., Narayanan, A., Hillier, A., & Hughes, J. D. (2007). Network model of decreased context utilization in autism spectrum disorder. *Journal of Autism and Developmental Disorders*, *37*, 1040– 1048.

Cohen, I. L. (1998). Neural network analysis of learning in autism. In D. J. Stein & J. Ludik (Eds.), *Neural networks and psychopathology* (pp. 274–315). New York, NY: Cambridge University Press.

Harm, M. W. & Seidenberg, M. S. (1999). Phonology, reading acquisition, and dyslexia: Insights from connectionist models. *Psychological Review*, *106*, 491-528.

Harm, M. W., & Seidenberg, M. S. (2004). Computing the meanings of words in reading: Cooperative division of labor between visual and phonological processes. *Psychological Review*, *111*, 662-720.

Hinton, G. (1989). Connectionist learning procedures. *Artificial Intelligence*, *40*, 185–234.

Hoeffner, J. H. & McClelland, J. L. (1993). Can a perceptual processing deficit explain the impairment of inflectional morphology in developmental dysphasia? A computational investigation. In E.V. Clark (Ed), *Proceedings of the 25th Child language research forum*, (pp. 1-25). Stanford University Press.

Huttenlocher, P. R. (2002). *Neural plasticity: The effects of environment on the development of the cerebral cortex.* Cambridge, MA: Harvard University Press.

Joanisse, M. F., & Seidenberg, M. S. (1999). Impairments in verb morphology after brain injury: A connectionist model. *Proceedings of the National Academy of Sciences of the United States of America*, *96*, 7592-7597.

Joanisse, M. F. (2004). Specific language impairments in children: Phonology, semantics and the English past tense. *Current Directions in Psychological Science*, *13(4)*, 156-160.

Joanisse, M. F., Seidenberg, M. S. (2003). Phonology and syntax in Specific Language Impairments: Evidence from a connectionist model. *Brain and Language*, *86*, 40-56.

Karaminis, T. N., & Thomas, M. S. C. (2010). A cross-linguistic model of the acquisition of inflectional morphology in English and Modern Greek. In S. Ohlsson & R. Catrambone (Eds.), *Proceedings of 32nd Annual Conference of the Cognitive Science Society*, August 11-14, 2010, Portland, Oregon, USA.

Kovas, Y., & Plomin, R. (2006). Generalist genes: implications for the cognitive sciences. *Trends in Cognitive Sciences*, *10(5)*, 198-203.

Li, S.-C. & Lindenberger, U. (1999). Cross-level unification: A computational exploration of the link between deterioration of neurotransmitter systems and the dedifferentiation of cognitive abilities in old age, (pp. 103-146) in L.-G. Nilsson & H. Markowitsch (Eds.). *Cognitive neuroscience of memory*. Toronto: Hogrefe & Huber

McClelland, J. L. (1993) Toward a theory of information processing in graded, random, interactive networks. In D. E. Meyer & S. Kornblum (Eds.), *Attention and performance XIV: Synergies in experimental psychology, artificial intelligence and cognitive neuroscience* (pp. 655-688). Cambridge, MA: MIT Press.

McClelland, J. L. (2000). The basis of hyperspecificity in autism: A preliminary suggestion based on properties of neural nets. *Journal of Autism and Developmental Disorders*, *30*, 497–502

Munakata, Y., & McClelland, J. L. (2003). Connectionist models of development. *Developmental Science*, *6(4)*, 413-429.

Plaut, D. C. (1997). Structure and function in the lexical system: Insights from distributed models of word reading and lexical decision. *Language and Cognitive Processes*, *12*, 767-808.

Plaut, D. C., McClelland, J. L., Seidenberg, M. S., & Patterson, K. E. (1996). Understanding normal and impaired word reading: Computational principles in quasi-regular domains. *Psychological Review*, *103*, 56-115.

Plomin, R., DeFries, J. C., McClearn, G. E., & McGuffin, P. (2008). *Behavioral genetics (5th Edition)*. New York: Worth Publishers.

Plunkett, K. & Marchman, V. (1991). U-shaped learning and frequency effects in a multilayered perceptron: Implications for child language acquisition. *Cognition*, *38*, 1-60.

Plunkett, K. & Marchman, V. (1993). From rote learning to system building: acquiring verb morphology in children and connectionist nets. *Cognition*, *48*, 21-69.

Richardson, F. M., Forrester, N. A., Baughman, F. D., & Thomas, M. S. C. (2006b). Computational modeling of variability in the conservation Task. In *Proceedings of the 28th Annual Conference of the Cognitive Science Society* (p. 2010-2015), July 26-29, Vancouver, BC, Canada.

Richardson, F. M., Seghier, M. L., Leff, A. P., Thomas, M. S. C., & Price, C. J. (2011). Multiple routes from occipital to temporal cortices during reading. *Journal of Neuroscience*, *31(22)*, 8239-8247.

Richardson, F.M., Baughman, F.D., Forrester, N. A., & Thomas, M.S.C. (2006a). Computational modeling of variability in the balance scale task. *Proceedings of the 7th International Conference of Cognitive Modeling*.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error propagation. In D. E. Rumelhart, J. L. McClelland and The PDP Research Group, *Parallel distributed processing: Explorations in the microstructure of cognition. Vol. 1: Foundations* (pp. 318-362). Cambridge, MA: MIT Press.

Sapolsky, R. (2005). *Biology and human behavior: The neurological origins of individuality (2nd Ed.)*. Chantilly, VA: The Teaching Company.

Seghier, M. L., Lee, H. L., Schofield, T., Ellis, C. L., & Price, C. (2008). Inter-subject variability in the use of two different neuronal networks for reading aloud familiar words. *Neuroimage*, *42(3-3)*, 1226-1236.

Simmons, D. R., McKay, L., McAleer, P., Toal, E., Robertson, A., & Pollick, F. E. (2007). Neural noise and autism spectrum disorders. *Perception*, *36(Suppl.)*, 119–120.

Strachan, T., & Read, P. (2003). *Human molecular genetics 3*. Garland Publishing.

Thomas, M. S. C. (2005). Characterising compensation. *Cortex*, *41(3)*, 434-442.

Thomas, M. S. C., Forrester, N. A., & Ronald, A. (2013, April 1). Modeling socio-economic status effects on language development. *Developmental Psychology*. Advance online publication. doi: 10.1037/a0032301.

Thomas, M. S. C. & Johnson, M. H. (2006). The computational modeling of sensitive periods. *Developmental Psychobiology*, *48(4)*, 337-344.

Thomas, M. S. C. & Knowland, V. C. P. (2014). Modeling mechanisms of persisting and resolving delay in language development. *Journal of Speech, Language, and Hearing Research*.

Thomas, M. S. C., Knowland, V. C. P., & Karmiloff-Smith, A. (2011). Mechanisms of developmental regression in autism and the broader phenotype: A neural network modeling approach. *Psychological Review*, *118(4)*, 637-654.

Thomas, M. S. C. & Karmiloff-Smith, A. (2002b). Are developmental disorders like cases of adult brain damage? Implications from connectionist modeling. *Behavioral and Brain Sciences*, *25(6)*, 727-788.

Thomas, M. S. C. & Karmiloff-Smith, A. (2003). Modeling language acquisition in atypical phenotypes. *Psychological Review*, *Vol. 110*, *No.4*, 647-682.

Thomas, M. S. C. & Karmiloff-Smith, A. (2003a). Connectionist models of development, developmental disorders and individual differences. In R. J. Sternberg, J. Lautrey, & T. Lubart (Eds.), *Models of Intelligence: International Perspectives*, (p. 133-150). American Psychological Association.

Thomas, M. S. C., & McClelland, J. L. (2008). Connectionist models of cognition. In R. Sun (Ed.), *Cambridge handbook of computational cognitive modeling* (pp. 23-58). Cambridge: Cambridge University Press.

Westermann, G (1998) Emergent modularity and U-shaped learning in a constructivist neural network learning the English past tense. In M. A. Gernsbacher & S. J. Derry (Eds.), *Proceedings of the Twentieth Annual Conference of the Cognitive Science Society* (pp. 1130-1135). Hillsdale, NJ: Erlbaum.

Woollams, A. M., Joanisse, M., & Patterson, K. (2009). Past-tense generation from form versus meaning: Behavioral data and simulation evidence. *Journal of Memory and Language*, *61*, 55-76.

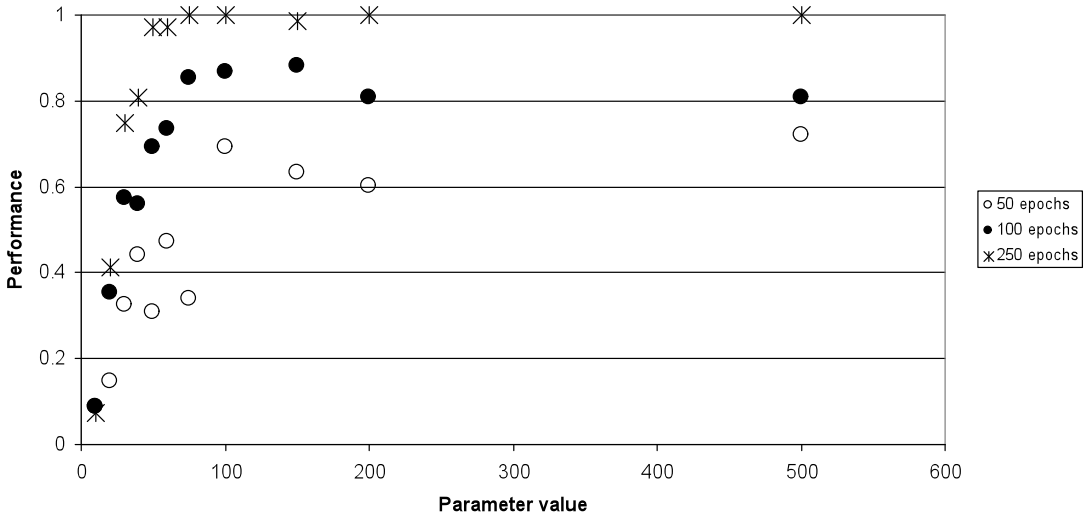
Zorzi, M., Houghton, G., & Butterworth, B. (1998). Two routes or one in reading aloud? A connectionist dual-process model. *Journal of Experimental Psychology: Human Perception and Performance*, *24*, 1131-1161.

Appendix A

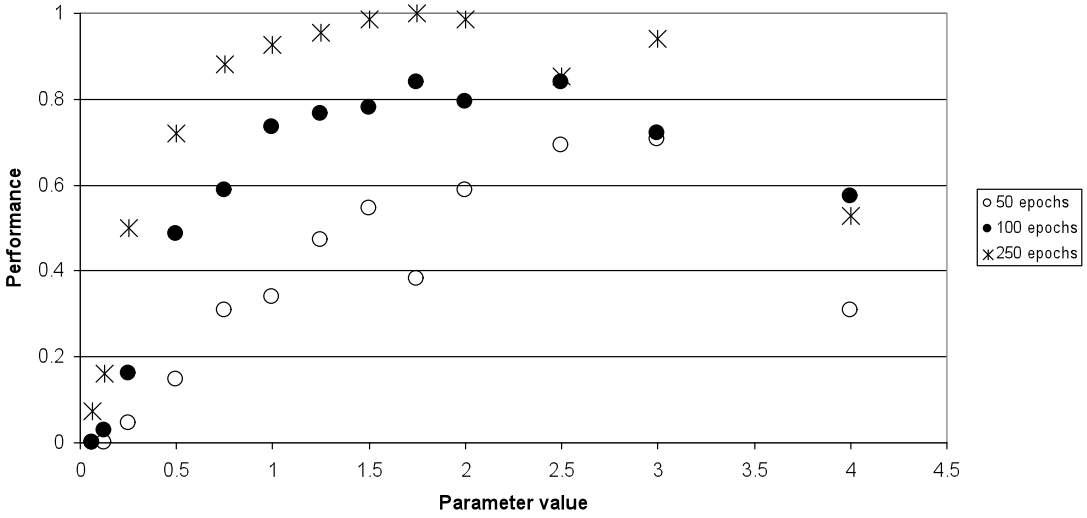
Parameter-behavior functions for the basic past tense network

All other parameters were held at default values, while a single parameter was varied. Performance is reported for regular verbs on the past tense task. The shape of the function is displayed for three points in training, 50, 100, and 250 epochs. The default parameter values were: hidden units: 50; temperature: 1; noise: 0; learning rate: 0.01; momentum: 0.2; weight variance: 0.5; architecture: 3-layer; learning algorithm: back propagation error measure; nearest neighbor threshold: 0.1; pruning onset epoch: 50; pruning probability: 0.1; pruning threshold: 0.5; weight decay: 0.000019; sparseness: 90% connectivity.

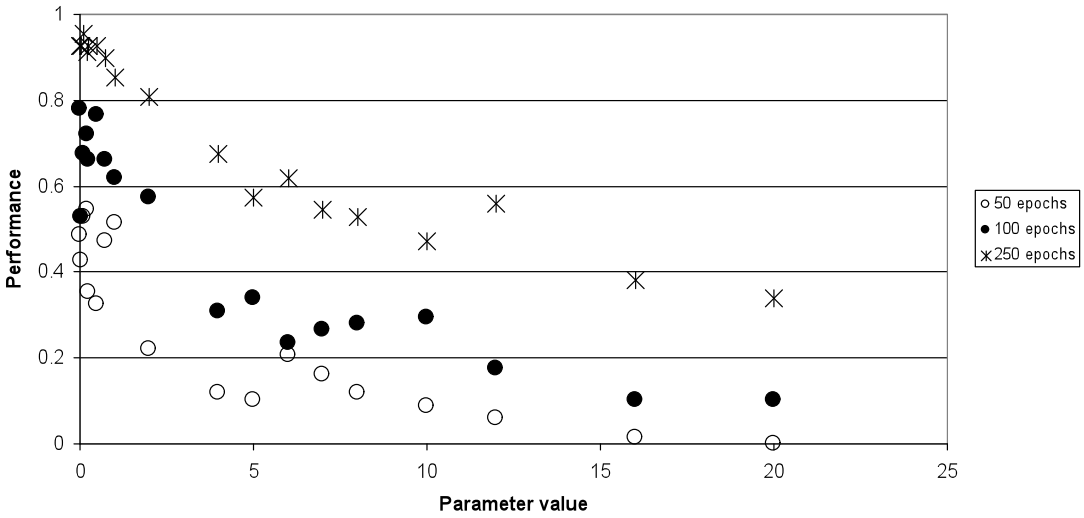
Hidden units:



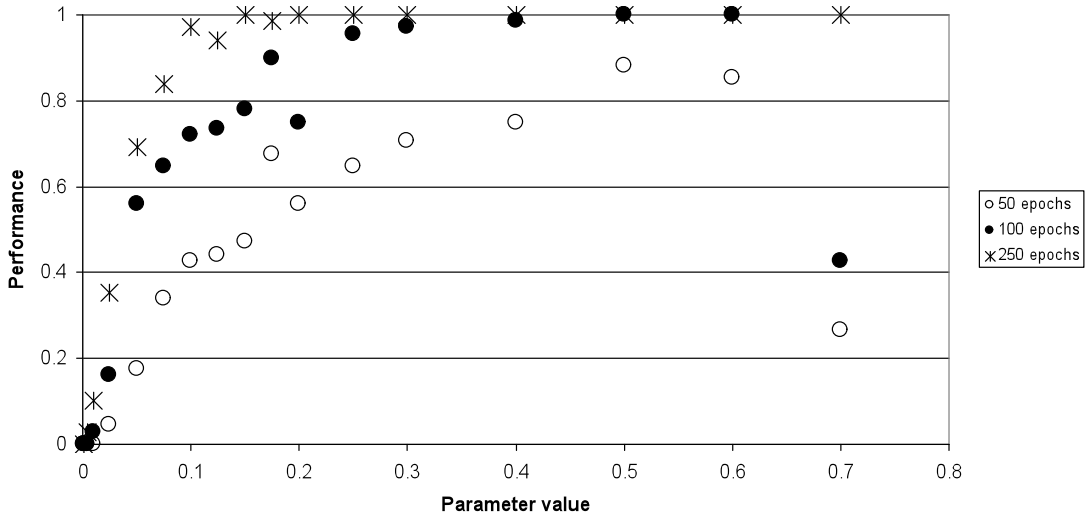
Unit threshold function (temperature)



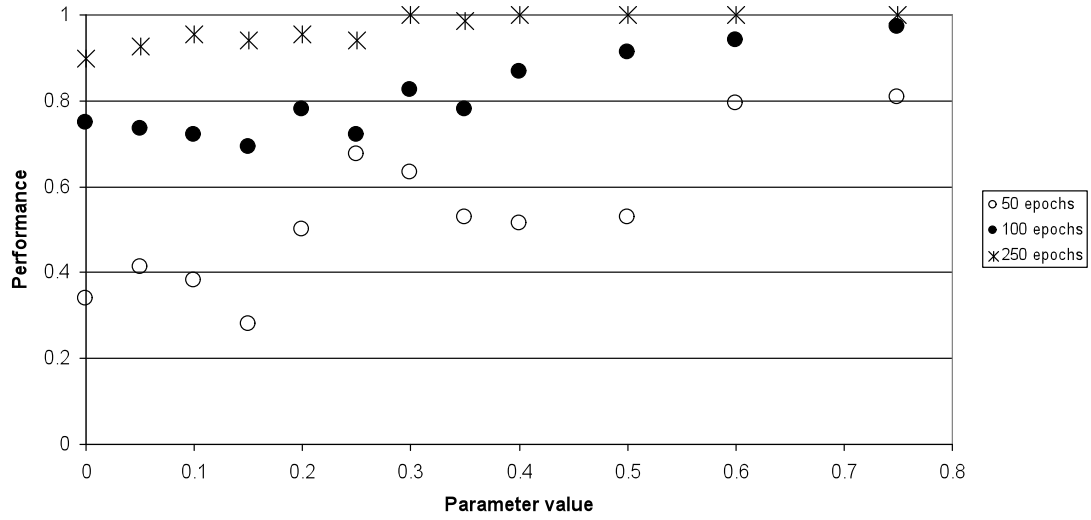
Processing noise:



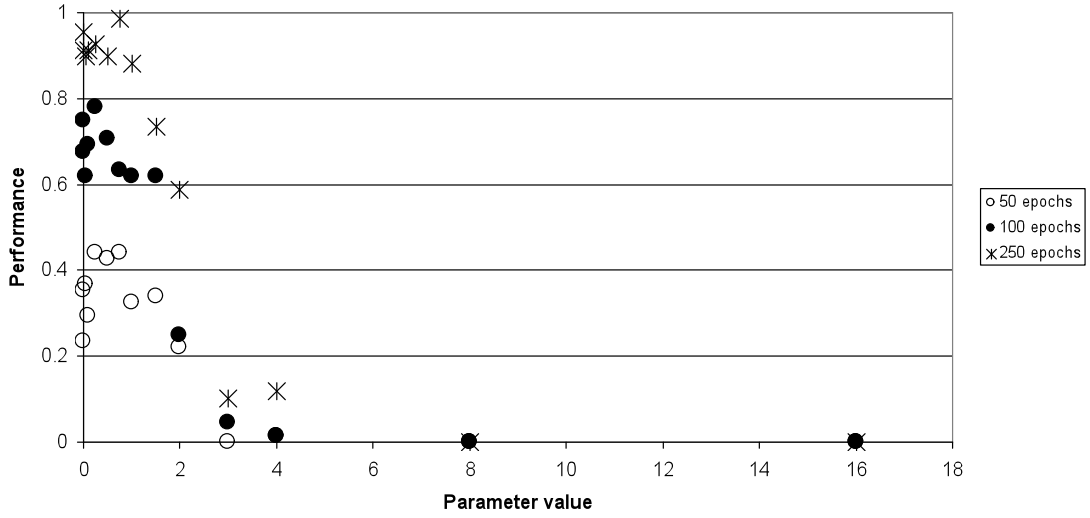
Learning rate:



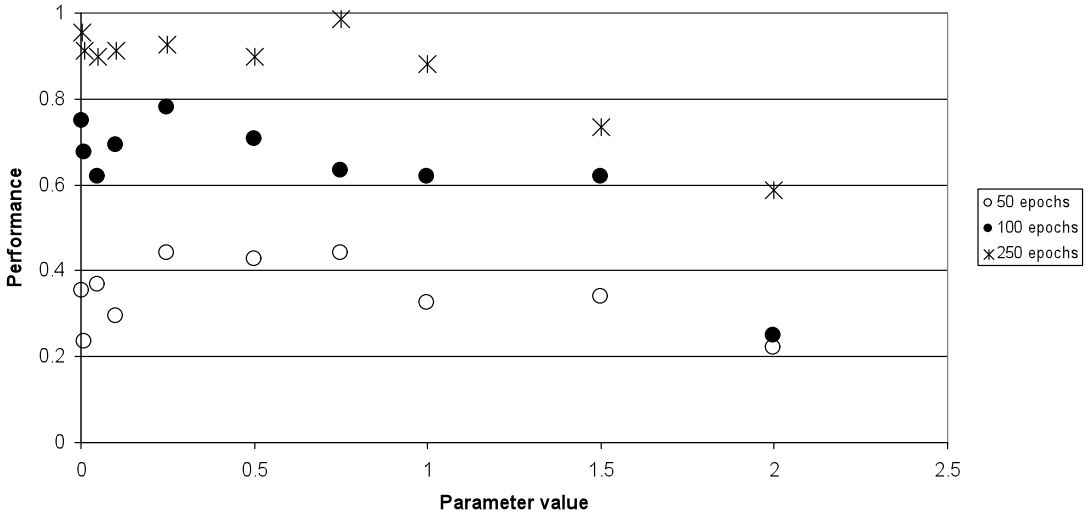
Momentum:



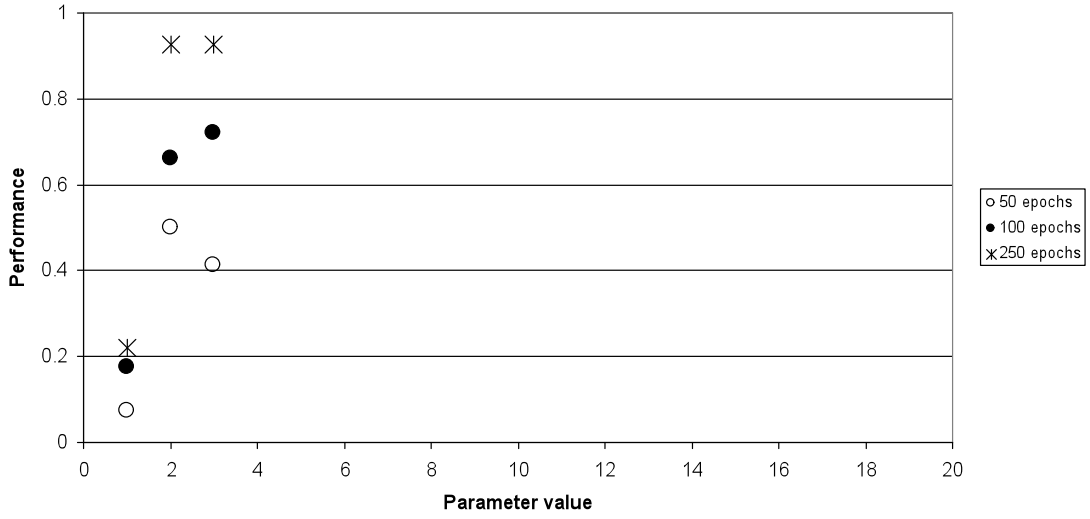
Initial weight variance???



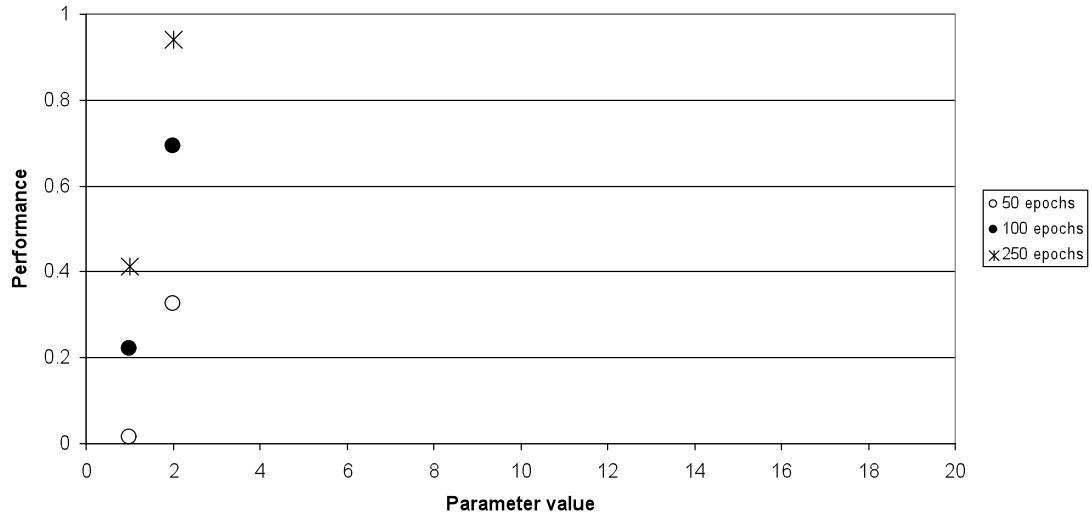
Initial weight variance????



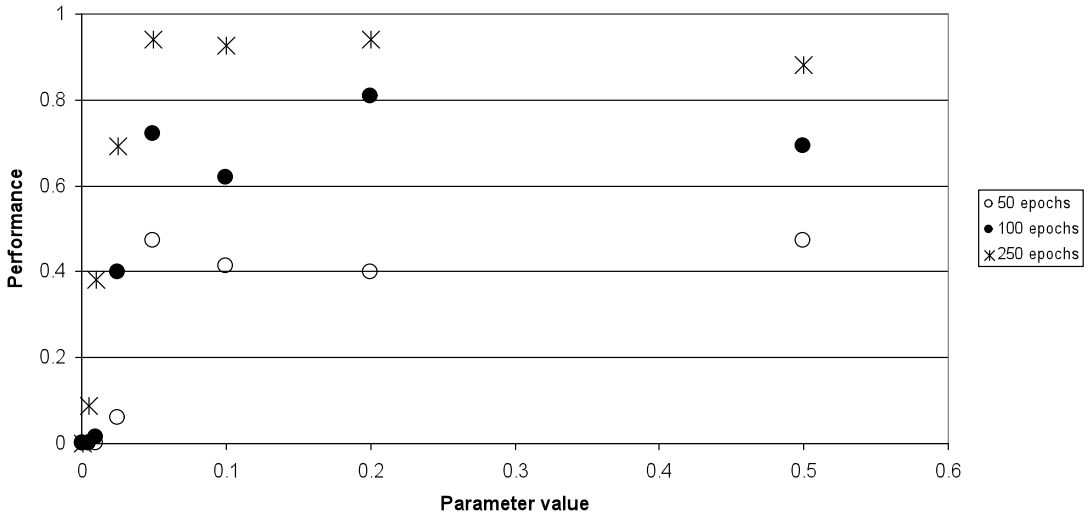
Architecture (0=2-layer; 1=3-layer; 2=fully connected):



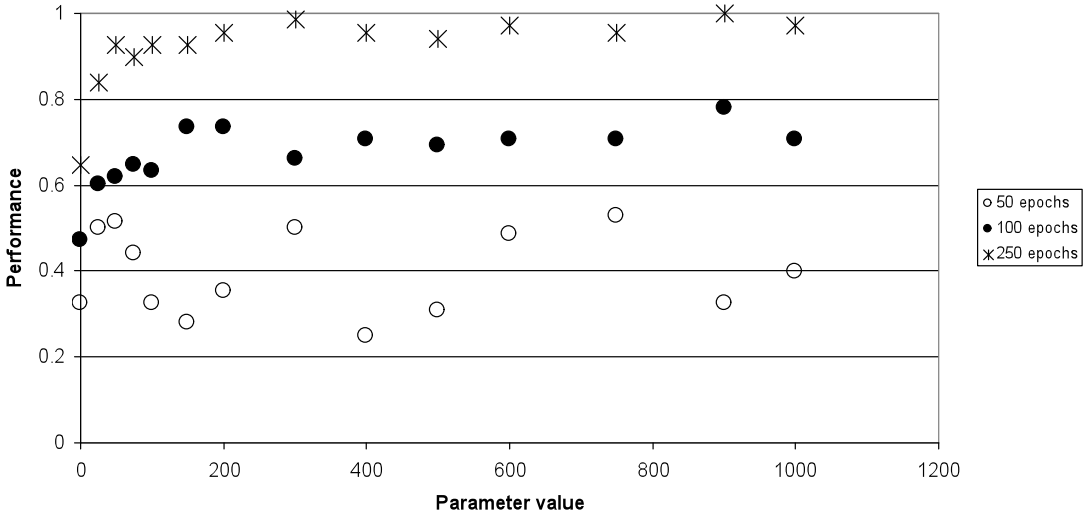
Learning algorithm error metric (0=Euclidean distance; 1=Cross-entropy):



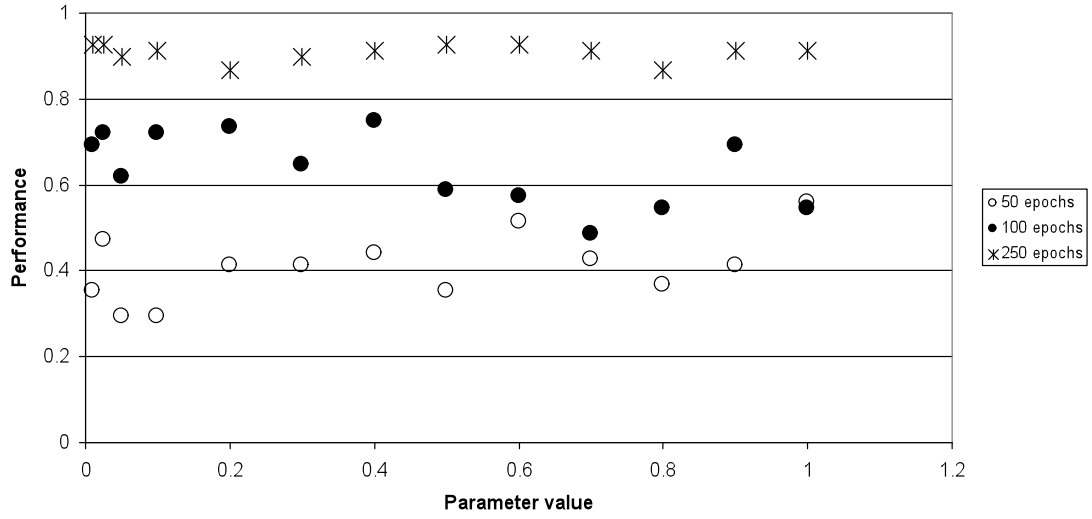
Nearest neighbor response threshold:



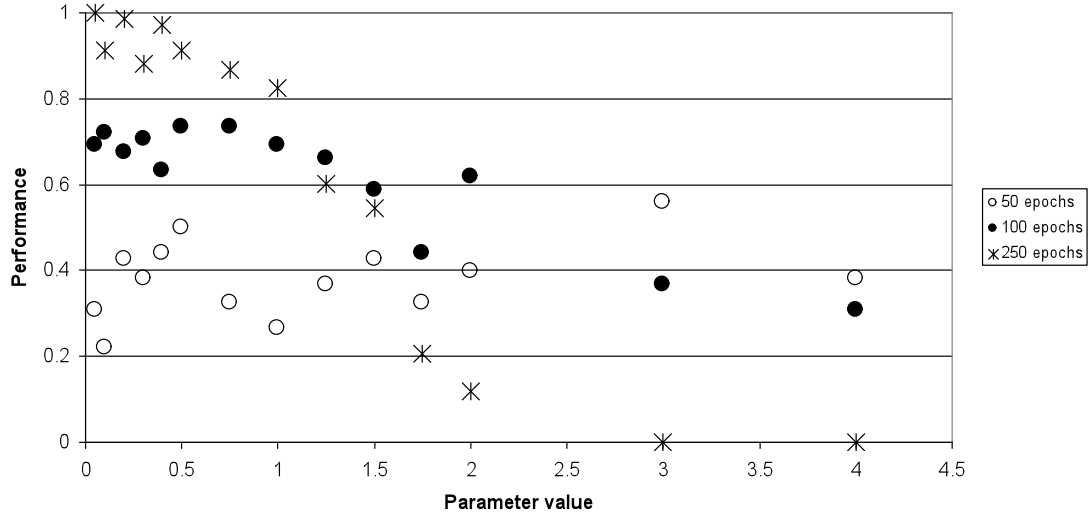
Pruning onset:



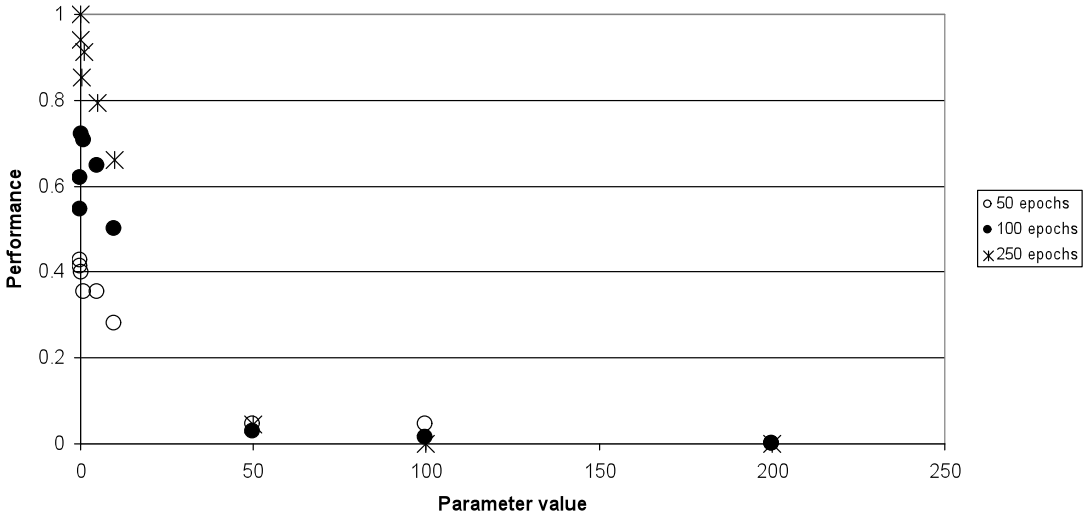
Pruning probability:



Pruning threshold:



Weight decay:



Sparseness of initial connectivity (proportion removed):