https://theses.gla.ac.uk/

Theses Digitisation:

https://www.gla.ac.uk/myglasgow/research/enlighten/theses/digitisation/

This is a digitised version of the original print thesis.

Copyright and moral rights for this work are retained by the author

A copy can be downloaded for personal non-commercial research or study, without prior permission or charge

This work cannot be reproduced or quoted extensively from without first obtaining permission in writing from the author

The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the author

When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given

Enlighten: Theses
https://theses.gla.ac.uk/
research-enlighten@glasgow.ac.uk

Connectionist and Process Modelling of Long-Term Sequence: The Integration of Relative Judgements, Representation and Learning

Kristina Mary Doing Harris, M.A., M.Sc.(Glasgow)

Submitted for the Degree of
Doctor of Philosophy

Department of Psychology
University of Glasgow

February 1995

ProQuest Number: 10992059

All rights reserved

INFORMATION TO ALL USERS
The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript
and there are missing pages, these will be noted. Also, if material had to be removed,
a note will indicate the deletion.

ProQuest 10992059

Published by ProQuest LLC (2018). Copyright of the Dissertation is held by the Author.

All rights reserved.
This work is protected against unauthorized copying under Title 17, United States Code
Microform Edition © ProQuest LLC.

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

Thesis
10251
Copy 1

GLASGOW
UNIVERSITY
LIBRARY

Abstract

A large amount of psychological research is devoted to the representation of sequences. It is a fundamental upon which most of the processes of cognition are based. Despite the amount of research into sequencing, there has been relatively little investigation of the types of representations generated in response to sequential information. These representations must allow operations to be performed on individual elements, as well as operations between and among elements. This thesis begins by describing the effects found when subjects are asked to make relative order judgements using sequences which are in long-term storage (e.g. the alphabet or number series). These effects are then used to examine some of the theories and models which have been developed, with a view toward generating a general purpose mechanism with the ability to model all of the different effects found with different types of stimuli. In the course of developing the new model, the neuropsychological findings in the area are examined in Chapter Two. Deficit studies and neuropsychological investigations are able to isolate which aspects of a task appear to be processed in different structures. If a patient loses the ability to perform one aspect of a task but not another, trying to model both of these aspects in one network may be counter-productive. The construction of a new model is begun in Chapter Three. This model is developed in the PDP environment as it offers the ability to change (learn) as a result of experience, and demands a more thorough definition of the mechanisms operating within the network. Chapter Four details a formal definition of the Serial Order Network (SON) model outlined in Chapter Three, including a section devoted to relative order judgements, called the Response Generation Network (RGN) and undertakes a comparison between the SON/RGN and Poltrock's (1990) random walk model described in Chapter One. A review of some of the sequence learning networks developed is undertaken in Chapter Five. This review is used to choose sequence learning networks, which may be used to learn the type of representation needed. These sequence learning networks are investigated in Chapter Six for their ability to learn the sequence incrementally. It is determined that not one of these networks is appropriate. Thus, in Chapter Seven a recitation mechanism is added directly to the representation in the SON. The resulting system is investigated, and it is determined that the system's success in recitation is not dependent on an idiosyncratic setting of the parameters in the network. The definition of the SON and complementary recitation network is not sufficient. The resulting mechanisms should also be compatible with the developmental literature for both children learning sequences for the first time, and

adults learning novel sequences. A review of this literature is conducted in Chapter Eight. It is also necessary to explain how this SON representation can be developed, and how the model can be used to explain the seeming hierarchic nature of some sequences. In Chapter Nine a mechanism designed to mimic a hierarchical structure for the representation of a sequence is developed. A learning mechanism is defined for the resulting system. This system is then investigated for its ability to recite both hierarchic and non-hierarchic sequences, and to generate the relative order and developmental effects referred to in Chapters One and Nine. The model developed in this thesis is the only model existent which is able to explain sequence learning, representation and relative order effects, and represents an advance in the approach to modelling sequence information.

CONTENTS

TABLES

Number	Description
23.	The SON Weight Structure Used in Simulation One.
24.	The SON Weight Structure Used in Simulation Two.
25.	Medium Parameter Values.
26.	Weight Configuration Used in Initial Observation.
27.	Weights in the Sequence SON Portion of the Network for Block Investigation.
28.	Weights in the Word SON Portion of the Network for Block Investigation.
29.	Interaction of Number of Epochs Before Queue Formed and the Inhibition Returned from the Competitive Filter, with in_eps = 1.
30.	Interaction of Number of Epochs Before Queue Formed and the Inhibition Returned from the Competitive Filter, with in_eps = 3.
31.	Interaction of Number of Epochs Before Queue Formed and the Inhibition Returned from the Competitive Filter, with in_eps = 10.
32.	Medium Parameter Values.
33.	Training Patterns used for Word to Sequence and Word SON weights.
34.	The Weight Structure Resulting from the Training Patterns in Table 49.
35.	Training Patterns used for Word to Sequence and Word SON weights in Blocked Sequence.
36.	Medium Parameter Values.
37.	Relative Order Effects to be Modelled.
38.	Medium Parameter Values.
39.	Epochs to Solution for Each Comparison at Separation 1 in a Blocked Sequence.
40.	Comparisons at Separation 1.
41.	Reduction in Epochs for Each Minimum Stimulus Across Separation 1 and 2.

Number	Description
1-1.	Pretraining File without Recurrence (pretrain.pat).
1-2.	Fully Recurrent Auto-Associative Training File (entire.pat).
1-3.	Semi-recurrent Training File (entire2.pat).
1-4.	Semi-recurrent Training File (entire3.pat).
1-5.	Fully Recurrent Training File (entire4.pat).
1-6.	First Subsequence Training File (part1.pat).
1-7.	Second Subsequence Training File (part2.pat).
1-8.	Third Subsequence Training File (part3.pat).
1-9.	Fourth Subsequence Training File (part4.pat).
2-1.	Weights in the Sequence SON Portion of the Network for Block Investigation.
2-2.	Weights from Sequence Layer to the Word Layer for Block Investigation.
2-3.	Weights from the Word Layer to the Sequence Layer for Block Investigation.
2-4.	Weights in the Word SON Portion of the Network for Block Investigation.

FIGURES

Number	Description
1.	Analog Representation.
2.	Sample Random Walks.
3.	Divisions of the Mechansisms implicated in the Use of Numbers According to McCloskey and Caramazza, 1985.
4.	Subdivisions Within the Model of Number Use by McCloskey, Caramazza and Basili, 1985.
5.	Dehaene's (1992) Number Processing Model.
6.	Fragment of the SON model.
7.	Activation Values from First Five Epochs.
8.	Activation Values from the First Five Epochs, with Initial Node One.
9a. and 9b.	Epochs to Maximum Activation Value for Networks with Initial Nodes Three (a) and One (b).
10a. and 10b.	Average Activation Differences for Nodes in Networks with Initial Nodes Three (a) and One (b).
11.	Activation Values Over the First Five Epochs in a Network, with Initial Node Three.
12.	Activation Values Over the First Five Epochs in a Network, with Initial Node One.
13.	Activation Values Over the First Six Epochs, with Initial Node One.
14.	Activation Values Over the First Five Epochs, with Inital Node Three.
15.	Approximation of Parkman's Data.
16a. and 16b.	The Average Activation Change (a) and Epochs to Maximum Activation (b) in Simulation One.
17a. and 17b.	Average Change in Activation (a) and Epochs to Maximum Activation (b) in Simulation Two.
18a. and 18b.	Average Change in Activation (a) and Epochs to Maximum Activation (b) in Simulation Three.
19.	SON Augmented by a Response Generating Network.

Number	Description
20.	Activation Values in the SON/RGN for the First Five Epochs.
21.	Comparison of Error Rates.
22.	RTs Predicted by SON/RGN v RTs Observed by Poltrock.
23.	Average RT Observed by Parkman v Average Epochs to Decision in SON/RGN.
24.	RTs Predicted by SON/RGN v RTs Obtained by Parkman.
25.	RTs Predicted by SON/RGN v RTs Observed with 10% to 20% Errors.
26.	RTs by Separation when Errors are Made on 5%, 15% and 25% of Trials.
27.	Forward and Backward Networks Connected Through a Gating Network.
28.	SON Designed to Explicate Structure of Alphabet.
29.	The Competetive Queue Model Described by Houghton (1990).
30.	Configuration of the Competetive Filter in the CQ Model.
31.	Sigmoid Activation Function used in Most Backpropagation Networks.
32.	Activation Values Across Hidden Layer Units.
33.	SON/CQ Extension of the SON Model.
34.	Heirarchical SON Structure.
35.	Structure of Recitation System with Heirarchical SON.
36.	Modified Response Generating Network Attached to SON.
37.	Comparison of Results of Final Network to an Approximation of Parkman's Findings.
38.	SDE Curve Obtained from a Network with Pre-set Weights Designed to Mimic a Blocked Sequence Training Regime.
39.	SDE Found with Fully Interconnected Sequence Training Regime.
40.	Change in SDE Curve Across Learning Rates.
41.	SDE Curve for Network Trained on a Blocked Sequence.

ACKNOWLEDGEMENTS

I would first and foremost like to thank my supervisor Prof. Tony Sanford, without whose help and patience this would not have been possible.

Secondly, I would like to acknowledge the immense contribution of Dr. Mark Harris, whose support both mental, and occasionally physical, kept me functioning throughout the work on this thesis, the years before it and, without doubt, for many years to come.

Thirdly, I would like to thank Park and Erica Doing, for want of whose support I would not only not be where I am today, but would not even be on this planet.

Finally, I would like to thank all of the friends and colleagues whose advice and sociability were a great help to me. These include, but are not limited to: Mr. Paddy O'Donnell, Dr. Linda Moxey, Prof. Mike Burton, Gillian Ferrier, Melody Terras, Dr. Matt Traxler and Lisa Frenz. I hope anyone whose name I have not mentioned will forgive me.

Chapter One - Behavioural Research: Findings, Theories and Implementations

Although there has been much research into the interpretation and use of sequentially presented material, this research appears to neglect what happens to a sequence once it has been learned i.e. how the representation of that sequence can be used to make decisions about the sequence elements. This apparent neglect is understandable since most of these systems are designed for the processing of language. They are concerned with the development of an overall interpretation of sequentially presented material. To retain the actual elements presented would be a problem as experimental evidence shows that this is not done while language is being processed.

Some sequences, however, are designed to be retained in the form in which they are presented. This type of sequence includes simple sequences such as the number series or the alphabet, and more complex sequences such as poems or speeches. When one examines subjects' ability to make order decisions using the elements of these sequences, one sees that the language interpretation networks thus far developed are not sufficient to model them.

For the purposes of this thesis the retention of the actual sequence elements has been termed long-term sequence learning. By contrast the extraction of meaning is considered to require only short-term retention of the sequence elements.

This thesis explores the issues related to the representations developed in systems designed to retain sequences as they are encountered.

This chapter examines the findings of behavioural research into relative order judgements and the effects on reaction time and errors associated with them. It goes on to outline the theories and implementations that have been generated to explain these effects. The goal is to combine this information with the neuropsychological findings in the next chapter and use this

combined information to design an implementation that will be able to explain these effects and which also has the capacity to explain the sequence learning effects, which are addressed in Chapter Five.

A. Relative Judgement Effects

The development of a theory of the long-term representation of a sequence requires an understanding of the types of phenomena generally associated with these representations. The two most common tasks used to elicit phenomena related to the representation of a sequence are recitation and relative order judgements. Sequence recitation is the more frequently investigated and modelled task. Therefore it seems likely that a model appropriate to this task can be generated or found. The same cannot be said for relative order judgement tasks. Therefore this thesis begins with an exploration of the phenomena elicited in relative order decision tasks, the theories generated to explain them and the implementations of these theories, with a view to finding or generating an implementation of this task. A similar exploration using the sequence recitation task is carried out in Chapter Five.

1. Numeric Stimuli

The most important and reliable effect found with relative-order judgements of numeric stimuli, is that subjects require more time when the magnitude values of the stimuli are closer. This has been termed the Distance Effect (noted in Parkman, 1971; Potts, et al., 1978; and Itsukushima, Tozawa and Itagaki, 1990; among others). It has been thought to be analogous to similarity effects obtained with psychophysical stimuli.

Parkman investigated numerical comparisons in his 1971 paper which consisted of two experiments. In the first experiment, he asked subjects to choose the larger of two numbers (both between 1 and 9) and found that variance in RT was affected substantially by the minimum stimulus of the pair (Min, 72%), but not by the

distance between the two stimuli (Distance, 9%), the maximum stimulus of the pair (Max, < 9%) or the sum of the two stimuli (Sum, < 9%). Digit pairs with Distance values of one, two or three caused systematically higher residual RTs than did digit pairs with Distances of between four and nine. He concluded that the RT by Distance curve could be thought of as having a linear component due to Min and a non-linear component due to Distance. No order effects due to left-right position of Max and Min (direction effects) were found. The subjects demonstrated low error rates, highly positively correlated with RT.

Parkman's second experiment attempted to replicate findings by Fairbank (1969); i.e. that it took less time to find the larger than smaller digit, and that digit pairs containing zero had higher mean latencies than would have been predicted, especially the pairs (one, zero) and (zero, one) [Zero-term effect]. The results for this experiment showed that Min still accounted for a substantial proportion of the variance in RTs and that the slope values for the Min were similar to those in the first experiment. A clear non-linear trend was expressed when Min residuals were plotted as a function of Distance. Subjects in Experiment two took significantly more time to indicate smaller digit than subjects in Experiment one, both with and without zero pairs.

Parkman (1971) also described the findings of Restle (1973). Restle's task was a comparison between the sum of a pair of digits and a third digit. The stimuli are presented in the form of two stimuli on one side of the screen (p and q) across from the third (r). In this mental addition task, RTs monotonically decreased with increasing distance between the sum of p and q, (p + q), and (r) and that latencies decreased monotonically as the distance between p and q increased.

In summary, Parkman found that RTs demonstrated a linear effect of Min and a non-linear effect of distance, with no direction effects. Choosing the larger of two stimuli was faster (Instruction effect) and pairs containing zero caused higher than expected RTs (Zero-term effect), see table 1.

Banks' (from Potts, et al., 1978) review of the number comparison literature indicated the existence of three interrelated effects. These were the Distance effect, End-term effect and Congruence effect. The End-term effect is closely related to Min and refers to faster RTs on pairs containing at least one term that is at or near one of the extremes of the ordering, in this case 1 is taken as the extreme of the ordering.

Further experimentation by Itsukushima, Tozawa and Itagaki (1990) involved the investigation of the representation and retrieval of two-digit numbers, using three investigation techniques :

(1) The direct comparison of pairs of single-digit numbers requiring larger/smaller decisions,
(2) Memorising a digit (called the Standard) and deciding if a presented digit is larger/smaller, and
(3) Equations in the form (p+q, r) - the decision required being whether p+q is larger/smaller than r, (p, q and r can be one or two digit numbers).

They determined that the technique least likely to produce response bias was technique (2). Response bias refers to the subjects' increased likelihood to respond in a certain direction. For example, when a one is presented subject's respond smaller without regard to the second stimulus. Technique (2) allows an equal number of larger and smaller decisions to be made, using the same stimulus. On experimentation using technique (2), they found a categorisation effect, i.e. RTs varied systematically with the tens-place digit, but only in decades which did not contain the Standard. However, this effect is contested by Hinrichs (1981), who did not find it.

Itsukushima, et al. also found that processing within the decade containing the Standard was slower than in other decades. They propose that the tens-place digit functions as a category marker after which analog information must be used. Their subjects produced low error rates which were centred on the standard decades. They also produced a reliable distance effect, with no

effect of choose larger versus choose smaller, when the standard was 25 and the comparators were from the 10's, 20's and 30's decades. The profile of RT fit well to the log distance model of psycho-physical function, with RTs outwith the standard decade almost constant against the position of two-digit numbers. When the standard was raised to 75 and comparators were from the 60's, 70's and 80's decades, significant main effects of distance and judgement were found, with a significant interaction between them.

Overall smaller judgements were faster than larger, but an interaction occurred because the 60's and 80's judgements for larger were faster than smaller, although the 70's (standard decade) smaller judgements were faster. Differences of mean RTs between the 60's and 80's decades were not significant, but were greater than in the "standard = 25" condition. Another result was that smaller judgements in the 75 condition were faster than in the 25 condition, whereas larger judgements in the 75 condition took longer than in the 25 condition. They explain that "the discrimination of difference between numbers becomes difficult as the absolute magnitude of the numbers becomes larger resulting in what many called the 'Numerical Compression Effect'" (Itsukushima, et al., 1990).

The effects found can be summarised as no significant effects of distance on RT outside the decade containing the standard (only the Categorisation Effect influenced RTs), but general DE found within that decade. The smaller stimuli produced faster RTs overall (Numerical Compression Effect). The effect of decision type varied across conditions, with smaller judgements faster for small stimuli and larger judgements faster for large stimuli (Congruence Effect), except inside the standard decade of the large stimuli, where smaller decisions were faster. The effects found are summarised in table 1.

Related effects using numeric stimuli configured in a different manner, are reported by Morin, DeRosa & Stultz (1967). They describe a DE, when subject's task was to memorise a set of digits

and then decide if a probe digit was present or absent from the memorised set. RTs for probes absent from the set decreased with the distance of the probe digit from the set. Dehaene (1992) reports that Morin et al., found a DE on present RTs. When the set was made up of consecutive digits, digits on the boundaries caused longer classification RTs, than digits in the interior of the set.

Jaffe-Katz, Budescu and Wallsten (1989) also conducted experiments involving paired comparisons on numerical and non-numerical expressions of uncertainty. In their experiment the instructions were varied across subjects (choose larger versus choose smaller). They found that numerical stimuli were faster than non-numerical, in both conditions, with consistent Distance and Congruence effects. "..illustrating that both numerical and non-numerical expressions of uncertainty contain subjective magnitude information and suggesting that similar processes are employed in manipulating and comparing numerical and verbal terms"(Jaffe-Katz, et al., 1989).

Potts et al. (1978) noted that "..there is a great deal of independent evidence for the notion that people have available to them a non-linear psychological scale of number, one that gives a quick intuitive 'feel' for the size of a number". In their review they explain several lines of evidence on this point, based on various methods of obtaining intuitive judgements of numerical size, with some recent evidence from Shepard et al. (1975). They also point out that semantic congruity is a reliable effect (Banks et al., 1976).

The effects on RT found during numerical comparison judgements and related judgements are outlined in table 1. This includes both single and multi-digit comparisons.

1. Intuitive 'feel' for the size of a number.
Single Digit
2. Linear effect of Minimum Stimulus (debated).
3. Non-Linear Distance Effect (DE).
4. Larger Decisions Faster for digits 0-9.
5. Zero-term Effect (Zero pairs slower than expected).
6. End-term Effect, if 1 taken as the end-term.
Multi-Digit
7. Categorisation Effect outside Standard decade (debated).
8. DE within Standard Decade.
9. Numerical Compression Effect.
10. Congruence Effect, except inside Standard decade.
11. Smaller Decisions faster inside Standard decade.

Table 1. Summary of Relative Judgement Effects for Numeric Stimuli.
(1) is from Potts, et al., 1978. The single digit effects are taken from Parkman (1971) and Potts, et al., (1978) and the multi-digit from Itsukushima, et al. (1990).

2. Alphabetic Stimuli

Any model of sequence learning and relative judgement, must be able to address the findings of experiments involving alphabetic and other sequences as well as those involving numeric stimuli.

Investigation in the area of alphabet comparisons was widely done throughout the 1970's. Researchers found the same decrease in RT with increasing separation between letters as with distance between numbers. Since this effect appears to apply to stimuli in which measuring Distance has little meaning, the name was expanded to the Symbolic Distance Effect [SDE]. As the interest in no longer solely on distance, from now on reference will be to the Separation between stimuli, instead of the Distance between them.

In one of the earlier studies, Lovelace and Snodgrass (1971) found SDE and a direction effect in letter-pair comparisons. The SDE contained a discontinuity between separation values of one (e.g. pair A B) and two (e.g. pair B D), at which point RTs increased instead of decreasing. Interestingly, this same discontinuity appears in a graph from Poltrock (1989), which shows subjects' RTs, while being asked to respond in approximately 300 msec, to a digit comparison task. The Direction effect took the form of increased RTs for backward pairs (non-alphabetic order), this effect interacted with SDE in that the direction effects decreased with increasing separation. They also found a Compression effect with letters i.e. slower for pairs near end and a Congruence effect for pairs containing Z (backward faster). Some subjects reported using rough position tags for letters i.e. "middle of the alphabet" or "near the end" as a basis for responding when separation was large. These effects are summarised in table 2.

Parkman (1971) used alphabetic stimuli for comparison to digital stimuli in decision tasks. Using the letters A through J and K through T, he found increased latencies over numbers, with the minimum stimulus no longer accounting for as substantial an amount of mean RT variance as for digits (actually accounting for almost no variance). His results showed that the separation between the letters accounted for approximately half the variance in the two groups (A-J and K-T). The maximum stimulus accounted for a substantial amount of the variance for the second 10 letters, (K - T). This group also had slightly increased RTs (Compression effect) and error rates. When letters were shown in alphabetical order (forward direction) RTs were, on average, faster, although not significantly, for both groups. The effects he noted are summarised in table 2.

1. Symbolic Distance Effect (SDE).
2. Discontinuity in SDE curve, between 1 and 2 (debated).
3. Direction Effect (alphabetic order is faster).
4. Interaction between Direction and Separation (Direction decreases with Separation).
5. Compression effect.
6. Congruence effect with pairs containing Z.
7. Rough position tags used.
8. Alphabetic comparisons slower than Numeric.
9. Maximum stimulus linearly related to RT for (K-T).
10. (K-T) slightly slower, with more errors than (A-J).
11. True/False judgements of correct order were slower than Larger and Smaller decisions.
12. An interaction between end-term and judgement only occurred in correct order judgements.

Table 2. Summary of effect associated with Relative Judgements for alphabetic stimuli. Numbers 1 through 7 taken from Lovelace and Snodgrass (1971) and 8 through 10 from Parkman (1971). The final two are taken from Polich and Potts (1977).

An important finding was made by Polich and Potts (1977) when they examined the role of instructions in letter comparison studies. They found that if subjects were asked to decide if letters had been presented in the correct order they responded consistently slower than when subjects were asked to identify the larger or smaller of the two letters. The interaction between end-terms and questions also changed with instructions. True/false questions produced an interaction while identification questions did not. They suggested this indicated that subjects had a variety of strategies available to them for dealing with a particular task and that even seemingly minor procedural variations could lead to subjects altering their choice of strategy. These effects are also listed in table 2.

Relative judgements of alphabetic stimuli have revealed a phenomenon termed 'run-through'. This is the need for sub-vocal recitation of a portion of the sequence before the element of interest can be located. For example, in response to the question: "Are the letters J Q in the correct order ?" a subject may report thinking: " H - I - J - K -L - M - N - O - P - Q - R, the answer is yes". Hamilton and Sanford (1978) reported that the frequency of run-through was linearly related to the separation between the letters and that RT was linearly related to run-length. On trials in which run-through did not occur RTs were not significantly different regardless of separation, except for a separation of 1. Their findings are summarised in table 3.

1. RT linearly related to run-length.
2. No SDE on trials without Run-Through, except separation one.
3. Frequency of Run-Through linearly related to Separation.
4. Start of Runs related to preferred entry points.
5. No clear relationship between error frequency and Separation.
6. Preferred entry points are related to subjective groups.
7. 14% of Runs lead away from second stimulus.
8. 27% are 'Gap Closure' Runs.
9. 84% of non-target initiated Runs began with first letter of a subjective group, when embedded groups included.
10. Pauses in recitation indicated subjective groups.
11. Run-Through speed appeared under conscious control.
12. Runs faster on After trials.
13. Frequency and average length of Runs higher for Before trials, as were errors.
14. RT longest when Before/After trials crossed group boundaries.

Table 3. Summary of Run-Through Effects. Numbers 1, 2 and 3 are taken from Hamilton and Sanford (1978). While the remainder are from Hamilton (1979).

Hamilton (1979) expanded on these findings, by producing an in-depth study of run-through in relation to RT. She reported an interaction between separation and the order of letters to be judged (direction). Her subjects took more time to determine that letters were in the incorrect order, with this time increasing when letters were closer in the original sequence. She found that run-through could begin with any letter, but appeared to be related to preferred entry points into the sequence. Some letters were used significantly more often as the first letter of a block, during run-through. Hamilton postulated that these letters represented the starts of sub-sequences used by subjects when learning the sequence. She also found that only slightly more than half of all runs contained both letters of interest. Her findings demonstrated no clear relationship between error frequency and letter separation. No runs of length 1 were reported.

Run-through did not always take the same form. Hamilton records several different types. Trials in which the direction of run-through led away for the second stimulus letter (i.e. pair 'R P', run 'R S T U V') accounted for 14% of trials, irrespective of letter separation or alphabetic ordering of the pair. The explanation that Hamilton offered was that when run-through proceeded away from the second stimulus item it may have been serving to increase the distance between first item and items in memory, until a response could be made. Runs which included only one letter of the stimulus pair, leading toward the second letter but not reaching it, were termed "Gap closure" runs, with 27% of trials showing this type of run. Again, run-through may have served to increase the distance between stimuli, such that a decision could be made before the second stimulus was reached. 49% of runs did not start with a stimulus letter, 58% of these began with letters which were at the start of subgroups, the remaining 42% at other points in the sequence. Of the 51% of runs which began with a stimulus letter, 42% were also the start letter of a subgroup. Closer scrutiny of the unaccounted-for starting points revealed that they were mainly explained by the consistent use of one different letter by three subjects. Hamilton explained this by theorising that groups of letters, overlapping the original subjective groups, were learned as

the alphabet was used over the years. She called these "embedded groups". When embedded groups were included, it was determined that 84% of non-target initiated runs began with the first letter of a subjective group. Starting letters which did not begin subjective groups were used with very low frequency, perhaps once or twice for each subject throughout the entire experiment.

As further evidence for the role of starting points in run-through, she showed that pauses in recitation of the alphabet could be used to indicate a subject's grouping structure and were indicators of groups used in learning the sequence. For trials in which subjects were asked to report the next item in the sequence, runs began at a point prior to the target letter and continued until the target was reached, subjects then answered with the next letter (After trial), Before trials were the same except subjects answered with the preceding letter. In this type of experiment subjects produce more runs allowing deeper investigation. Some runs began in the wrong place and were abandoned and started over. She felt that the speed of run-through may be under conscious control, since runs of the same length were produced during trials with different RTs. Subjects ran-through items faster in response to an After query than a Before query, even though in both cases runs were forward, but the frequency and average length of run-through were significantly higher on Before trials, where 63% of trials showed runs with an average length of 3.91 items, than on After trials, where subjects produced runs on 35% of trials and the average length was 2.96 items. There were also significantly more errors on Before than After trials. Trials on which the stimulus and response items fell across one of the group boundaries, had the longest latencies associated with them, in both directions (After last item and Before first).

The overall run-through effects are summarised in table 3. These run-through findings should be examined in relation to counting in children, because it may be that counting and run-through as solution strategies require the same type of underlying mechanism.

3. Other Stimuli

SDEs have been reported with a variety of procedures, and a variety of materials, such as animal names, conceptual size (Moyer, 1973; Pavio, 1975), linear orderings (Potts, 1972) and artificially generated stimuli (Kosslyn and Pomerantz, 1977; Moyer and Dumais, 1978; Potts, et. al., 1978).

Audley and Wallis (1964) described two relative judgement effects found when relative brightness/darkness judgements are made, the Funnel effect and the Cross-over effect. The Funnel effect is distinguished by RT differences of zero at the lowest level of the stimulus dimension and maximum at the highest level, for the two decisions (brighter than/darker than). That is, deciding whether one stimuli is brighter than the other takes the same amount of time as deciding which is darker when both stimuli are dim, but deciding which is brighter is faster when both stimuli are bright, with the difference between the two increasing as the stimulus intensity increases. The Cross-over effect is when stimuli at one end of the stimulus dimension are more quickly related to that end point. For example, one of two dark stimuli was determined darker quicker that brighter and one of two bright stimuli was determined brighter quicker than darker. This effect has since been termed the semantic Congruence effect. The Cross-over/Congruence effect appears to be unaffected by practice.

In another study, Jamieson and Petrusic (1975), discovered that preferred colours were selected more quickly when subjects were asked which colour they preferred and the opposite was true when they were asked which colour they preferred less. This is the semantic Congruence effect.

Holyoak and Walker (1976) looked at three methods of predicting RTs for relative judgement decisions, using stimuli ordered along a semantic dimension i.e. day-week, average—fair. They examined three dimensions along which the stimuli were measured.

(1) rated distance - subjects were shown pairs of words and asked to rate on a seven point scale the distance between them.

(2) grouping - subjects were asked to divide the words into groups according to their similarity in meaning.
(3) ordinal distance - words were placed in increasing/decreasing order. Subjects were asked at the beginning of each section if they agreed with the orderings.

The rated distance scale was used to examine if an underlying analog scale was used for making judgements. While the grouping and ordinal scales where to examine of stimuli were compared in groups or along an ordinal scale.

They examined the effects found for alphabet and numerical stimuli and found that end-terms were not compared more quickly (no End-term effect) and that the order of terms, in pairs for judgements, did not effect results (no Direction effect). For time measures (millisecond to millennium), rated-distance was a better predictor of RT than grouping or ordinal distance. Error rate was very small and almost all were made on adjacent pairs. For quality measures (perfect to awful) the most decision time was taken by the (fair, average) pair. Again, rated-distance was the best predictor of RT, and there was a low error rate, almost all errors were on adjacent pairs. Temperature scale (torrid to frigid) comparisons seemed to follow a categorical effect, i.e. cold terms compared to hot always had faster RTs than cold to cold or hot to hot. However, since people predicted three groups and not two for these terms, rated distance was still the best predictor of RT. There was the same error pattern, except there were no warm/cool errors. Adjacent terms always produced the longest RTs. There were no cross-group fast RTs, and semantic Congruence effects occurred.

In summary, rated distance was the best predictor of RT for all stimuli examined, which indicates that the underling comparison dimension is analog in nature. No End-term or Direction effects were found, but the Congruence effect was apparent.

Believing that interval data was used in relative order decision, but wishing to determine if interval (perceptual) information was used where available, Moyer (from Potts et al. 1978) conducted an

experiment in which subjects were shown circles of differing sizes, each size of circle was given a name. In one group the size difference between adjacent items was large in the other it was small. Every name pair from the large-range condition was compared more quickly than those from the small-range condition, even though ordinal position was held constant. He further carried out detailed tests, which ".. appeared to force the conclusion that subjects retrieved more than ordinal information about the symbols that they compared, and the parallel to the perceptual data cannot be ignored" (Potts, et al., 1978). This supports the idea that both kinds of information are used. Moyer also notes that data from memory experiments shows the opposite of the Compression effect, pairs from the 'large' end are compared quicker. He concludes from this that values for the stimuli must first be retrieved in a memory task, but are given when a perceptual task is performed. Value and stimulus are, therefore, the same in perceptual tasks.

Potts (et el., 1978) list experiments which have generated evidence that subjects construct a linear ordering of items in memory, not simply quantitative information about each item. When presenting subjects with pairs of items with which to construct an ordered list, Smith and Mynatt (1977) showed effects in processing time of both storage load and reordering of items required when the pairs were not presented in a natural order. These findings supported their view that subjects hold pairs presented in STM until they can be linked together in a linear order. Potts (1972) presented subjects with pairs of terms (i.e. A > B, B > C, C > D). He found an SDE despite the exclusion of non-adjacent pairs from the training set. Barclay (1973) presented adjacent pairs and remote relations in the form of sentences. One group was instructed to remember the linear ordering, the other to remember the sentences verbatim. The linear ordering group was able to distinguish true and false sentences on the basis of the order of the elements, but could not remember which sentences had been presented. The verbatim group could not distinguish which sentences had been presented or discriminate between true and false sentences, and many could not produce an ordered list. Potts

(1972) found that the order of presentation of paired associates was important. The intuitive natural order of presentation was better than end-points first. Smith and Foos (1975) found that subjects were only unable to produce a linear ordering when the last two pairs were switched and when the final pair was first. The construction of linear orderings is addressed further in Chapter Five of this thesis.

The results from relative judgements of non-alphabetic, non-numeric stimuli are summarised in table 4. They consist of: the SDE, Funnel effect, Cross-over effect (which is also called the Congruence effect), most errors on adjacent pairs, no End-term effects, RTs best predicted by rated distance, subjects use interval data as well as ordinal, the opposite of the Compression effect, and the construction of a linear ordering where possible.

1. SDE.
2. Funnel Effect.
3. Cross-Over/Congruence Effect.
4. Errors almost solely on adjacent pairs.
5. No End-term effects.
6. RTs best predicted by Rated Distance.
7. Interval data used in conjunction with ordinal.
8. Opposite of Compression effect.
9. A linear ordering is constructed.

Table 4. Summary of Relative Judgement Effects found with Various Stimuli. They are found with (1) animal names, conceptual size (Moyer, 1973; Pavio, 1975), linear orderings (Potts, 1974) and artificially generated stimuli (Kosslyn, 1977; Moyer and Dumais, 1978; Potts, et al., 1978), (2-3) brightness judgements (Audley and Wallis, 1964), (4 - 6) semantically ordered stimuli (Holyoak and Walker, 1976), (7 & 8) named circles and (9) various stimuli (Moyer and Potts, respectively, both from Potts, et al., 1978).

4. Comparing Stimuli

Examining tables 1, 2 and 4, it is apparent that all stimuli show SDE, but that all other effects appear unique to their stimuli. It is perhaps likely that the reason for these differences is that the experiments using non-numeric stimuli have not examined the same variables as numeric stimuli experiments, but when they have, the differences have been due to the difference in the amount of exposure received to the number series or the differing nature of the stimuli. Parkman used both numeric and alphabetic stimuli. He found SDE for both, but minimum stimulus, Direction, Zero-term and Instruction effects in numbers only and maximum stimulus and a tendency toward the Direction effect in alphabetic stimuli only. The minimum and maximum stimulus effects may be due to the large amount of interaction between separation and Min/Max. stimulus, as the separation increases the pool of stimuli available to represent either the Max or Min decreases, until for a separation value of nine (the highest used by Parkman) the only stimuli available are 9, J, and T for Max and 0, A, and K for Min. This is a problem that Itsukushima et al. (1990) hoped to avoid by using Standard/Comparator trials.

The effect differences that may be due to the exposure discrepancy between the types of stimuli, are the Direction effect (alphabetic), the zero-term effect (numeric) and the Compression effect (either direction). The Direction effect for alphabetic stimuli may be due to the relative inexperience most subjects are likely to have of the alphabet in reverse order, whereas the number series 9 to 1 is commonly taught and used. The zero-term effect of the number series has been postulated to be due to nature of teaching of the number series which does not include zero as often as the other numbers, Ashcraft (1992) reports studies from Hamann and Ashcraft (1987) and Ashcraft and Christy (1991) which examined text books from grades Kindergarten (approximately 5 years old) through Third grade (approximately 8 years old) and Third through Sixth grades (approximately 10 years old), respectively. Facts about numbers containing zero appeared least frequently of all number facts. The same does not appear true of the alphabet. It

seems likely that A is mentioned as often as any other letter and it is probably better compared to one than zero. However, a formal investigation of the alphabet has not been done.

The Compression effect is another that may result from the amount of exposure subjects are likely to have had. The two studies cited above, which examined textbooks used in Kindergarten through Sixth grade, also found that problems containing small operands appeared more often than those containing large operands. Thus, representations of small numbers is likely to be richer than representations of large numbers. This also seems likely to be true of the alphabet. A further argument may be made that exposure is increased for the very early letters (i.e. A through D) and the very late letters (i.e. W through Z), which are occasionally used as substitutes for numbers. Again no formal analysis has been done. It seems unlikely that the same arguments could be made for all other types of stimuli. It is more likely that changes in exposure lead to different effects throughout the sequence, thus, leading to the reverse Compression effect found by Moyer (in Potts, et al., 1978).

One of the remaining effects, Categorisation, may be unique to the number series due to its structure. Categorisation is thought to stem from the decade structure of the number series. Itsukushima, et al. (1990) postulate that this effect is due to subjects comparing only the tens-place digit. An hypothesis that is discussed further in Chapters Three and Seven. The alphabet does not share this structure and therefore the possibility of using this type of mechanism does not exist.

The differing nature of stimuli could also be responsible for seeming differences in effect. It has been suggested that the intuitive 'feel' for a number stems from subconscious access to perceptual representations (Ashcraft, 1992), which Moyer (in Potts, et al., 1978) demonstrated that subjects have access to as relative decisions are made. Since subjects are not aware of accessing this material, they label it a 'feel' for the number. This

particular aspect of numerical judgements is discussed further in Chapter Three of this thesis.

There is some debate about the origin of differences in RT, generated by changes in instruction (i.e. Larger versus Smaller decisions). Polich and Potts (1977) believe that these effects are due to the variety of solution methods available to subjects, which they contend make it impossible to generate a model of these judgements as one cannot be sure the subject is using only one strategy. The variety of solution strategies may be a product of increased use of the number series. The varied use of numbers is evidenced in the vastness of the definition of counting. Counting is defined as containing five principles. In the following list of these principles, the number words are referred to as numerons to distinguish them from fully defined numbers.

(1) *One to one correspondence:* each element in the set must map onto one and only one numeron.
(2) *Stable Order:* The numerons must be ordered and mapped in a reproducible sequence onto the items to be counted.
(3) *Cardinality:* the last numeron used during a count represents a property of the entire set (its cardinality or numerosity).
(4) *Abstraction:* counting applies to any collection of entities (all sorts of physical objects, possibly forming a heterogeneous set, as well as purely mental constructs).
(5) *Order irrelevance:* the order in which different elements of the counted set are mapped onto numerons is irrelevant to the counting process.

(Restatement of Gelman and Gallistel, 1978, by Dehaene, 1992)

This demonstrates the complexity of the counting task, which leads to rich and varied representations of the sequence and its elements. These representations may in turn give rise to solution strategies for relative judgement tasks. The alphabet, on the other

hand, although it may be related to more varied information than other sequences due to the connection with written language, is not as richly connected in the sense of an ordered set as the number series. It is therefore less likely to support the same number of solution strategies. A simple demonstration of this difference is the exposure subjects will have had to the number series in reverse order. This exposure will surely have led to the development of a reverse order representation which can be used for solutions to relative judgement problems. It is unlikely that the analogous representation exists for the alphabet in most people. These separate representations may generate similar RTs, thus, keeping the number series from demonstrating a Direction effect. The Direction effect found for alphabetic stimuli could be the result of translating the results of only one representation. These representations may also be sensitive to instructions and thus generate the instruction effects found. This theory is supported by the Polich and Potts finding of an Instruction effect for alphabetic stimuli, which has a different nature from the Instruction effect found for numbers. The relative slowness of the Incorrect Order decisions could be due to the higher likelihood of subjects using run-through on these trials combined with the increased amount of time taken for run-through. The increased likelihood of run-through would be due to the need to access the sequence in correct order then run-through a portion of it to determine how it would be constructed in reverse order.

5. Conclusions

It is a worthwhile hypothesis that all relative judgements of scalable or ordered material are made using the same type of underlying representations, the only difference being in the amount of learning, with numeric stimuli representing the highest level of learning possible and alphabetic stimuli a relatively low level of learning. Other stimuli fall somewhere between or below these two, with newly learned sequences as examples whose learning is below that of the alphabet. Such an account is supported by findings from the literature on children's acquisition of the number word sequence (Fuson, 1982). During the time a child is learning

the number sequence it passes through phases which exhibit phenomena much like those found with alphabetic stimuli, for example, only forward recitation with entry at selected points.

The common representation theory is also supported by relationship between number processing and language processing. Dehaene (1992) points out that number processing, in its fundamental form, seems to be linked to the ability to mentally manipulate sequence of words or symbols according to fixed transcoding or calculation rules. He also notes that Hurford (1978) found that the ability to manipulate numbers emerges through the interaction of language capacities and other cognitive abilities to recognise and manipulate concrete objects and collections. This type of support is further examined in Chapter Two.

According to this theory SDE, as the only shared effect, stems from the common underlying representation, which is likely to be related to subjects disposition toward creating a linear ordering of stimuli, while the remaining effects arise and are replaced as learning of the sequence proceeds. The previous section outlines explanations for the effects defined that are consistent with this theory. A model consistent with this theory must show SDE (perhaps, with discontinuities) and Compression across development, with initial effects of Direction, interaction of Direction and Separation, and the ability to use rough position tags and interval as well as ordinal data. As training progresses processing speed should increase, and these effects should decrease. This model needs also to have the facility to produce a discontinuity in the SDE curve between separations One and Two, Congruence, Categorisation, Zero-term, End-term and Funnel effects. It should also have the potential of accommodating more than one solution strategy.

B. Relative Judgement Theories

As researchers found new effects associated with comparison judgements, they developed new theories, or refined existing ones about the mechanisms responsible. However, as yet no one has

combined the comparison judgement data with the sequence-learning and production data to produce an unified theory. After reviewing some of the effects that have been found in section A, the numeric and alphabetic theories they have inspired are described in the next two sections (B.1 and B.2). The following section (C) examines machine models, which have grown out of the theories. The beginnings of a unified account of sequence learning, production and comparison judgements is presented in Chapter Three.

By the reasoning presented in section A.5, a model which can account for numeric, alphabet and other stimuli results must show SDE (perhaps, with discontinuities) and Compression across development, with initial effects of Direction, interaction of Direction and Separation, the ability to use rough position tags. As training progresses processing speed should increase these effects should decrease. This model needs also to have the facilities to use interval data and produce a discontinuity in the SDE curve between positions One and Two, Congruence, Categorisation, Zero-term, End-term and Funnel effects, as well as variance in responses and solution strategies and changes in processing with experience.

1. Theories

Since the modern theories depend upon aspects of the older theories, this review begins with some older theories.
In 1964, Audley and Wallis defined the Funnel and Cross-over/Congruence effects, which are effects of the instructions to subjects much like those found by Polich and Potts (1977). The difference between the two effects is that the Cross-over/Congruence effect is defined on two stimulus dimensions (bright - not bright, and dark - not dark), while the Funnel effect addresses only one dimension (bright - not bright). They explained these effects by suggesting a theory of covert responses. These are internal reactions to a stimulus (A), taking the form of a response to the stimulus (i.e. 'A is bright'). The responses are either stronger (or always) along the dimension compatible with the

salient property of the stimulus. For example, covert responses are along the bright dimension for bright stimuli.

They presented evidence for their theory with the finding that when asked to comment on two bright stimuli, subjects tended to identify the brighter without prompting. This is a simple response competition theory – each possible response to a situation is imagined to be continually occurring in covert form with response rates varying with the task. These possible responses compete with each other to be the final response. The winner of the competition is declared when one pattern of covert responses is of a specified kind. The specific kind of pattern Audley and Wallis used was an uninterrupted run of one type of covert response.

Thus, the Funnel effect is thought to be due to the fact that if a bright stimulus is encountered it is unlikely that a covert response of darker will be generated, so an uninterrupted run of brighter responses is the only likely occurrence, so the brighter of the two stimuli will be found more quickly. To identify the darker of two bright stimuli, they propose, either a slower covert darker response or a time consuming translation of the dominant response to match instructions. However, with a dim stimulus there is more chance that it may generate darker responses as well as brighter responses, allowing the brighter or darker stimulus to be identified equally fast. The Cross-over effect is then the conjunction of the two Funnel effects for the two dimensions. The cross-over occurs where the two funnels meet.

This theory is designed to account for Funnel and Cross-over/Congruence effects. It may also be expanded to explain the SDE. The expansion might say that the closer two stimuli are on the dimension to be judged, the more likely it is that any response run (i.e. A is brighter) will be interrupted by the opposite response (i.e. B is brighter). The Direction effect occurs when the stimuli are to be judged in the non-dominant dimension, RTs are increased by translation of the response or waiting for a run of the slower, non-dominant responses. The interaction of Direction and Separation cannot be explained as both mechanisms postulated to be

responsible for the Direction effect would predict a uniform increase in RTs for all backward decisions. Effects consistent with the use of interval data may be due to the sensitivity of the covert responses. The zero-term effect and the Compression effect, could arise if one contends that the difference in covert response rates changes as a function of learning, with zero and higher stimulus values less well learned. But it cannot account for the Categorisation effect, the variability in responses or solution strategies, discontinuity in the SDE curve of the use of position tags. For a summary see tables 5a and b.

Restle (1970) postulated that in comparisons between (p+q) and (r), (p) and (q) were converted to internal magnitudes, which take the form of line segments and their sum estimated by combining these line segments and estimating the resultant length. This model was developed to account for the inherent imprecision in processing. It appears to suggest that the ability to map operands to line segments is an innate ability. The use of interval data is inherent in the design of the system, since it relies on the use of perceptual data. The Compression effect can also be explained by subjects' discrimination linked to the proportional difference between the two sets of stimuli. Thus, if two stimuli are both 500 pixels long a difference of 100 pixels will be easier to notice than if they are 5000 pixels long. The effects which are not explained are Congruence, Categorisation, zero-term, Funnel, Direction, or the interaction of Direction and Separation, the use of position tags, discontinuities in the SDE curve and the variability in response solutions nor how processing would change with experience. See table 5a and b for a summary of effects accounted for by this model.

To explain the Minimum stimulus effect that they found, Parkman and Groen (1971) hypothesised the presence of some type of counting mechanism embedded in the processes of mental addition and comparison of single-digit numbers, if not multi-digit numbers as well. Their finding was that the minimum addend accounted for substantially more RT variance than did separation. Their model is a discrete model in which both stimuli are encoded and held in an

internal store. The subject begins counting, and after each step compares the result to the two encoded representations. When the count matches one of the encoded representations that stimulus is pronounced to be smaller.

Potts et al. (1978) outlined the difficulties with this model. The Min effect has a strong non-linear component found by Moyer and Landauer (1973). When Min is held constant, there is still a Distance effect (Parkman, 1971). The results of experiments where a standard is held in memory (i.e. Itsukushima, et al.,1990 and Sekular, et al., 1971) cannot be explained, neither can the Congruence, Categorisation, Compression or zero-term effects, the use of position tags, variability in response solutions, changes in processing with experience or discontinuities in the SDE curve found with alphabetic stimuli. Although, Parkman and Groen (1971) expanded their counting theory to account for Compression by hypothesising that comparison of internal representations and the count register became more difficult as the magnitudes of the numbers encoded increased, this does not allow for an explanation of the reverse Compression effect found with non-numeric, non-alphabetic stimuli, nor does it counter all the other basic problems with their theory.

To co-ordinate an examination of the modern comparison judgement theories, Maki (1981) defined a classification system for proposed theories, which includes three major types:

(1) Pure analog models - stimuli are thought to possess or be represented by continuous characteristics along dimensions to be judged.
(2) Pure discrete models - subjects are thought to generate semantic codes representing size information, if two stimuli are close on size dimension, size codes with gross information are not enough.
(3) Mixed (Hybrid) models - both discrete and analog coding occur, when decisions are made either analog processing occurs first or both occur simultaneously.

Kosslyn lists the three main characteristics of both model types (in Potts, et al., 1978). For analog representations in memory these are:

"(1) They are 'continuous' in two senses: (a) so dense syntactically as to be practically undifferentiated. (b) so dense semantically as to be undifferentiated in this regard (Kosslyn)." He explains this characteristic by means of an analogy. If there were infinitely many degree marks on a thermometer it would be extremely syntactically dense, but even with this number of markings, if each marking is only meaningful to the nearest degree than they are semantically differentiated.

"(2) They bear a 1:1 structural isomorphism to the internal representations underlying perception of the referent (Kosslyn)", which means that the organisation of the parts of the memory representation is the same as the organisation of the parts of the thing it represents.

"(3) They may be appropriately processed by mechanisms usually recruited only during like-modality perception (Kosslyn)". That is, that the mechanisms used in perceiving the object are believed to be analog, thus the mechanisms used in processing these perceived objects (i.e. compare them along some dimension) must operate on analog data. These processing mechanisms can then be used to process analog memory representations. This 'analogous' processing is termed "analog processing".

Kosslyn notes that for these properties to be meaningful, the mechanism which interprets the memory representations must be sensitive to these characteristics.

The characteristics of discrete models, on the other hand, are:

(1) They are both syntactically and semantically differentiated (not very dense). They are practically unambiguous with regard to both identity and meaning.

(2) They bear no necessary structural isomorphism to representations underlying perception of the referent; the correspondence between a discrete representation and the

represented object may be established on entirely arbitrary grounds.

(3) They cannot be processed by mechanisms specialised for dealing with perceptual representations.

Holyoak and Walker (1976) designed an experiment to determine which of these types of model (analog, discrete or hybrid) best predicted the relative judgement data from semantic stimuli. They tested three measures: rated distance, group membership and ordinal distance. Each measure was related to a different model. The rated distance measure was designed to test an analog representation. An analog model would be supported if an absolute rating of the magnitude of a stimulus was the best predictor of RT. The group membership rating was an investigation of the predicting ability of a type of complex discrete representation, with high level discrete data used to distinguish between groups, and low level, difficult to access, discrete data used to distinguish stimuli within groups (that is, more precise ordering information). In order to explain the continuous nature of the SDE curve, this model hypothesises that the group boundaries fluctuate from subject to subject or trial to trial. Ordinal information was used to represent a relational model, which does not fall into any of Maki's categories. The relational model states that the only information stored for semantic orderings is the relative magnitude of one concept to another. This model does not fall into any of the categories, because it does not predict SDE. Holyoak and Walker's finding was that the best predictor of RT was rated distance, which corresponds to an analog model.

Holyoak (1977) went on to investigate the role of imagery as the analog magnitude representation used in the size comparison process, which amounts to an investigation of Restle's (1970) theory. This line of inquiry was suggested to him by the striking similarity of the functions relating RT and objective distance in mental size comparisons and in perceptual comparisons, such as comparisons of line lengths. He conducted two experiments. The results from one were fully consistent with the view that subjects could use imagery in these tasks and that when they did, they

compared objects at their canonical sizes in order the reach a decision. Imagery subjects performed mental operations to produce normal-sized images of the first object as soon as the second object was presented. A second experiment showed that imagery subjects performed comparisons slower than control subjects, which implies that imagery is not the comparison method of choice. Holyoak noted that, "the present negative conclusions about the role of imagery in size comparisons in fact support the possibility of a domain-independent model, since imagery seems an implausible vehicle for comparisons involving numerical magnitude, or terms differing in degree along such dimensions as time and quality". He finds support for this view from Pavio's (1975) finding that comparisons between objects drawn from different semantic categories can be made just as quickly as comparisons between objects from the same category, which implies that at some level the size of all objects is represented on a common scale. His findings also represent a rejection of Restle's (1973) line comparison theory as it too rests in imagery.

Also in an attempt to examine the relationship between discrete and analog processing, Kosslyn and Pomerantz (1977) trained subjects on a task in which different sizes of object were labelled with different colour names. One group overlearned the 'small' and 'large' labels 200%, while the other overlearned them by 500%. Since overlearning should effect only the discrete (label) portion of processing, if subjects only use this information there should be an overall decrease in RT. However, if they use both discrete and analog data, discrete should be differentially speeded up. If discrete processing is fast enough, analog processing should never occur, and the SDE should disappear. If analog finishes first, the results should look like perceptual comparisons. He found that the amount of overlearning was critical in determining whether SDE was obtained, only with large amounts of overlearning did SDE disappear.

He also examined the issue of imagery used in size comparison as an indisputably analog process. Looking at whether both types of representation are compared serially or in parallel and whether

imagery is essential or epiphenominal. The serial contention is that discrete information is always accessed first, with analog only used when necessary, while the parallel view is that both representations are processed in parallel. His size-to-colour experiment produced results consistent with the parallel model and inconsistent with the serial model. He then replicated Holyoak's (1977) imagery experiment, which found that imagery could be used in the comparison process as an analog encoding, but was not the method of choice.

Riley (Potts et al., 1978) points out that the finding of end-term effects for sentences with embedded relational information, leads to problems for analog models. The distance measure on true (A < B) and false (B < A) was the same, therefore by the analog model, RTs should be the same. However, true sentences showed an advantage for the initial element and false sentences showed an advantage for the terminal element. Potts (1972) explained this finding by some subjects adopting the strategy of testing the first term for initial/terminal element. This is supported by the finding that there is no interaction between true and false in sentences which do not contain end-terms. This indicates that subjects have a variety of solution strategies available and that a single model of relative judgement is unlikely to account for all effects.

In an investigation of analog models, Moyer (in Potts et al. 1978) noticed that the analog model, having no trouble with experiments in which subjects are only given ordinal data or in which ordinal and interval data are confounded (i.e. stick length comparisons and digit- and letter-comparison work), also predicts that the analog internal representations preserve more than ordinal information if quantitative information is available as well. Thus, mental comparison time should also be influenced by this quantitative information (i.e. long RT if the difference between adjacent items in a series is small, short RT if the difference is large).

Referring to his findings from the named circles study, Moyer points out that they support an analog comparison model. The results indicate that perceptual and ordinal information is used,

when available, for comparison judgements. Thus, the perceptual values and stimuli appear to be the same. His version of this type of model supposes that subjects order the names of the circles in memory according to size and that when the stimuli for the task are encountered the subject conducts a self-terminating search beginning at the 'large' end of his or her order list of names for either of the stimuli. Congruity is explained by the subject beginning the search at the end of the list indicated by the instructions. For experiments in which a *linear* congruence effect is not found, he cites Trabasso and Riley (1975), who found time advantages for pairs at both ends of the series. He indicates that both his analog model and Banks' semantic code model can be used to explain these effects.

This model encounters all of the problems of the Parkman (1971) counting model, among the most severe being the inability to explain the findings of Itsukushima et al. (1990) and Sekular, et al. (1971) that SDE occurs even when the subject holds one of the stimuli in memory. These models would predict linear response times for 'larger' decisions, which do not occur.

In the Potts et al. (1978) paper Banks counters Moyer's support of an analog model by citing an example of coding an analog stimulus into discrete categories. His example is, pronouncing coffee as "too cold", which takes the analog temperature value and translates it into a discrete category of "too cold". He explains that in accordance with this approach the appreciation of a subtle distinction would require a new category boundary, say within the category "a bit too cold". His "..general finding is that, whatever the stimulus attribute in question, subjects can recognise only a limited number of absolute levels." The number of absolute levels is set at 7 ± 2 (Miller, 1956). This finding leads to the conclusion that since representations are not semantically dense, they are functionally discrete.

Banks furthers this argument by reiterating Bower's (1971) finding that "..memory for stimulus levels is not mediated by 'mental images of specific referents', ...[but] that a general cognitive

structure underlies the use of category labels in an absolute judgement task, whatever the stimulus modality." Banks reports Bower's contention that this structure is a linear ordering that allows relations such as 'between' or 'greater than', but only carries ordinal information. However, Moyer (1978) showed that subjects use interval as well as ordinal information. The linear order structure is hypothesised to be abstract and amodal. Bower equates this structure with a "scaffold" onto which tags from any modality can be grafted. This is a discrete, semantic coding model.

Banks goes on to say that in cases where such "scaffolds" cannot be constructed, subjects rely on their real-world or linguistic knowledge. The entire system is sited in the realm of semantic memory and is considered to be constructed out of "..any of a variety of structures, depending on which model of semantic memory ultimately proves to be correct". He concedes that Pavio (1975) found that comparisons between categories of animals did not produce an RT advantage over comparisons within categories, which counter-indicates this theory.

The differences between these two theories are explained by Potts et al. (1978). They first note that the analog approach assumes the data base contains interval or ratio scale information and that mental processing computes responses using analog and not discrete data. While the semantic coding model avoids assumptions that analog information is contained in the data base, without necessarily restricting the types of coding available. In the same manner analog processes may be used in encoding, but the main operations are performed using discrete codes. They point out that the Banks model is a hybrid model combining an analog encoding process with a discrete choice stage.

According to Banks, et al. (1976) the semantic congruity effect indicates that processing of the digits is done in terms of size codes rather than analog representations. This is an adaptation of Audley and Wallis' (1964) covert response theory. Banks, et al. contend that the congruity effect emerges at the stage of processing at which the size codes for the digits are matched to

the codes for the instructions. They explain Min and SDE effects in terms of size codes. They contend that initially numbers are perceived as quantities and that very early in processing the quantitative intuitions get converted to discrete semantic codes, and the major part of processing is devoted to operating on the semantic codes. Once the sizes of the digits to be compared have been encoded, the processing is very similar to the processing of comparatives with other types of stimuli. A major failing of this model is that they do not explicitly postulate how the conversion mechanism operates or how the ability to convert quantitative intuitions to semantic codes is learned. Moyer and Landaur's analogy between digit comparisons and other comparative judgements is supported, but in a curious way. The similarities between the processing of comparisons among digits and among other sorts of stimuli derive from a common, discretely encoded form into which they are translated rather than from a common analog medium in which they are processed.

Banks, et al., contend that SDE indicates that the ordered information is represented by an analog code, but that the end-term and congruity effects are most easily explained by the Discrete Semantic Code Model. This model assumes a logarithmic subjective scale for the encoding of digits, and that subjects distribute the codes 'evenly' over the scale, which accounts for the Compression effect. Meaning that they do so in a subjectively linear manner. Subjects are seen to make a binary decision, items are either encoded as 'larger' (symbolised as Lt) or as 'smaller' (St). These encoding symbols (Lt and St) match the choice codes. Because one of these codes will match any instruction, no transformation is necessary to ensure congruity. This reduces the congruity effect. The model predicts SDE as a side-effect of using semantic memory. They justify this by citing studies by Pavio (1975) and Banks and Flora (1977) which showed strong SDE when semantic memory was used. Banks, et al. do not explain why SDE should be a result of using semantic memory.

Banks, et al.'s model can account for the Congruence, Funnel and Compression effects. By expanding it to include two scales of 'Lt'

and 'St' for each number, it may be used to explain number stimuli direction data. Again if zero was not automatically given a code the Zero-effect could be due to the generation of a code for each zero trial. Interval data can also be reconciled by attributing it to the translation process. However, this model cannot predict variability in solution strategies, discontinuities in the SDE curve, the interaction of direction and separation, the use of position tags, Category, and End-point effects, see table 5a and b.

At the discrete end of the analog - discrete spectrum, Lovelace and Snodgrass' (1971) discrete model postulates position codes which account for the SDE and direct letter-to-letter associations for their finding of an increased RT from separation one to separation two. These direct letter connections are not considered to exist for use in backward decisions, a consideration indicated by subjects' inability to recite the list backward. They found significant direction effects at separation eight, which, they suggest, indicate a stable component of direction only minimally related to recitation strategies. This is because very little run-through is found at such a large separation value. It is presumed to be a "set" effect, i.e. subjects are looking for forward pairs, making yes/no judgements about correct alphabetic order. This presumption is contra-indicated by the Polich and Potts (1977) finding that true/false judgements of alphabetic order are slower than larger/smaller decisions.

Lovelace and Snodgrass contend that position tags are more precise at the beginning of the list and that the confusibilty of position tags increases toward the end of the list. Thus subjects should need to recite more toward the end of the list, causing the Compression effect. But this also predicts direction effects should be greater near the end of the sequence, which does not occur. They postulate instead, that beginning positions are more overlearned than later parts, hence either the time taken to identify the stimuli increases or inter-letter associations decrease as one moves down the alphabet. Fitts and Switzer's (1962) findings support the idea that time to identify the stimulus increases,

because subjects take more time to call out the names of the letters they see as they approach the end of the alphabet.

This model accounts for SDE, Direction, Compression, End-term and Funnel effects and the use of rough position tags. Zero-term effects, may arise as the result of no position code for zero, one would then have to generate such a code to make decisions involving zero. It may be expandable to account for number data, by postulating the existence of position codes for forward and backward recitation (diminished Direction effects, and the arrival of Congruence, Categorisation), however, it is not immediately obvious how you would generate backward position codes with respect to an infinite sequence nor how processing would change as a result of experience. Problems for this model also arise when one looks for explanations of variability in solution strategies and responses, and discontinuities in the SDE curve (summarised in table 5a&b).

Marks (1972) offers another alternative to Audley and Wallis' theory. He discusses judgements of probability as opposed to brightness. Using abstract stimuli, he hypothesised that eliciting a judgmental response may be conceived of in three stages: (1) choice of pole P, (2) assignment of a stimulus value, I_p, which the judge assumes the pole label accurately represents, (3) determination of the judgement of the stimulus relative to the pole. He defines this third stage as a special case of Thurstone's (1927) "discriminal dispersion" theory. The internal representation of each stimulus is a normal distribution of so-called discriminal processes, with the standard deviation of the distribution known as the "discriminal dispersion". Mark's model makes the assumption that the discriminal dispersion of a stimulus is proportional to its distance from I_p - the ideal polar stimulus value. The speed of relative judgements is directly proportional to the discriminal difference of the two stimuli. That is, the less the distributions for the two stimuli overlap the faster the decision. The overlap in distributions is postulated to increase with the distance from the pole. Thus, decision time between two proximal stimuli should also increase with the distance from the

pole - this is the Compression effect. Practice reduces dispersion, thus decreasing SDE.

The summary in tables 5a&b shows that, while this model accounts for SDE, Compression, Congruence, End-point, Funnel, Direction and the interaction of Direction and separation effects and variability in responses, it fails to account for Zero-term and Category effects, discontinuities in the SDE curve, and also fails to provide facilities for the use of position tags and variable response strategies.

Another analog model is the Relative Judgement Theory presented by Link (1990). He postulates that a comparison stimulus, S_C, is transformed into a waveform having amplitude at time t, $S_C(t)$, which is compared against an internal psychophysical referent represented by another waveform, $S_r(t)$. The comparison between S_C and S_r must produce a third waveform, $d_S(t)$, that contains information about the difference between them. He envisages these waveforms as characterising the electrical signals representing the stimuli and their difference.

Information is then extracted from $d_S(t)$ by a mental algorithm which operates on $d_S(t)$ to produce a comparative value. These comparative values are accumulated over time during a trial either until enough positive values produce an accumulated total equal to, or larger than, the response threshold for a "Larger" response, R_L, or until sufficient negative values accrue to produce a "Smaller" response, R_S. The starting position of an accumulation of comparisons is labelled C and the two possible terminal values are labelled A and -A. Note that the comparison values are variable across time steps. This method of comparison is termed a Random Walk as the path to the threshold is stochastic.

The values of A, -A, and C are assumed to remain constant during an experimental trial. However, across trials these values may change to meet experimenter imposed task demands. The value of A can be reduced to meet response time deadline restrictions. As the value of A is reduced, response time decreases. Changes in the C

value are believed to reflect a bias, in the subject, toward one response or the other. As C increases toward either A (or -A), the subject is more likely to make that response.

The mental algorithm is assumed to generate a constant expected value for the duration of the judgement process. This provides that for any value of t, the mean or expected value of the mental algorithm equals μ. This is the rate at which the sum of the differences drifts toward one or the other response threshold, A or -A. Thus, large values of μ lead to quick termination at the response threshold, while smaller values result in longer termination times. The termination time for reaching the response threshold is believed to reflect the decision time and, thus, forms the component of RT that is of major interest.

This theory is capable of explaining SDE. Through the manipulation of the start of the walk, C, it can also explain the Compression effect, the End-term effect, Congruence and Funnel effects. If A and -A are allowed to vary independently interaction between Direction and Separation may also be explained. For a summary of the effects explained see table 5a. The representation effects explained are the same as most of the other models. Variations in RT and the use of interval data are the only effects explained, these are summarised in table 5b.

Overall, none of the theories presented is able to explain all of the relative judgement results. Therefore, it is necessary to look elsewhere for a theory, bearing in mind the failings of the theories outlined in this section and the findings that a successful theory is likely to combine discrete and analog processing with a combination of solution strategies.

2. Conclusions

From the preceding two sections, it is apparent that a comprehensive theory of how long-term sequences are represented must have the capacity to account for a wide variety of effects. These effects change with experience and the stimuli to be learned.

Thus any model must also have the capacity to learn from experience. Connectionist models are designed with learning in mind. To some extent they developed as a means of utilising neuroscientific findings about the way in which neurones could be affected by experience to allow the brain to learn.

A further conclusion, from the theories discussed in section 1.C, is that a successful model should probably contain the ability to use both discrete and analog data solution strategies operating in parallel. It seems that this will be just two of a variety of solution strategies available. Al least one other solution strategy employing run-through and counting is discussed in detail in Chapter Four.

In Chapter Three, the development of a new connectionist model of the representation of long-term sequences is begun. Before that task is undertaken, it is informative to examine other machine models, to see if they can account for more of the effects outlined in section A.

		General Effects		Initial Effects			Exp. Effects		Instruct Effects	
Author	Theory	SDE	Compre-ssion	SDE Discont.	Direct	Interact DirxSep	Zero-term	End-term	Congru-ence	Funnel Effect
Audley & Wallis (1964)	Covert Responses	X	X		X	X	X	X	X	X
Restle (1970)	Line Comparison	X	X							
Parkman & Groen (1971)	Counting by Ones	X	X							
Moyer (in Potts, et al., 1978)	Perceptual Stim Search	X							X	
Banks (in Potts, et al., 1978)	Size Codes	X	X				X		X	
Lovelace & Snodgrass (1971)	Position Tags	X	X	X	X			X		X
Marks (1972)	Pole and Stimulus	X	X		X	X		X	X	X
Link (1990)	Random Walk	X	X		X	X		X	X	X

Table 5a. Judgement Effects Explained by Relative Judgement Theories. The general effects are those that have been found across stimuli and across exposure amounts. The initial effects are found with alphabetic stimuli, which are considered to represent a lower level of learning than numeric stimuli. The experience effects are thought to arise for either a large amount of or dearth of experience with certain elements of the sequence. Instruction effects are changes in RT with the instructions given to subjects.

Author	Theory	Solution Variance	Response Variance	Learning Changes	Position Tags	Category Info	Interval Data
Audley & Wallis (1964)	Covert Responses		X				X
Restle (1970)	Line Comparison		X				X
Parkman & Groen (1971)	Counting by Ones						
Moyer (in Potts, et a.,1978)	Perceptual Stim Search						X
Banks (in Potts, et al., 1978)	Size Codes						X
Lovelace & Snodgrass (1971)	Position Tags						
Marks (1972)	Pole and Stimulus		X				X
Link (1990)	Random Walk		X				X

Table 5b. Representation Effects Explained by Relative Judgement Theories. These effects include subjects' ability to employ more than one solution strategy, the variability in RTs across trials, the changes in RT that occur as a experience with a sequence increases, the reports by subjects of using rough position tags, the effect found in multi-digit experiments that for decades not containing the Standard RTs were approximately uniform, and the finding that subjects use interval as well as ordinal data when making decisions.

C. Implementations

Along with the theories outlined, implementations have been developed. These implementations are more informative as they are constrained by the necessity for realization on computer. These constraints require a clearer definition of terms leading to more specific models.

Interest lies in an implementation which can explain SDE and Compression, with initial effects of Direction (alone and interacting with separation), developing into Congruence, and Category effects. This implementation should also have the capacity to produce Zero-term effect, End-term effect, Funnel effect, and a discontinuity in the SDE curve, as the result of experience. It should generate variable responses using a variety of solution strategies including, but not limited to, discrete data (i.e. position tags) and analog data (i.e. interval information).

After reviewing some of the implementations from the literature, construction of a new model, including a machine implementation, is started in Chapter Three.

1. Existing Models

Three approaches to modelling relative order decision data have emerged. The first is by extension of models originally designed as sequence learning mechanisms. Many of these sequence learning models are designed around the effects demonstrated by short-term retention of sequence information, which means that they are usually unsuitable for the tasks outlined in this thesis. The second approach to modelling these effects is to take for granted an underlying analog representation and design a response generating mechanism that accounts for the variability in subjects' responses. The third approach is purpose-built for relative order judgements, but restricts the effects examined to a single type of stimulus. This review includes two examples of the first approach and one of each of the others. The example of the third approach focuses on alphabetic stimuli only.

1.1 Model One

The first short-term sequence model is proposed by Lewandowsky and Murdock (1989), it is based on the Theory of Distributed Associative Memory (TODAM) first presented by Murdock (1982, 1983). This is very like a network defined in the PDP environment, but is not a classic connectionist network. In TODAM, serial order information is represented by a series of pairwise associations between successive items. It is based on the traditional chaining approach, except that the beads on the chain and their links are all stored in a common location. With this configuration, no item-to-position associations are required to model separation.

A stimulus item is represented as an N-dimensional random vector. That is, each item consists of a number (N) of features. A feature is a primitive in this theory, an abstract concept that is not to be identified with physical or semantic properties of the stimulus. All newly acquired information is added to a common memory vector, by convolution. During this process, the numeric value of each newly acquired feature is added to the previous value in the corresponding element of the memory vector. Thus, all information in memory is stored in a common pool with many locations or elements, and because features of all stimuli are superimposed, the memory vector itself bears no direct resemblance to any of the individual items.

Each input vector is bounded by a start term (x) and an end term (y). After presentation of vectors (A, B, C) the memory vector contains the item information for (x, A, B, C, y) and the serial order information consisting of the diagrams (xM_0, Ax, BA, CB, yC). M_0 = memory vector at time 0, with (N) elements. Retrieval is initiated by probing with the start signal (x). Following recall of item (A) the retrieved item is added to the memory vector, using the same process as in learning, and then serves as the probe for the next item, etc.

They point out that an analytic (mathematical) solution is only available for recall in which the preceding item is available as a

probe regardless of the success of the prior retrieval - as in an anticipation method or a partial report paradigm, memory span paradigms or strictly scored serial recall.

They discuss two implementations of their model: Open-Loop and Closed-Loop. Both use the same memory vector representation and recall procedures, but in the Closed-Loop implementation the retrieved vector is converted to an overt response by a competition among a pool of possible responses. This requires that the user specify exactly which items are competitors and which are not. In their implementation all items following the item to be recalled are competitors. In the Open-loop implementation no such competition pool exists, and the retrieved vector serves directly as the response without determining if it is a member of the sequence.

Lewandowsky and Murdock show that TODAM can account for many of the serial order acquisition and recall effects except the distance effects. Their model will never simulate distance effects since it relies on the implicit retrieval of all intervening items, which renders it useless for the purposes of the present task. Murdock (1993) updates this model to include "chunking" (found in short-term memory experiments) among other effects. However, since he is interested in short-term effects, he does not address distance among the effects modelled.

1.2 Model 2

A second short-term model was designed by Lacouture and Marley (1991). They propose a model of serial learning which is a classical PDP model. It uses an adaptation of the backpropagation (bp) learning algorithm, called mean variance bp (MVBP). They use this algorithm to train an encoder network (an encoder network is one with n input units, n output units and m (<< n) hidden units). Their network contains only one input and output unit for each training pattern, with the maximally active output unit given as the network's response.

Backpropagation networks contain two or more levels. The outer two levels are called the input layer and the output layer, any intermediate levels are called hidden layers. All layers are attached to each other by means of weighted connections. A vector representing the stimulus is presented to the input layer. The values contained in this vector are then passed to the next layer by multiplication with the value with the weighted connections.

$$net_j = \sum \omega_{ij} * A_i \dagger \quad \forall \text{ i connected to j}$$

where:

net_j = net input to unit j.
ω_{ij} = weight from unit i to unit j.
A_i = activation of unit i.

Each unit in this layer generates an activation value using the activation passed to it from the previous layer. The function used to calculate the activation in a unit is generally the sigmoid activation function:

$$A_i = \left(1/1 + e^{(-net_i)} \right) \dagger$$

where:

A_i = activation of unit i.
net_i = net input to unit i.

This continues until the output layer is reached, at which point the resultant activation vector is compared to a target vector for the stimulus. The difference between these two vectors is the error for the stimulus pattern. This error is used to change the weights in the network.

$$\Delta\omega_{ij} = \varepsilon(\delta_j * A_i) \dagger$$

where:

ε = learning rate.
A_i = activation of unit i.
δ_j = error value specific to each unit.

---

† formula taken in essence from McClelland and Rumelhart (1988)

Initially this is done for the weights on connections to the output layer. For these weight changes the error value is:

$$\delta_i = (T_i - A_i)A_i(1 - A_i)\dagger$$

where:

T_i = target activation for unit i.

A_i = actual activation of unit i.

The error is propagated 'backward' through the network to be used to change the weights between all of the layers. For units in the remaining layers, the error becomes:

$$\delta_i = A_i(1 - A_i)\sum \delta_j \omega_{ij} \dagger \quad \forall \text{ unit j connected to i.}$$

where:

A_i = activation of unit i.

δ_j = error value for unit j.

ω_{ij} = weight of connection from i to j.

These weight changes represent the network 'learning' to associate the input pattern with its target output.

In regular bp networks the number of stimuli the network is capable of correctly learning is roughly equal to (2^h) where (h) is the number of hidden units. The other stimuli are not learned. This is the optimal solution when the patterns to be learned exceed the capacity of the network, but humans do not always perform optimally. Data show that humans allocate resources in such a way that their performance is essentially the same over all stimuli presented. Lacouture and Marley designed MVBP to use human type resource allocation, by implementing a form of selective attention. The learning mechanism of the network is made more sensitive when a stimulus with relatively large mean-square error is encountered. The weight change equation becomes:

---

† formula taken in essence from McClelland and Rumelhart (1988)

$$\Delta\omega_{ij} = -\frac{\partial E_p}{\partial \omega_{ij}}\left\{\alpha + \gamma\left(E_p - E\right)\right\}$$ § for some fixed $\alpha > 0$.

where:

ω_{ij} = the weight from unit i to j.

E_p = the mean-square error for pattern p.

E = mean-square error over all patterns.

and

$$\gamma = \frac{2\lambda}{n}$$ § for some fixed $\lambda > 0$.

where:

n = the number of input units.

This adaptation to the weight change equation allows the network to 'attend' to the poorly learned stimuli.

Since the network on which they implement MVBP allocates one output unit to each stimulus, and since stimuli are not learned to 0% error, each stimulus becomes associated with a probabilistic (Gaussian) distribution of activity on the output layer. The discriminability of stimuli is then dependent on the overlap of these Gaussian distributions, thus producing a representation which echoes the "Discriminal Dispersion" theory outlined by Marks (1972). To enhance the discriminability effect random noise is added to each stimulus, with the mean of the absolute value of the random numbers equal to 0.20 or 20% noise.

It has been found that the main limitation of the model is the need to determine in advance the number of hidden units to be used. But Lacouture and Marley point out that Ash (1989) and Kruschke and Movellen (1991) have demonstrated that it is possible to have a device that dynamically allocates resources.

To use this network to model relative judgements would require a module that implemented a decision based on the difference in the

---

§ formula taken in essence from Lacouture and Marley (1991)

Gaussian distributions of the elements of interest. One such decision method is a random walk. To use this method would require simultaneous access to the representation of the two stimuli, to determine the difference between them. This would require two networks because, if only one network was used, it would be impossible for the model to determine which output units were responding to which stimulus. Although this network is capable of modelling more of the effects of interest than the TODAM model of Lewandowsky and Murdock, the unnecessary complexity of two BP networks again renders it useless for modelling this task.

1.3 Model 3

The model developed by Poltrock (1989) is a mathematical model designed to explain relative order judgement effects without specific reference to the interpretation of sequence information. He implements a random walk model, so called because the decision mechanism takes random sized steps to a goal which represents the model's response. Steps are calculated by subtracting two internal psycho-physical referents, which are representations of the stimulus events, one from the other. His model is based on Link's (1978) Relative Judgement Theory, described in the "Theories" section of this chapter. Response time for a decision about a pair of digits (i) and (j) is expressed as:

$$RT_{ij} = DT_{ij} + R$$

Where DT_{ij} is the decision time and R is the duration of all other processing. In this model all errors are attributed to incorrect decisions, with no mechanism for response execution errors. The inputs to the decision process are the analog magnitudes of the digits (i) and (j). These are the internal psycho-physical referents. They are believed to be in the form of distributions of possibilities that a value drawn to represent the number will be equivalent to that magnitude on a number line. This underlying structure, which Poltrock takes for granted is pictured in figure 1.

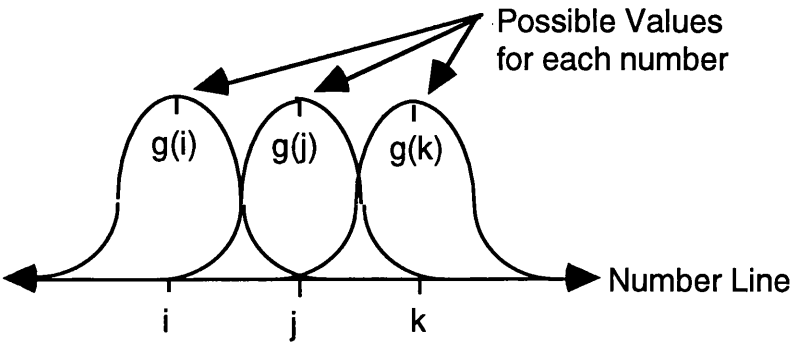


Figure 1. Analog Representation. The analog representation from which Poltrock builds his model, based on a "discriminal dispersion" idea like that outlined in Marks (1972). Each number is represented by a distribution of possible representations along a number line. These distributions are the probability that a value drawn as representative of that number will be of that magnitude, thus as the magnitude usually associated with number is approached (n), so is the mean of the distribution (g(n)). Note that the less the magnitude difference between the two numbers the more their representations overlap.

In each interval (Δt), the decision process computes the difference between the magnitude values drawn from the distribution for each stimulus and adds this difference d_{ij} to the accumulated value DT_{ij}. If g(i) represents the mean value of the analog magnitude for the digit (i), then the mean value of d_{ij} is μ_{ij} given as:

$$\mu_{ij} = g(i) - g(j)$$

The mean step size is equal to the difference between the means of the two distributions. Each step d_{ij} in the random walk is assumed to be equivalent to an independent random sample from some stationary distribution $f(d_{ij})$ with mean μ_{ij}. It is assumed (as in Link, 1975) that the distributions of the steps for the two pairs of digits (i), (j) and (j), (i) are symmetric: $f(d_{ij})$ is equal to $f(-d_{ij})$. DT_{ij} is initially zero and accumulates successive values of d_{ij} until it reaches boundary (A). The random walk stops and the decision is made when the absolute accumulated difference exceeds a

boundary. If $DT_{ij} \geq A$, the left-hand digit i is judged greater; if $DT_{ij} \leq -A$, the right-hand digit j is judged greater. The decision time DT_{ij} is Δt multiplied by the number of differences that are computed. A sample random walk is illustrated in figure 2.

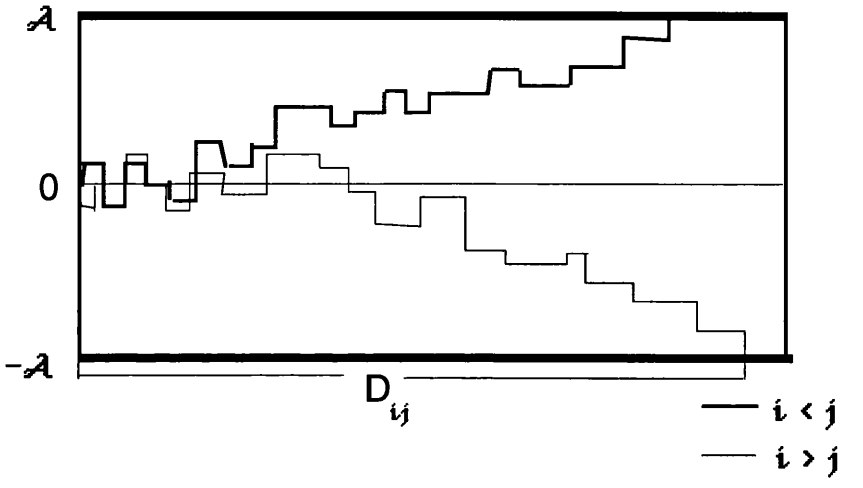


Figure 2. Sample Random Walks. The dotted line represents a walk in which i < j, so the desired result is -A, while the solid line is the opposite condition in which the desired result is A. D_{ij} represents the total time for the random walk to -A.

For the sake of the mathematics the random walk is assumed to end on a boundary and not overshoot it. The parameters that are important to note include the mean step size (μ_{ij}), which must be small relative to the distance to the boundaries or, equivalently, many steps must be needed to reach either boundary. Also important is the initial value (C) of DT_{ij}, this value may be non zero, resulting in a bias for one of the response choices. Explanations of the calculations for this model's parameters are not necessary for understanding the remainder of this thesis, but can be found in Poltrock (1989).

The random walk model is the first to address directly the issues of interest in relative order judgement data. It can account for variance in RTs across trials, SDE, Compression effect and the use of interval data. The model may also be expanded to predict the

discontinuity in the SDE curve as an epiphenomenon due to the amount of overlap in the first two distributions. However, this is not sufficient for the Poltrock finding that the discontinuity also appears in judgements of numeric stimuli when RT deadlines are imposed. It can also account for changes in RT due to learning, if the "discriminal dispersions" are hypothesised to alter with experience. The effects that this model does not explain are: Direction, interaction of Direction and Separation, the use of position tags or category information, Congruence, Funnel, End-term or Zero-term effects.

The example of the third approach to developing machine models for relative order judgements is the only model which has addressed run-through in alphabetic comparison judgements directly. It was developed by Klahr, Chase and Lovelace (1983). This model is based on a hierarchical representation of a simple linked list structure. The alphabet is divided into subgroups with each subgroup linked only to the following subgroup. The letters within each subgroup are linked in order as well (i.e. A to B to C etc.). Each letter is then associated with the name of the subgroup containing it.

A search process is defined to locate the letter following or preceding a given letter, involving first a serial self-terminating search of the subgroup names. Once the subgroup containing the letter of interest is located, its contents are searched, also using serial self-terminating search process. On trials in which the preceding letter is required, the search of the subgroup is carried out while maintaining the preceding letter in memory. Once the letter of interest is located, the subject either reports the next letter or the letter presently maintained in memory.

While this model simulates the differences in RT on tasks which require the location of preceding or following items, it cannot simulate relative judgement tasks. For some comparisons this model would be successful. (A, Q) would be compared faster than (A, B) because (A, Q) would probably only require a search of subgroup names, while (A, B) would require entry into a subgroup.

The serial self-terminating searches would cause comparisons within subgroups to be unable to produce the symbolic distance effect. (A, Z) would be compared more slowy than (A, Q) and (A, D) would be compared more slowly than (A, B). Since SDE is strongest at separations of one to five, it is necessary that comparisons within subgroups show this effect.

Because of the symbolic nature of this model, it is possible to restrict the order in which stimuli are processed. Thus, the model can account for Direction effects, and effects arising from instructions (i.e. Congruence and Funnel effects). The grouping nature of the model allows for the use of discrete information (category markers and position tags). It is also able to account for the End-term effect, because of the serial self-terminating searches. However, these same search methods generates only a partial implementation of SDE, which also precludes an explanation for the interaction of Direction and separation. There is no learning method defined for this model, thus it cannot change with experience. Nor can the model explain solution or response variance.

A summary of the effects for which these models can account is to be found in tables 6a and 6b.

2. Conclusions

The first two models defined in this section have been shown to be unsuitable for modelling all of the effects found in relative order judgements and have, therefore, been discarded. The second two models have both proven to be effective in different ways. The Poltrock (1990) model appears to be best able to explain the effects which are hypothesised to arise from an analog representation of the data, while the Klahr, et al., (1983) model is an effective implementation of the discrete information processing aspect.

Chapter Two contains a review of the neuropsychological literature, which offers a structure with which the useful aspects

of these two models may be combined. This structure is then used in the model, which begins development in Chapter Three. This new model is designed to produce all of the relative order effects outlined in section A.

		General Effects		Initial Effects		Exper. Effects		Instruct Effects	
Author	**Theory**	**SDE**	**Com-pression**	**SDE Discont.**	**Direction**	**Zero-term**	**End-term**	**Congru-ence**	**Funnel Effect**
Lewandowsky & Murdock (1989)	TODAM								
Lacouture & Marley (1991)	MVBP	?	?						
Poltrock (1989)	Random Walk	X	X	?					
Klahr, Chase & Lovelace (1983)	Alphabet Subgrouping	1/2		?	X		X	X	X

Table 6a. Relative Judgement Effects Explained by Machine Models. The general effects are those that have been found across stimuli and across exposure amounts. The initial effects are found with alphabetic stimuli, which are considered to represent a lower level of learning than numeric stimuli. The experience effects are thought to arise for either a large amount of or dearth of experience with certain elements of the sequence. Instruction effects are changes in RT with the instructions given to subjects. (?) indicates that the model may be expanded to account for the effect, but it was not designed to do so. (1/2) indicates that the model can explain half of the effect.

Author	Theory	Solution Variance	Response Variance	Learning Changes	Position Tags	Category Info	Interval Data
Lewandowsky & Murdock (1989)	TODAM		X				
Lacouture & Marley (1991)	MVBP		X	X			X
Poltrock (1989)	Random Walk		X	?			X
Klahr, Chase & Lovelace (1983)	Alphabet Subgroup		X		X	X	

Table 6b. Representation Effects Explained by Machine Models. These include subjects' ability to employ more than one solution strategy, the variability in response times across trials, the changes in RT that occur as a experience with a sequence increases, the reports by subjects of using rough position tags, the effect found in multi-digit experiments that for decades not containing the Standard RTs were approximately uniform, and the finding that subjects use interval as well as ordinal data when making decisions. (?) indicates that the model may be expanded to explain the effect but was not designed to do so.

CHAPTER TWO - Neuropsychological Research: Findings, Theories and Implementations

One goal of Neuropsychological research is to hypothesise the functioning and architecture of the "normal" subject by investigating the ways in which that functioning is changed as the architecture is damaged. McCloskey (1992) summed up this goal in the form of a question: "..what must the normal cognitive system be like in order that damage to the system could result in just this performance pattern?". This chapter investigates the patterns of performance found in patients with deficits in number processing, with a view to outlining ways in which the normal sequence processing system could be modelled.

This issue is addressed by examining the findings of the neuropsychological research and the theories it spawned, leading to three implementations. Finally, the implications for a new model of sequence employing these findings, theories and implementations are outlined.

A. Findings

Chapter One addressed findings in behavioural research using numeric, alphabetic and other stimuli. However, most of the neuropsychological research into relative order decisions has been restricted to numeric stimuli. The main focus of this thesis is using ideas about the representation of sequences to construct a model which can be used to represent any stimuli which are sequenced. For this reason, the neuropsychological research focused on in this chapter, which is mainly concerned with the number sequence, will be investigated with a view to extrapolating the findings to other stimuli. It is interesting to note that number series processing can be divided into two subsections - number processing and calculation processing. The first subsection may perhaps be related more to language processing than calculation. This link to language may mean that number processing constitutes a semantic interpretation of order information. To explore this

possibility, the following sections investigate the distinction between number and letter processing, then the distinction between number processing and calculation.

1. Numbers v Letters

In their review of the neuropsychological literature, Deloche and Seron (1987) find that aphasics and alexics are often reported able to perform better in number than language processing (i.e. Gardner, 1974; Albert, et al., 1973, respectively). Patients are able to read aloud, understand and write numbers despite alexia and agraphia (but with preserved abilities to rapidly produce the number-word sequence both forward and backward without error, also to count in discontinuous series [e.g., by threes; Bensen and Weir, 1972]). Deloche and Seron note that patients have shown written and auditory number comprehension even in cases of severe aphasia (Barbizet, Bindefeld, Moaty, and Le Goff, 1967; Ruffieux, et al., 1981); and in Wernicke's aphasia with Gerstmann syndrome (Assal and Jacot-Descombes, 1984). They conclude that, because the reverse pattern of digit- or number-reading impairment without alexia for letters or words also occurred (Grewel, 1973), language and numbers appear to be represented and processed in separate systems that possess a degree of independence.

They also note that the variety of calculation disorders observed (linguistic, spatial, etc.) points to a variety of the representational tools (verbal, visual, etc.) preferentially used by individuals performing arithmetic (Leonard, 1979). They point out that this representational variety explains how Bensen and Denckla (1969) can describe calculation errors which were induced by language disorder (verbal paraphasia), while Spalding and Zangwill's (1950) patient produced calculation error with a left-sided occipito-parietal lesion that grossly impaired the visual spatial pattern of number form commonly handled by the patient.

Deloche and Seron also note the relevance here of cross-cultural studies that analyse the effect of abacus training and of the verbal learning of basic addition and subtraction number facts in Japanese

and Chinese children as compared to the emphasis in Western countries on a more abstract approach to numbers. They cite examples in Singer and Low (1933) and Lawler (1981) showing that apparently the same addition problem can be handled by different procedures depending on its context (arithmetic problems presented auditorally or in writing, or money problems involving pence, i.e. 75 +26 vs. £0.75 + £0.26).

Finally, they note that there may exist a dissociation between written (calculations done with a paper and pencil) and oral (calculations done "in the head" with answers given aloud) calculation, with the former being generally better preserved (Assal, Buttet and Joliver, 1981; Gardner, Strub and Albert, 1975).

McCloskey (1992) has pointed out that one area that has not been adequately addressed in recent work concerns the relationship between mechanisms for numerical and non-numerical processing. A central question in this realm concerns whether numerical processing mechanisms are incorporated within, or are separate from, the cognitive language-processing system. The relationship between numerical and language processing mechanisms has been widely discussed in the dyscalculia literature (Benson and Denckla, 1969), but no clear conclusions have emerged. Relationships between forms of dyscalculia and loci of brain lesions have been extensively discussed (see Kahn and Witaker, 1991, for a recent review). The localization findings are addressed in Section B of this chapter.

2. Number Processing Distinguished from Calculation

Number processing can be related to three areas of neuropsychological research. These areas have been outlined in Deloche and Seron (1987):

1. In the semiological and classificatory approaches to acalculia, the distinction we propose between digit/number and calculation processing echoes the classic distinction between alexia/agraphia for numbers and "true acalculia" or anarithmetia.

2. In the neurolinguistic approaches to aphasia, the question of the dependence or the autonomy of digit-number and letter/word processing has frequently been raised.
3. In lateral difference studies, a differential implication of each hemisphere in number and language processing has often been hypothesised but remains unresolved. This issue is generally situated in the more general theoretical framework of differential hemispheric processing capabilities according to the locographic versus the phonographic surface code of the linguistic units. [Deloche and Seron, 1987, p. 137]

This distinction is supported by the work of Warrington (1982), who investigated patient DRC. This patient could read and write numbers without difficulty, he was able to judge rapidly and accurately which of two numbers was larger, and he could give reasonable estimates of numerical attributes (e.g., how tall is the average English woman?). Testing of basic arithmetic, however, revealed a deficit: in tasks requiring speeded responses DRC was much slower and somewhat less accurate than controls, even for very simple problems (e.g., 5+7).

Warrington interpreted these results as evidence for a distinction between number knowledge (knowledge about meaning and general use of a number) and arithmetic knowledge. Given this distinction, she argued, DRC's performance could be explained at a general level by assuming that his arithmetic knowledge was disrupted while his number knowledge remained intact and further that DRC's performance motivates a distinction within arithmetic knowledge between knowledge of operations and facts.

Further support is offered in a review by a single case study presented by Singer and Low (1933). Their patient was able to read, write, and repeat single- and two-digit numbers, but these abilities contrasted with severe calculation disorders. The patient was unable to solve simple two-digit addition problems if the result was greater than ten and simple two-digit subtraction problems (either in the verbal or in the written modality).

3. Number Processing

Number processing is defined as all processes which involve the number series, including reciting the series, calculation, relative order decisions, numbering objects in a set, etc. It has been divided into several subcomponents based on the findings of dissociations between digital/arabic processing (i.e., numbers in the form 13), verbal/oral processing (i.e., numbers spoken aloud) and written verbal numbers (i.e., of the form "thirteen"). This third category has also been referred to as writing/reading. Dissociations have also been found between lexical (simply the form of the number), semantic (the meaning of the number, i.e., "how many is four") and syntactic (the correct placement of ones, ten, hundreds, etc.) processing.

3.1 Digital/Arabic, Verbal/Oral Processing and Writing/Reading

Number processing has been divided into reading (producing the phonetic form "th-r-ee" from either 3 or three), writing (being given the phonetic form and writing either 3 or three) and oral components, with each component further subdivided into comprehension and production mechanisms, these components and subcomponents have been shown to dissociate in patients with number processing deficits. The evidence for these distinctions is outlined in this section.

Benson and Denckla (1969) presented results from a patient who was able to point to the correct arabic numeral when a verbal numeral was dictated, suggesting intact comprehension of arabic and spoken verbal numerals. However, on tasks requiring production of arabic or spoken verbal numerals the patient was severely impaired. When asked to say or write the answer to simple arithmetic problems, he was often incorrect. Benson and Denckla concluded that their patient showed a dissociation between comprehension (intact) and production (impaired) for both arabic and verbal numerals.

McCloskey, et al. (1986) probed production of spoken verbal numerals in patient HY. Results from several numeral-processing tasks suggested that HY's errors in reading arabic numerals aloud reflected a deficit in production of spoken verbal numerals, but no impairment in comprehension of arabic numerals. These results were supported when HY performed well on tasks requiring arabic numeral comprehension, but made lexical substitution errors in tasks requiring spoken production of verbal numerals.

McCloskey, et al. (1990) reported results from a patient whose errors in reading arabic numerals were quite different from those of HY, patient JE. They found that approximately a quarter of JE's errors were within-class lexical substitutions (e.g. stimulus 5097, response "five thousand ninety-*five* "). The remaining three quarters were "quantity shift" errors, in which a quantity associated with one power of ten in the arabic stimulus was associated with another power of ten in the verbal response. For example, JE read aloud 8900 as "eight thousand ninety". On tests of arabic numeral comprehension JE's performance was perfect, suggesting to them that his errors in reading arabic numerals aloud stemmed from disruption of the system for producing numbers verbally.

Sokol and McCloskey (1988) studied a patient, JS, whose overall error rate was 27% for a spoken production task and 18% for a written production task. When he was tested for arabic numeral comprehension however, his performance was excellent. They suggest that the errors in the spoken and written verbal numeral production tasks reflected a deficit in the production of the verbal responses, and not in the comprehension of the arabic stimuli.

The pattern of errors in his written production task was virtually identical to his pattern in the spoken production task, they took this as support for the assumption of a syntactic frame generation process common to spoken and written production of verbal numbers. Whereas JS made lexical substitution errors for 13% of the stimuli in the spoken production task, he made no lexical errors in the written production task. Sokol and McCloskey interpreted

this dissociation by assuming that JS was impaired in retrieval of phonological number-word representations from the phonological output lexicon, but not in retrieval of graphemic number-word representations from the graphemic output lexicon.

Cohen and Dehaene (1991) studied a French patient YM. They reported that this patient's results indicated an impairment in verbal number production and a deficit at a level of representation involving spatially arrayed visual representations of digits. They also found that YM had a deficit in comprehension of arabic numerals. For example, the further to the left a digit the more difficult YM found translating it into an abstract quantity representation. They did find one result which argued against a deficit in arabic numeral comprehension. When YM was presented with pairs of 1- to 4-digit arabic numerals and asked to judge which number in each pair was larger, he was 100% correct, but when asked to read aloud 88 of the numeral pairs, he made 38 errors, with six of these errors resulting in a reversal of the relative magnitudes of the two numbers. Cohen and Dehaene argue that YM's deficit is in the digit identity retrieval process, but McCloskey (1992) points out that in their description the digit "identities" are not explained and therefore the representation retrieved is unclear. McCloskey further states that they do not say if this deficit is to apply to verbal numeral production in general, or only to arabic-to-spoken-verbal transcoding.

McCloskey concludes that nearly all of the findings from patient YM can explained by assuming deficits in both arabic numeral comprehension and verbal numeral production. However, he concedes that the only result raising difficulties for this interpretation is YM's perfect performance in the numerical comparison task, because these findings suggest intact comprehension of arabic numerals. However, he notes that not all of the reading-aloud errors were due to an arabic numeral comprehension deficit, but that some of these errors reflected a deficit in verbal number production.

McCloskey, et al. (1985) found that subject VO could judge which of two numbers was larger. However, when presented with a written verbal number (e.g. seven thousand forty) and asked to write the number in arabic form (e.g., 7040), VO performed poorly for numbers above one thousand (e.g., 700040). They interpreted this as the result of intact comprehension, with deficient production.

Another of their subjects, AT, showed excellent performance (over 99% correct) when she was presented with a written verbal number (e.g., six thousand seven hundred five) and was asked to write the number in arabic form (e.g., 6705). However, when written verbal numbers (or arabic numbers) are presented and AT was asked to read the numbers aloud she showed a clear impairment, producing only about 80% correct responses. They point out that AT's excellent performance in writing arabic numbers from written verbal stimuli indicates an intact ability to produce arabic numbers and an intact comprehension of written verbal numbers. This led to the conclusion that the errors in reading verbal numbers aloud cannot readily be attributed to an impairment in comprehension of the written verbal stimuli. Instead, they appear to reflect a deficit in producing the spoken verbal responses. They also note that these errors cannot be attributed to some peripheral production deficit (e.g., an articulation problem) - the errors take the form of the incorrect number word produced in place of the correct word and hence clearly stem from impaired processing within the number-processing system.

McCloskey, et al. (1985) also investigated patient HY, who produced 100% correct responses in judging which of two arabic numbers was larger (e.g., 6 vs. 7; 405,034 vs. 400, 534), suggesting intact arabic number comprehension. However, he performed poorly on magnitude comparison judgements for written verbal numbers (e.g., six vs. seven; four hundred five thousand thirty-four vs. four hundred thousand five hundred thirty-four). Another of their patients, patient K, in contrast, performed without error in judging which of two number words was larger (e.g., four vs. five), but showed near-chance performance on magnitude judgements for

digits (e.g., 4 vs. 5). They conclude that these patients show a dissociation between arabic and verbal number comprehension.

Deloche and Seron (1987) found data that point to a possible dissociation between spoken, read and written numbers. They reported that of the subjects with digit alexia, 20% presented no oral calculation disorders and 5% no writing calculation disorders, and, of those with digit agraphia, 8% presented no oral and 10% no written calculation disorders.

Deloche and Seron report a French study conducted by Collignon, Leclerq, and Mahy (1977) on a group of 26 unilaterally brain-lesioned subjects. There was one case of oral acalculia without number alexia, and the reverse, one case of number alexia without oral acalculia. They also observed dissociations in intramodality tasks. Two patients demonstrated written acalculia without number alexia, three patients demonstrated number alexia without written acalculia, and finally, five patients demonstrated an apparent dissociation between number agraphia and written acalculia.

Although they did not address this question directly, Grafman, et al. (1982) detailed the presence of digit-written calculation difficulties in subjects otherwise able to write digits corresponding to a set of various tokens and to perform a magnitude comparison task on pairs of two-digit numbers. However, Deloche and Seron (1987) noted that, because the analysis of calculation errors was mainly designed to examine the incidence of spatial factors, the kind of dissociations exemplified in such cases were not specified.

An example of the dissociation between comprehension and production processes is reported in McCloskey, Macaruso and Whetstone (1992). They studied lexical processing of numerals in patient RH. Their examination of error patterns across tasks led to the conclusion that RH had experienced damage to independent numeral comprehension and production processes. For example, on three tasks requiring written verbal responses RH showed a

virtually identical pattern across tasks in error rates and types of errors for individual words. They point out that these results support the assumption that the same disrupted numeral production process was employed in all three tasks.

They also found that the error data provided evidence for the involvement of semantic representations in the transcoding tasks. For each of the tasks showing substantial impairment RH displayed a very strong tendency to produce incorrect digits or words similar in magnitude (and not visually or phonologically similar) to the correct digits or words. For example, in tasks involving production of spoken verbal numerals, the incorrect number-words produced by RH were consistently close in magnitude to the correct words (e.g. *eighty-three* read aloud as "ninety-three" and 47 read as "forty-six"). They believed this result to support the assumption of a functional architecture in which comprehension and production processes communicate via abstract semantic representations.

The distinction between verbal and idiographic material was emphasised by Coltheart (1980). He investigated both pathological (split-brain subjects, alexics) and normal subjects for tachistoscopic recognition of numbers or magnitude comparisons in Stroop-like conditions. For instance, incongruence between the physical size of number symbols and their values was shown to affect numerical comparative judgements with digital but not with alphabetic English forms (Besner and Coltheart, 1979; but see Peereman and Holender, 1985, for contradicting results).

The results of this section are summarised in table 7. Dissociations between arabic numeral comprehension and production, verbal comprehension and production and written comprehension and production appear to be very strong, as do those between impaired arabic numeral production and intact verbal and written processing. Gaps in the table are noticeable for all other dissociations.

Intact ->		Arabic - Digital		Verbal - Oral		Written - Read	
Impaired		Comp	Prod	Comp	Prod	Comp	Prod
Arabic - Digital	Comp	XXXXXX XXXXXX		D&S	D&S,JE	D&S,K	D&S
	Prod	B&D YM, RH	XXXXXX XXXXXX	D&S	D&S	D&S	D&S
Verbal - Oral	Comp			XXXXXX XXXXXX			
	Prod	YM, HY JE, JS	AT	B&D	XXXXXX XXXXXX	AT	
Written - Read	Comp	HY				XXXXXX XXXXXX	
	Prod	JS VO					XXXXXX XXXXXX

Table 7. Dissociations Documented in the Number Processing Review.
D&S refers to findings listed in Deloche and Seron (1987), B&D refers to findings reviewed in Bensen and Denckla (1969 - reported in McCloskey, 1992). HY, JE, JS, VO, YM, AT, K and RH are patients, whose results are reviewed McCloskey (1992).

3.2 Lexical Processing as Distinct from Semantic Processing

A second distinction found in the number processing literature is between lexical and semantic (syntactic) processing of numerals. On the one hand subjects seem able to comprehend digits, but fail in the production or comprehension of "grammatically" correct larger numerals.

Singer and Low (1933) reported a patient, whose performance in the production of arabic numerals revealed a dissociation in which lexical processing was intact but syntactic processing was impaired. In writing arabic numerals to dictation he was uniformly correct for 1- and 2- digit numerals, but only the non-zero digits were consistently correct for larger numerals, responses were of the wrong order of magnitude 242 was written as 20042.

Another approach is that of Dehaene (1992), who postulates two largely distinct number-processing pathways - one processing numbers as symbols and the other transducing then into approximate quantities.

He reports the findings from patient NAU. He found that, the only knowledge that NAU could access about a number was its approximate magnitude. NAU was almost 100% correct in comparing 1- and 2- digit numerals, even when he could not read the stimuli aloud. The only errors he produced were in comparing two close quantities (i.e. 4 and 5), which is the same pattern of results generally produced by normals (Chapter One). But he was at chance level for judging whether a given digit was odd or even. Dehaene found that his number knowledge was similarly affected. For instance, he would state that a dozen eggs were 6 or 10, or that a year was made up of about 350 days, again lacking the exact knowledge but preserving the correct order of magnitude. Dehaene also noted that he could read "three" by counting on his fingers, which indicates that he knew when to stop counting.

McCloskey, et al. (1985) Also report a patient whose processing, they believe, shows a Lexical/Syntactic distinction. They report that patient VO showed intact lexical but impaired syntactic processing in the production of arabic numbers. VO selected the appropriate digits to represent the individual magnitudes in the stimulus, but was unable to assemble these digits into an arabic number of the appropriate form.

They contrast this with the number-production deficit evidenced by Benson and Denckla's (1969) patients, which apparently involved lexical but not syntactic processing. Benson and Denckla present three case studies of patients with similar lexical deficits in the production of spoken verbal numbers. For example, RR who is severely impaired in reading arabic numbers aloud, shows excellent performance in judging which of two single-digit or multi-digit arabic numbers is larger. Patients responded with a 0-9 for single digit numbers 10-19 for teen numbers, 20-29 for two-digit numbers etc. Only one of the three patients produced errors close

in magnitude to the correct response. They take this pattern of errors in the three patients as strongly suggestive of a production lexicon that is organised into three functionally distinct classes, ONES, TEENS, TENS. First the class is chosen, then the item within the class. They explain that this pattern of performance described suggests that the patient is impaired in producing the individual elements of a number (that is, the individual digits, or the individual number words), but is intact in assembling the elements into a number of the appropriate syntactic form and order of magnitude. They report that dissociations of lexical and syntactic processing may also be observed in the production of verbal numbers.

This section reported on the distinction and possible dissociation between lexical and semantic or syntactic processing of numbers and the division of the number series into three categories; ONES, TEENS and TENS.

4. Calculation

The calculation mechanism itself appears to be divided into two subsections. The first is dedicated to the accumulation of "facts" (i.e. 1 + 2 = 3) and the second contains the procedural information for solving calculation problems (i.e. 'add the right most column first' in a multi-columnar arithmetic problem). The distinction between these two subsections is well documented as outlined below. The debate about the form of representation used in the "fact" section is also discussed.

Sokol, McCloskey, Cohen and Aliminosa (1991) studied a patient PS who was clearly impaired on single-digit multiplication (20% errors). They found that these errors were not attributable to numeral comprehension or production deficits, but instead reflected a deficit in retrieving multiplication table facts. PS presented with a clear dissociation between retrieval of arithmetic facts (impaired) and execution of calculation procedures (intact).

Spiers (1987) reported that Warrington (1982) systematically investigated oral addition, subtraction and multiplication operations with digits between 1 and 19, with a patient who often knew the approximate answer or was aware that it should be an odd or even number almost immediately, but could not reliably access the exact solution. Warrington found that at times the patient reverted to counting by ones or twos in order to reach an answer and was invariably correct if permitted to calculate slowly and check his solution. She reports that the patient was not entirely acalculic, but his error rates for the various types of verbally presented computations ranged between approximately 2% and 24%, in comparison to an error rate of zero for control subjects. His error rates increased when the order of presentation of digits was altered. He had no deficit in the time taken to determine which was the greater of two numbers.

Warrington concludes that her patient's calculation deficit represented a failure in direct semantic entry access for number facts. Based on the variability in his performance, she proposes that the entries themselves were intact and that the patient had a normal calculation strategy, but with faulty access.

Deloche and Seron (1987) describe a patient with a frontal lesion who, in addition to difficulties in tasks like counting, solving arithmetic problems, and comprehending the digit positional value system, presented some difficulties in writing numbers and a preserved ability to do simple addition and subtraction. This indicates that simple addition and subtraction could be located separately from counting, solving other arithmetic problems, comprehending the value of digits from their position in a number and writing numbers.

They also refer to a case presented by Benson and Weir (1972) of a patient with anarithmetia who had difficulties only with written multiplication and division. They note that this patient could count forward and backward rapidly, and could read, write, and recognise all numbers. His calculation abilities, which were assessed with formal oral, written, and multiple-choice tests of computation

ability, indicated no problem with addition or subtraction but almost total failure with multiplication and division. They found, in analysis of the errors, the indication "that the calculation difficulties did not result from an inability to retrieve multiplication tables or from with the addition part of the multiplication procedure, but specifically with the multiplication steps of multi-digit multiplication problems"(Deloche and Seron, 1987). This indicates that multiplication and division may be carried out separately to addition, subtraction and counting, among other processes.

In a further description of the separate addition processing system, Dehaene (1992) reports on the investigation of patient NAU, who could not solve simple addition problems but could estimate the results. This ability allowed him to determine if presented problems were wrong i.e. 2 + 2 = 9. Contrary to normal subjects, NAU did not response faster or with less errors when the addition operands were small or with tie problems such as 3 + 3. His performance seemed to depend almost exclusively on the degree of falsehood of the addition. When a grossly false result was proposed, he responded so rapidly that the use of counting seemed implausible. Dehaene takes this result as a reflection of a basic magnitude estimation device available in number processing, which operates separately from the process of solving the problem itself.

Examining the problem solving system by investigating the distinction between a fact retrieval system and one in which a procedure for determining the answer to the problem is implemented, Geary (1993) summarised his studies by suggesting that the primary deficit associated with anarithmetia is difficulty in arithmetic fact retrieval, although difficulties in the execution of arithmetic procedures are sometimes (e.g. Ashcraft et al., 1992; Spiers, 1987) but not always (e.g. Sokol et al., 1991) associated with these retrieval deficits. Moreover, he reports that fact-retrieval deficits typically co-occur with verbal deficits (McCloskey, Aliminosa and Sokol, 1991; Richman, 1983; Rourke and Strang, 1978; Sokol, et al., 1991). The relationship between

procedural deficits and verbal skills is less clear. These findings seem to indicate that fact retrieval in calculation is stored in verbal memory like other semantic information.

An aspect of calculation that may interfere with the interpretation of deficit results is the understanding of mathematical operation symbols. Ferro and Botelho (1980) studied two patients (AL and MA), both with selective deficits in comprehension of written operation symbols (e.g. +, x). They found that when single- and multi- digit arithmetic problems were presented in written form both patients often performed (correctly) the wrong operation. McCloskey points out that the fact that incorrect operations were performed correctly, suggests that the patients were intact in comprehension and production of arabic numerals, in retrieval of arithmetic facts, and in the execution of calculation procedures. Thus, he reasons, the performance pattern suggests a deficit in comprehension of operation symbols, and indeed several tests of operation symbol comprehension revealed clear deficits in both patients. He notes that, the operation comprehension deficit was specific to the written operation symbols. When arithmetic problems were presented aurally, the patients performed well, and in particular had no difficulty comprehending the spoken operation words "plus", "times", and so forth. These findings indicate the operation of separate verbal and written operations comprehension mechanisms, which could lead to difficulty in determining that the problem stems from the operator and not the operation performed.

Further difficulties in interpretation of results have also been reported. Deloche and Seron (1987) restate a reminder by Bensen and Denckla (1969) that some calculation deficits may be due to "secondary effects" of transcoding problems with numbers and digits. Bensen and Denckla report on a subject, who in answer to the written problem 4 + 5 said "eight", wrote "5", and chose "9" from the multiple-choice list. A second subject having to solve the problem 22 minus 17, repeated it as "22 minus 18" , but gave the correct answer "5" and, when asked to write and solve the problem 4 plus 7, wrote the problem correctly, gave the incorrect answer "16" orally, but wrote the correct solution "11". They contend that

these differences in response do not relate to intact processing in one domain, which is impaired in another, but that the processing is correct and the translation from abstract representation to some domains is correct, while to others it is not. The debate about whether numerical processing is done in domain specific representations, or in an abstract representation which is then translated to and from the domain required, is addressed below in the "Theories" section of this chapter.

This section presented evidence for the division of the calculation mechanism into two sections. The first section would be composed of arithmetic facts for retrieval, while the second contained procedures for solving arithmetic equations. However, it must be remembered that the interpretations presented here may have ignored the various extraneous causes for patient's difficulty with calculation.

B. Localisation

The localisation of numerical processing within the brain is hampered by the need to account for the interaction of the various components outlined in the above sections. It may be difficult to differentiate deficits which arise from damage to an individual component from those which arise from damage to the connections between components. These connections may be long-range as the evidence below points to some processing components being based in the right hemisphere while others may be based in the left hemisphere.

Benton (1981) has defined cerebral localisation as:

> The identification of the neuronal mechanisms within the brain that are responsible for the mediation of defined behavioural capacities. These mechanisms are never located in a single discrete neural aggregate but are always defined by the dynamic sets of inter-relationships among neuronal aggregates. Lesional localisation consists of identifying those specific junctures within a system of interrelationships that derange the system with sufficient severity to impair its mediating functions. [Benton, 1981]

Spiers (1987) explains that the ***centrist*** view of localisation position would argue, as Henschen (1920) did, that there is a single consolidated calculation centre. This seems impossible when the findings listed in section A are considered, because calculation processes appear to be separated with some indication the arithmetic facts are stored with other semantic memories.

Spiers states that "modern variations on this theory are implicit in much of current neuropsychological research and tend to be expressed as a parcelling out of a given function into separate smaller functions, each with its own centre or localisation". He explains that for calculation this would mean that the spatial algorithms are located in one cerebral centre, column alignment in another, table values in a third, number reading in yet a fourth, and so on. All that is then required is a CT Scanner with which to identify the lesions corresponding to these centres.

Spiers notes that at the opposite extreme would be the ***holistic, equipotentiality argument*** (Lashley, 1929) that a lesion almost anywhere in the brain may cause some disruption in calculation. However, this is contradicted by the evidence suggesting that lesions in the posterior aspect of the left hemisphere are far more likely to produce calculation deficits and deficits of greater severity than lesions at any other site.

In contrast to the centrist or equipotential views, Spiers remarks, a ***network approach*** would conceptualise calculation in terms of several component processes. He suggests that these component processes are as yet unidentified and may or may not correspond to component aspects of the number/calculation system (Grewel, 1969). Although each process may have a distinct localisation in a certain site, it is also interconnected with all of the other sites in the integrated network subserving calculation and may be mediated at different levels within the nervous system. He reasons that lesions at different sites might, therefore, be expected to give rise to different types of error, whereas lesions at different levels may

give rise to variability in the performance of certain components of the calculation process.

"The network approach differs from the parcellated centrist position in that each site may also contribute the component process it is capable of performing to functions other than just calculation, and because each site relies on more fundamental mechanisms and structures, such as those underlying arousal or visualisation, in order to ensure its intact functioning" (Spiers, 1987). He also recognises that these same neural substrates probably also help to constitute sounds into digits, contribute to the correct sequencing of these digits to form numbers, and at yet another level, may partially determine the sequence or order in which those numbers are manipulated to correctly complete a calculation.

Similarly, the retrieval of table values, such as 4x3 or 7x8, which appear to be automatically accessed from semantic storage (Ashcraft and Battaglia, 1978), may be disrupted by lesions to component sites that partially control the retrieval of other semantic categories of information and may overlap with memory as well as with spatial or sequencing networks. "It is likely that different approaches to localisation may have validity depending on the function under consideration. Thus, visual acuity within segments of the visual fields is organised in striate cortex according to the centrist point of view. In contrast, generalised attributes such as intelligence, creativity, or personality may well follow the equipotentiality model of organisation. On the other hand, functions such as directed attention, language and memory may be organised according to the network approach. (p320)" Mesulam (1981). Spiers believes calculation should be added to this last list.

According to Spiers, one final implication of this approach to the analysis of acalculia is that different components of this network may be active in the acquisition of calculation skills than are responsible for their maintenance once learned. He finds this particularly relevant to Guttman's (1936) contention, based on

work with children, that it is the loss of the fundamental processes of quantity and magnitude estimation that forms the basis for calculation disorders. Spiers notes that although there can be no question that spatial abilities and schema underlie the development and acquisition of number concepts (Wohlhill and Lowe, 1962), it is not clear that these schema participate actively once the number system is learned and incorporated.

Spiers points out that children whose right hemisphere has been lesioned may not be able to develop the spatial relations and schema required to understand the concepts underlying the number system and calculation. In children with right hemisphere congenital injury, the ability to appreciate aggregates and manifold may be disrupted, and number conservation, rather than emerging as an apparently intuitive outgrowth of development, may require laborious instruction. In the left hemisphere syndrome, on the other hand, the child may have highly developed spatial skills but may be incapable of efficiently sequencing the digits required for calculation or unable to store the necessary number facts in semantic memory.

Spiers (1987) reports that despite these efforts, no clear statement of the relationship between calculation ability and brain function currently exists. In large measure, this is due to a lack of uniformity in the various methods employed to examine patients with calculation disorders. Regardless of perspective, however, it is clear that these is a difference in the distribution of errors that are more common to each type of aphasia, with the Broca's errors resulting primarily from grammatical difficulties and the Wernicke's difficulties centering primarily around sequential organisation of numerals.

Although Deloche and Seron (1987) felt that this pattern suggested a parallel between the dissolution of linguistic skills and digit transcription difficulties in the number system, they did not speculate on the nature of the relationship underlying this observation. However, it would seem likely that the similarity between the deficits observed in the linguistic and number realms

is due to the functional specialisation of the anatomical structures that have been damaged. Grafman, et al. (1982) conclude that even though visuoconstructional and language skills undoubtedly contribute to intact calculation performance, left posterior lesions are particularly prone to produce impaired calculation abilities independent of these disorders. They also feel their findings are generally compatible with Henschen's (1920, 1926) hypothesis that the angular gyrus in the left hemisphere has particular significance for calculation.

Deloche and Seron (1987) reason that the absence of any syllabic effect in latencies for naming two-digit numbers presented in arabic form seems to favour the hypothesis of a direct recognition procedure, without a syllable-dependent implicit speech process (Henderson, Coltheart, and Woodhouse, 1973), which could be handled by right-hemisphere mechanisms (however, see Klapp, 1974, for contrary results).

These localisation findings imply that general sequencing and fact retrieval is grounded in the left hemisphere, while the remaining number processing components are grounded in the right hemisphere.

C. Theories

Most researchers appear to agree that the number processing system can be divided into several functionally decomposable subsections, with no clear agreement on what these subsections should contain. Further debate centres on the form of processing within each subsection and the way in which the various systems pass information between themselves.

A survey of the literature by Gonzalez and Kolers (1987) suggests that three classes of models can be identified as offering descriptions of the underlying operations. These classes are analog, counting and network models. They are the same classes as discussed in Chapter One and contain many of the same models. These classes have been supplemented by alternative explanations

of number processing that do not rely on separated and dedicated number processing systems.

1. Analog models

The defining characteristic of the analog models developed in response to the neuropsychological data and those developed in response to the behavioural data is, as Gonzalez and Kolers (1987) note, the fundamental nature of the notion of the number line. It is supposed that numbers extend from some value to some value (minus infinity to plus infinity in theory) and that in performing the operations of mental arithmetic, especially in addition and subtraction, people actually use features of the number line.

Gonzalez and Kolers (1987) note that the notion of the number line stems from Restle's (1970) claim for spatial cognition, which has been amplified in some of Shepard's (1981) claims that space is the fundamental basis of cognition and provides the base for much else besides measurement and quantitative comparison. They reason, however, that the spatial metaphor may not be the necessary basis of cognition, but only one alternative - a strategy rather than a fundament (Kolers, 1983). This leads to the idea that claims for spatial representation as a necessary basis underlying mental arithmetic or numerical comparison may be too narrowly selective, which would limit the explanatory powers of the analog models.

2. Counting Models

These models (i.e., Parkman, 1971) were exposed as inadequate in Chapter One for the main reason that they would predict the opposite of the SDE found. However, Gonzalez and Kolers (1987) point out that these models may have some merit in the idea that the influence of the relations underlying the variables of the composition of the reaction varies according to the way the person uses the information supplied in the experiment. Rather than some purified mental state, the behaviour measured reflects both the

means of acquisition of information and the nature of the produced reaction.

3. Network Models

Stanzyk (1981) found that simple sums such as 3 + 6 = 9 were verified faster after tests of sums such as 4 + 5 = 9 than after other sums. The argument was that the quantity 9 in this case was facilitated in retrieval or primed by its prior elicitation. This finding has been used to develop models in which representations of problems and answers are connected in what are referred to as networks. Generally, these connections take the form of a table of solutions which is indexed by the operands of the problems. Thus, he is not referring to the models described as "network models" in Chapter One.

Gonzalez and Kolers (1987) point out that the simple analog models of Restle, among others, assumed that the mind represented quantity in some attributed mental space, whereas network models seem to have internalised the analog idea of such a space and proposed that operations are carried out upon this mental entity much as they would be if the objects and the space they occupied were external to the person. They explain that Kolers and Smythe (1984) referred to the confusion of physical events and mental events as the confusion of consensual symbol systems with personal symbol systems.

Perhaps in fact the greatest weakness of network models, as identified by Gonzalez and Kolers (1987), is that they emphasise stored knowledge so greatly that they give little if any attention to the procedures by which the knowledge is acquired, combined and used, or how it is modified (they are talking here about the network models that use tables of operands and solutions).

4. Alternative Proposals

Gonzalez and Kolers (1987) point out that investigators seem to agree that although young children count, or perform related

arithmetic operations in solving simple problems, practised adults are more likely to show arithmetic knowledge as part of their general knowledge structure. If arithmetic is but another form of knowledge, they reason, its expression should be sensitive to many of the conditions that affect knowledge representation generally.

4.1 Coding by Language

A number of authors have reported on the anecdotal account given by bilingual persons that they perform their mental calculations in the language of acquisition and practice (Kolers, 1968; reported in Gonzalez and Kolers, 1987). Many report a ready ability to translate among sentences in the two languages they know, and a far lesser ability to perform the operations of arithmetic other than in the language of acquisition. This indicates that arithmetic facts and procedures may be stored in a language dependent structure and not in a separate number processing system.

Gonzalez and Kolers (1987) reason that the same sort of specialisation might be analogised to musical performance. The ability to play the clarinet, however skilfully, does not by itself aid in playing the piano or violin. Basic notions of tempo, rhythmic structure and composition may be acquired with one and transferred to the other. Thus, by analogy, basic notions of quantity, equality, and relation may be acquired in learning arithmetic in one language and be transferred to another. In neither case, however, do the special skills underlying the mental operations controlling the behaviour transfer - each skill is learned as an individual.

Kolers (1964) did a simple experiment in which several people, bilingual in different languages, were taught to say the alphabet backwards in either their native language or English. They were subsequently taught to say the alphabet backward in the untrained language. Kolers reasoned that if the skill of inverting a list were a general mental capability, the degree of transfer of learning from practice in one language to test on the other should be everywhere the same. In fact, he found, the extent of transfer was related to the degree of similarity between the English alphabet and the

native alphabet. The transfer was very high between English and French or German, all of whose alphabets are very similar. While transfer was moderate between English and Arabic, the latter of which has initial letters of *alif, be, se,* and a middle section of *kef, lam, mim, nun.* Finally, transfer was effectively zero between English and Korean or Thai, whose alphabets have no symbols or names of symbols in common.

Gonzalez and Kolers point out that mental arithmetic also requires the manipulation of symbols. They conclude that the experiment with alphabets suggests that skill at the manipulation of symbols will be intimately tied to the means learned for instantiating the symbols.

In further support of the contention that calculation facts and procedures are stored in language specific codes, Marsh and Maki (1976) required bilingual subjects to speak the response that resolved equations such as 7 + 3 = ?. They found that people were generally faster to respond in their preferred language than in their second language.

4.2 Common Representation

Since the evidence clearly points to number processing as divided between several component processing system, a debate has ensued as to how information is passed between the components. On the one hand, researchers have proposed an asemantic representation which is used by all components, on the other this common representation is considered to be semantic in nature. Two other views have also been presented. The first is related to the analog processing model and proposes some association between an underlying number representation and a continuous visual form. The second proposes that there is no common representation that processing is tied to the form in which it is originally learned

semantic vs. asemantic

The numerical distance effect (SDE) has been obtained for stimuli in a variety of forms, including arabic digits (Moyer and Landauer, 1967; Sekular, Rubin and Armstrong, 1971), written number-words

(Foltz, Poltrock, and Potts, 1984), patterns of dots (Buckley and Gillman, 1974), and Japanese kanji and kana numerals (Takahashi and Green, 1983). McCloskey notes that these results have been interpreted as evidence that regardless of the form in which stimuli are presented, performance on the task is mediated by abstract quantity representations that reflect magnitude relations among numbers. However, he concedes, some researchers (Besner and Coltheart, 1979; Takahashi and Green, 1983; Tzeng and Wang, 1983; Vaid, 1985; Vaid and Corina, 1989) have argued that performance in the numerical comparison tasks is influenced by the format of the stimuli, at least with respect to phenomena other than the numerical distance effect, (i.e. the "size-congruity" effect). McCloskey (1992) concludes that at present the nature of the representations underlying performance in numerical comparison tasks remains an open issue.

In support of a common representation, McCloskey reports the results of patient PS. In testing PS on single-digit multiplication Sokol et al. (1991) manipulated the format of stimuli and responses. Results from several tasks indicated that PS was intact in comprehending and producing numbers in each of the tested formats (arabic numeral, written verbal numerals and dots). They concluded that PS's striking consistency in performance across the various forms of stimuli and responses strongly suggests that as stimuli and response formats were varied, the arithmetic fact retrieval process remained constant - that is, the same representations of problems were used to address the same representations of answers.

Although, McCloskey notes, they have no definitive evidence to offer on this point, the nature of PS's errors provides tentative support for the assumption that abstract quantity representations underlie the fact retrieval process. PS's fact retrieval errors were predominantly "operand" errors. PS's erroneous responses were correct for problems with operands numerically (as opposed to phonologically or visually) similar to the operands in the stimulus problem. This *operand distance effect* is also observed in errors made by normal subjects (Campbell and Graham, 1985; Miller,

Perlmutter and Keating, 1984). This numerical similarity leads McCloskey to conclude that the common representation contains semantic information about the numerals in question. In this case semantic information takes the form of an approximate value for the numeral.

Deloche and Seron (1987), however, have suggested that transcoding, or transfer between different numerical representations, is accomplished by "asemantic" transcoding algorithms that translate from one numeral form to another without computing a semantic representation. That is, some transfer may occur without the need to interpret the number (i.e., to transfer from "three" to "3" all that is needed is the lexical information about the two number forms). They do not posit asemantic transcoding algorithms *instead* of the numeral comprehension and production processes, but *in addition to* these numeral comprehension and prediction processes.

In support of an asemantic representation, McCloskey notes that, at the least, translation between phonological and graphemic number-word representations may presumably be accomplished through the grapheme-phoneme and phoneme-grapheme conversion process postulated for words in general by most models of reading (e.g. Patterson and Morton, 1985) and spelling (e.g. Ellis, 1982), although these processes would only reliably generate correct translations for number-words with regular spelling-sound correspondences, such as *seven,* and not for irregular words such as *one*, as these rely on access to semantic information as well. The same explanation would hold for phonology-to-orthography translation and vice versa (e.g. Bub, Cancelliere and Kertesz, 1985; Goodman and Caramazza, 1986).

At a slightly higher level, McCloskey notes, it may be hypothesised that people learn some basic asemantic transcoding rules for mapping between "simple" multi-digit arabic numerals - that is, numerals comprising a non-zero digit followed by one or more 0's - and their verbal counterparts (e.g. x000 ↔ x thousand).

McCloskey (1992) sees the issue as whether there is reason to postulate algorithms dedicated to translation of numerals from one form to another, when numeral comprehension and production processes sufficient to accomplish the translations must in any case be postulated for other reasons. He points out that the straight mapping from arabic to verbal numbers is quite complex, thus semantic encoding might be a simpler process.

McCloskey points out that Deloche and Seron's (1987) discussion illustrates the asemantic transcoding algorithms must be comparable in complexity to the (semantic) numeral comprehension and production processes postulated by the McCloskey et al. (1985) model. Indeed, he states, the arabic-to-verbal transcoding algorithm proposed by Deloche and Seron (1987; see also Cohen and Dehaene, 1991) is similar in many respects to the verbal numeral production algorithm sketched by McCloskey et al. (1985) and developed in more detail in McCloskey et al. (1986).

McCloskey (1992) concludes that clarification of the rules of semantic and asemantic processes in numerical cognition will require not only additional data, but also (and perhaps more importantly) more detailed and explicit formulations of the various theoretical positions.

Visual Representation

Seron et al. (1992) discuss subjects who when they thought of the number series, the numbers "..show themselves in a definite pattern that always occupies an identical position in their field of view with respect to the direction in which they are looking"(Galton, 1883). These visual patterns called by Galton the "number-form" (hereafter NF), consist either in a simple line with or without shifts in orientation or take more complex forms such as rows or grids. They may be coloured, present changes in luminosity at some locations or occupy different planes.

As for the importance of NFs, Seron et al. note that even if they are frequent, NFs may well constitute a peripheral aspect of cognitive functioning with no pertinence for the understanding of the basic

aspects of number processing and calculation. At present, they have no solid data to reject or accept this argument but they advance some hypotheses to orient future research. They underline that NFs are associated with specific semantic linear orders, and emphasise that the production of the conventional sequence of numbers is one of the most crucial competencies underlying basic counting abilities, which constitute a central skill for the development of subsequent mathematical cognition in children (Fuson, et al., 1982; Gelman and Gallistel, 1978). In this way, they contend, the NF could constitute a spatial medium in which the number sequence is represented, maybe because of some difficulty encountered for the coding and learning of such a sequence by language alone. From the existence of some structural similarities (left to right progression of the number magnitude) between NFs and the number-line they contend that an NF may be the visual realisation of the number-line proposed in analog models to underlie number processing.

No Common Representation

Gonzalez and Kolers (1987) point out that it may be that there is no common representation on which numerical processing is carried out. They take the view of Kolers and Smythe (1984), who concluded that "mental operations cannot be divorced from the symbols on which the operations are carried out". Gonzalez and Kolers carried out a series of experiments, using bilingual subjects, to examine this issue.

Gonzalez and Kolers (1982) conducted an experiment in which arabic and roman numerals were used separately and in combination (controlling for familiarity with roman terms) for presenting simple arithmetic problems for subject verification. They found that response times varied systematically with the number of roman terms and their placement in the equation. They summarised that the subjects knew the value of 3 + 4 in arabic units, but did not know the value of three plus four in mentalese. To put it another way, they conclude, the mental operations are tied to the symbols in which the quantities are expressed.

Sokol et al. (1991) questioned the Gonzalez and Kolers (1982, 1987) conclusions based on their lack of a specific formulation of how processing differed across representations. Sokol et al., presumed that Gonzalez and Kolers did not intend to suggest that addition facts were stored separately in the two forms, arabic and roman, Yet if this is not the intended argument it is difficult to imagine what is intended. They also suggest that Gonzalez and Koler's findings could be interpreted simply by assuming that roman numerals take longer to comprehend (i.e. translate to abstract internal representations) than arabic digits. However, Gonzalez and Kolers had controlled for this variable by drilling subjects on roman numerals until they were able to name them at a speed within 10% of that at which they could name arabic numerals. Thus, if the abstract representation underlies articulation roman numerals were encoded just as quickly as arabic numerals.

Also in support of format-specific representations is the cross-lingual studies of Marsh and Maki (1976) and McClain and Huang (1982). These studies found that responses were faster in the preferred language than in the non-preferred language. However, the slope of the function relating RT to number of arithmetic operations was the same for both, suggesting that the arithmetic fact retrieval process was the same in both conditions. As Marsh and Maki (1976) pointed out, this pattern of results in consistent with at least two interpretations. First, subjects may have translated the problems into the preferred language and carried out the calculations on representations tied to this language. Second, subjects translated the problems into abstract language-independent representations and then translated the answers into the preferred or non-preferred language for output, with translation slower to and from non-preferred language.

Bertin (1980) wrote a paper whose thrust is that quantities cannot be successfully abstracted from their representational system; people seem to use a notation as an expression of mental activity rather than as a mere means of delivering abstractible relations to mind. Gonzalez and Kolers explain that the implication of these

observations is that mental operations do not exist in some abstracted space and are not carried out in some abstracted metalinguistic medium, but depend for their successful use upon the specific learning that acquired them and that expresses them. They contend that there are likely to be strong implications from such results for neuropsychological investigations of mental operations.

Summary
The debate about the utility of a common representation continues, with much evidence for semantic, asemantic, visual and no common representations exists. However, it appears that while such a representation may exist, it may not be used in all number processing.

5. Conclusions

The number processing system seems to be divided into several sub-sections, with each sub-section handling a different type of processing. Thus any model of number processing would have to include various sub-sections. The model may also have to explain either how a common representation could be implemented to aid information transfer between the sub-sections or explain how the functioning of the model produces results which indicate the existence of a common representation.

D. Implementations

1. McCloskey, Caramazza and Basili (1985) Model

McCloskey, et al., (1985) listed four reasons to motivate the modelling of number processing:

1. Looking at performance to sketch a general architecture of the number-processing system.
2. General model provides a framework to interpret impairments. As the model is elaborated increasingly specific interpretations of deficits is possible.

3. Well-articulated theory of normal and impaired cognitive processing is needed to relate cognitive processing to brain mechanisms. Need sophisticated analysis of brain mechanisms and cognitive processes.
4. Explicit model of normal and impaired processing needed for diagnostic tests and cognitive rehabilitation. [McCloskey, et al., 1985].

In considering the cognitive mechanisms implicated in the use of numbers, they began by distinguishing between the *number-processing system* and the *calculation system*, see figure 3. The number-processing system comprises the mechanisms for comprehending and producing numbers, whereas the calculation system consist of the facts and procedures required specifically for carrying out calculations.

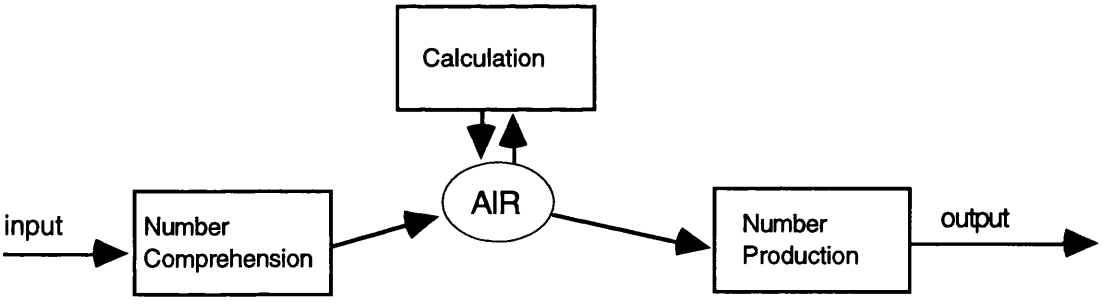


Figure 3. The Mechanisms Implicated in the Processing of Numbers and Their Interaction According to McCloskey, et al., (1985). The AIR refers to an Abstract Internal Representation for each number, which is believed to correspond to the semantic information about that number.

McCloskey et al. postulate that the division of mechanisms outlined in figure 3, communicate via an Abstract Internal Representation [AIR] for each number. These AIRs are postulated to be semantic representations and are assumed to specify the number in abstract form. The assumption appears to be that the system "understands" the meaning of the digits 0 to 10 and the processes of raising 10 to

an exponential value. Thus, specifying the composition of a number in these terms is "defining" it for the system. For example, the arabic numeral comprehension process is assumed to generate from the stimulus 5030 the semantic representation {5}10EXP3, {3}10EXP1.

Each of the sections described in figure 1, operate independently. The sections are further divided into subsections, with each responsible for a different kind of processing. For example, the Numeral Comprehension and Production sections are divided into components for processing *arabic numerals* (i.e. numerals in digit form, such as 362), and components for processing *verbal numerals* (i.e. numerals in word form, such as *three hundred sixty two*). This mechanism is expressed diagramatically in figure 4.

Within the arabic and verbal numeral comprehension and production components, a further distinction is drawn between *lexical* and *syntactic* processing mechanisms. Lexical processing involves comprehension or production of the individual elements in a numeral (e.g. the digit *3* or the word *three*), whereas syntactic processing involves processing of relations among elements (e.g. word order) in order to comprehend or produce a numeral as a whole. (Note: in referring to verbal numeral comprehension and production components, they did not intend to claim that the mechanisms for processing numbers in word form are necessarily separate form the mechanisms for processing language in general; this remains an open question).

Finally, within the lexical processing components for verbal numeral comprehension and production, the model distinguishes between *phonological* processing mechanisms for processing spoken number words and *graphemic* processing mechanisms for processing written number words.

The McCloskey, et al. (1985) model postulates what might be considered a minimal repertoire of cognitive mechanisms for accomplishing arabic and verbal numeral processing and basic arithmetic. Further, the flow of information is tightly constrained

- the various processing components are assumed to communicate via a single form of internal numeral representation.

The McCloskey, et al. (1985) model assumes that within the phonological output lexicon the representations of the basic number words (i.e., one, two, .., eighty, ninety) are partitioned into three functionally distinct classes. The ONES class contains phonological representations of the words *one* through *nine*, the TEENS class contains phonological representations of the words *ten* through *nineteen*, and the TENS class contains phonological representations of the words *twenty, thirty*, and so forth, up to *ninety*.

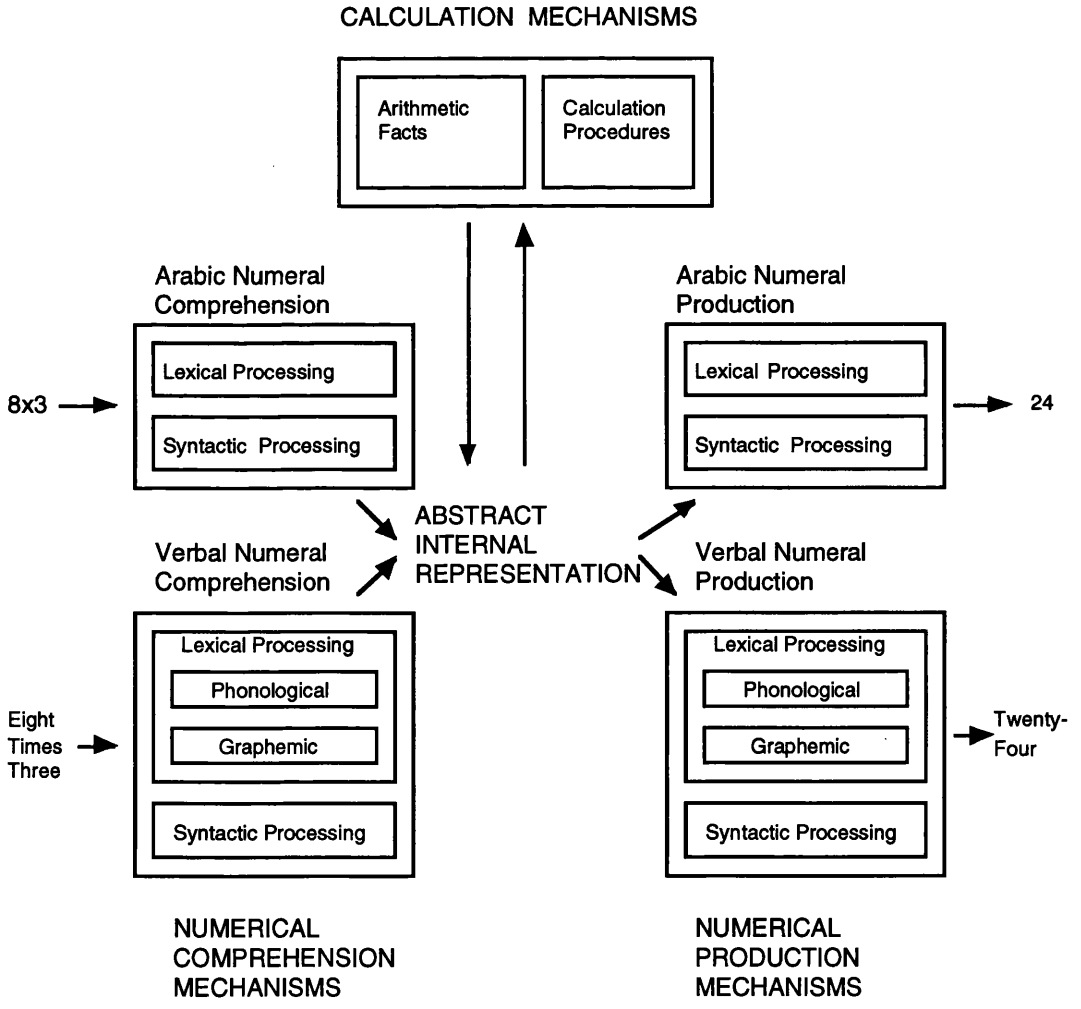


Figure 4. Subdivisions Within the Model of Number Use by McCloskey, Caramazza and Basili, 1985. Note that all processes operate off an abstract internal representation. Figure taken from McCloskey (1992).

2. Dehaene (1992) Model

The Dehaene (1992) model is built on the principle that "Number processing in its fundamental form, seems intuitively linked to the ability to mentally manipulate sequences of words or symbols according to fixed transcoding or calculation rules" (Dehaene, 1992). He notes that abilities with numbers can be seen to be highly related to abilities with letters. It therefore seems unnecessary to postulate a separate numerical mechanism. He also refers to priming effects between arithmetic facts, which can be found to last for one minute or more, indicating that residual activation is carried from one problem to the next. Also, subjects are slow to verify the incorrectness of problems like 4+3 = 12 (where the result is due to the wrong operation) indicating that both multiplication and addition are initiated irrepressibly from the presentation of an arithmetical problem. He notes that when comparing numbers there are no discontinuities in the RT by separation line at decade boundaries i.e. it takes roughly the same amount of time to compare 58 and 59 as it is to compare 59 to 60. Conflicting results are termed the categorisation effect and found by Itsukushima, et al. (1990), reported in Chapter One.

On investigating number processing Dehaene found a strange interaction between relative number magnitude and response key in a parity judgement task. Regardless of their parity, the larger of the set numbers yielded faster responses with the right hand than with the left, and the reverse was true for the smaller numbers. He termed this the SNARC effect (spatial-numerical association of response codes). This effect was not mitigated by the handedness of the subject and was not effected by crossing the subjects hands, but was reversed in Iranian subjects who write from right to left. He found that the comparison times were better predicted when the distance between the two compared numbers was measured on a logarithmic rather than a linear number line. Since the results of experiments which compared visual numerosities and arabic digits produced essentially the same results for both types of stimuli, Dehaene believes that "the possibility must be entertained that a single representation of approximate numerical quantities, obeying

Weber-Fechner's law can be accessed either via numerosity estimation, or via transcoding from arabic notation" (Dehaene, 1992).

Dehaene concludes from these findings, as did the researchers in section C.4.2 (common representation) that numbers are transformed into a mental representation using a subjective scale of numerical magnitude (a semantic common representation). He goes on, however, to say that they are thereafter treated just like other physical quantities. He takes the evidence for two separate numerical pathways (symbols and approximate quantities) from the neuropsychological findings of dissociations between these functions in brain-lesioned patients, such as patient NAU who could estimate the magnitudes of solutions to simple arithmetic problems, but could not solve the problems. He could retrieve approximate quantities, but could not manipulate the symbols well enough to solve the problems.

Dehaene calls his model the Triple-Code Model (see figure 5) and it is based on two premises. First, numbers may be represented mentally in three different codes. An auditory verbal code (or auditory verbal word frame), created and manipulated using general purpose language modules. Also a visual arabic code (or visual arabic word frame), numbers are manipulated in arabic format on a spatially extended medium. Lastly, analog magnitude code, in which numerical quantities are represented as inherently variable distributions of activation over an oriented analog number line.

Dehaene points out two clarifications of premise one. The first is the translation paths to and from the magnitude representation cell work only approximately i.e. there is no syntactic sophistication. Thus, they do not produce well-formed verbal/arabic number forms. They are used to produce "round" numbers (i.e. one-hundred) in response to a "how many?" query. For a more sophisticated estimate (i.e. one-hundred and seventeen) verbal counting is required. The second clarification is that the translation to and from magnitude representations may be through only one privileged path, either verbal or arabic and not both.

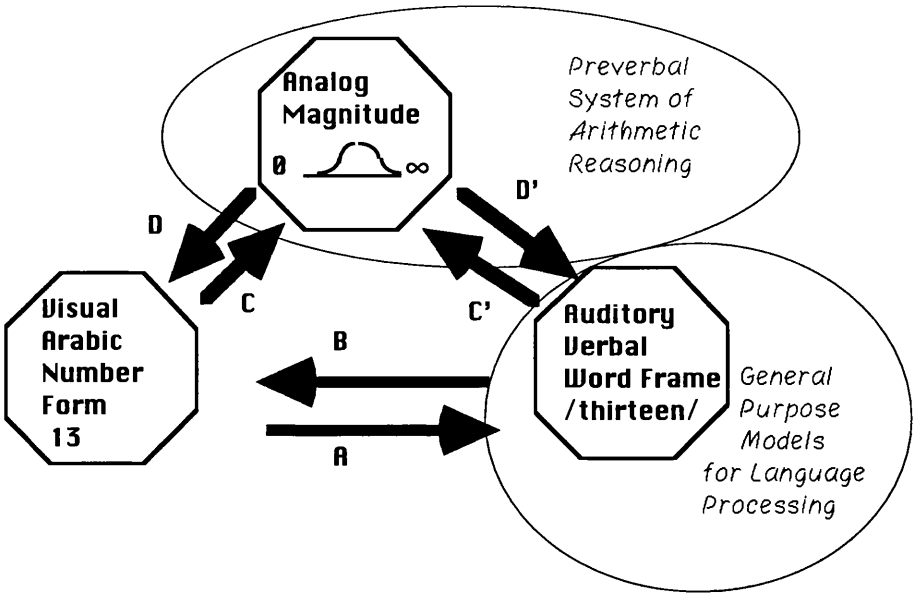


Figure 5. Dehaene's (1992) Number Processing Model. Each of the codes is associated with different types of processing. Arrows are the paths between them. Opposite pointing arrows are converse operations. A & B represent the translations between arabic and verbal codes. C & D are conversions between arabic number forms and quantity estimates. C' & D' are the same for verbal forms and quantity estimates. Auditory verbal information is seen as one of the general purpose models for language processing, while the analog magnitude information is considered the basis of a preverbal system of arithmetic reasoning.

The second premise on which Dehaene has built this model is that each numerical procedure is tied to a specific input and output code. This includes complex procedures which should allow decomposition into code specific subprocesses i.e. every process used in the manipulation of numbers should be traceable to a single medium, either the Visual Arabic number form, the Auditory Verbal word frame or the Analog Magnitude information. He does not believe that procedures are tied to different codes before learning. Thus, the actual code used by an individual is dependent on idiosyncratic learning experience. His model encompasses a common representation (analog magnitude information), but does tie all processing to this representation and so avoids the problems outlined in section C.4.2 (no common representation).

Dehaene defines some code-function assignments. Numerical comparisons are found to translate arabic input into analog magnitude codes where the comparison takes place, while multi-digit operations seem to involve the manipulation of a spatial image of the arabic notation. He notes that there is evidence that addition and multiplication tables are stored as verbal associations, thus explaining why bilingual subjects report accessing these facts in their native language. Parity decisions are postulated to depend exclusively on arabic code. Experimental evidence shows that on these tasks stimuli from a variety of verbal forms appears to be translated into a common form before parity information could be accessed. He believes this common form to be the arabic form.

He sees the analog magnitude representation as the main (if not only) semantic representation of number in adults. That is, the only way it is known "how much" a number corresponds to is to plot that number in relation to all other known numbers. This model views numerical cognition as a layered modular architecture, with the preverbal representation of approximate numerical magnitudes supporting the progressive emergence of language-dependent abilities such as verbal counting, number transcoding and symbolic calculation. The contention appears to be that an underlying intuitive 'feel' for the size of a number allows the development of the other number processing abilities.

3. Ashcraft Model

Three theoretical assumptions about mental arithmetic led Ashcraft's (1982, 1987) model design. First, the simple arithmetic facts (i.e., 3 + 4 = 7) are stored in a memory network. Second, this network is dimensioned in some fashion analogous to the semantic distance or relatedness effect. Third, the process of spreading activation is the basic mechanism of memory retrieval from the network.

He notes in Ashcraft (1992) that the two most important structural aspects of the network involve the concepts of strength

and relatedness among nodes. In the network, each problem-to-answer association is represented in terms of strength or degree of accessibility. They are also coded on the degree of relatedness among problems and answers, in that adjacent ("near neighbour") nodes are more strongly interlinked than more distant, non-adjacent nodes. In this way, his model operates like the SON model. The closer to nodes (problems and solutions in Ashcraft's model) the closer their activation values and the more easily they are confused. Consider the problem 8 x 3 = 32. If *x* and *y* are, respectively, the answer stated in the problem (32) and the answer received from memory (24). To decide which is the correct answer to the problem the decision stage mechanism has to discriminate between these values, based on their respective levels of activation. Neighbour nodes that received especially high levels of activation during search (for example 32 will receive more activation than 100 since it is closer to 24) should disrupt decision processing to a greater degree due to the contention that the discrimination is more difficult as the levels of activation of *x* and *y* converge. This mechanism is thought to account for the split and confusion effects found in arithmetic (e.g., Ashcraft and Stanzyk, 1981; Stanzyk et al., 1982; Winkelman and Schmidt, 1974; Zbrodoff and Logan, 1986).

Another aspect of this model is especially relevant to the modelling of children's data. The entire retrieval and decision process is said to occur in parallel with a procedural-based solution attempt, in which counting or other reconstructive processes began simultaneously with retrieval.

He postulates two changes or elaborations to the model. The first, that each operand has stored pathways or associations to a variety of answers, correct as well as incorrect. This is from a Siegler and Shrager (1984) model. In this scheme, associations between problems and answers (and, later, multipliers, etc.) are formed each time a child encounters an arithmetic problem, regardless of the setting, and regardless of the correctness or incorrectness of the answer. These associations will vary in strength as a function of experience, will tend to reflect the error, confusion, and

possibly the split characteristics described above. This provides an explanation of retrieval errors.

The second change or elaboration is that the procedural knowledge component is elaborated to capture the variety of strategies observed among children, as well as the more idiosyncratic strategies that are sometimes invented (e.g., the variety of specific methods that fall under the "decomposition" or "solve from known facts" strategies; see Hamann and Ashcraft, 1985; Siegler, 1987). Notice that, as in the original model, some associations in the network will be of very low strength, such that functionally the relevant problem may always be solved by a rule rather than by retrieval (e.g., N x 0).

With these two changes the Ashcraft network model is believed to account for the arithmetic problem findings. However, it does not address the other number processing mechanisms that are of more interest in the framework of this thesis.

4. Encoding Complex Model

A very different view of numerical processing has been put forth by Campbell and Clark (1988; Clark and Campbell, 1991). The Campbell and Clark *encoding complex* view posits a non-modular architecture in which multiple numerical codes activate one another in the course of numerical processing and arithmetic tasks. The codes, which may include phonological, graphemic, visual, semantic, lexical, articulatory, imaginal, and analog representations are assumed to be interconnected in an associative network, so that individual codes may activate one another to produce a multi-component "encoding complex". Campbell and Clark, according to McCloskey, apparently assume that any code may potentially be recruited at any phase of a numeral processing or calculation task, and that multiple codes may be implicated at any point of the processing. Furthermore, they posit individual differences in the complex of codes involved in particular tasks.

McCloskey notes that although intriguing in many respects, Campbell and Clark's encoding complex view has not yet been developed into a specific model capable of generating clear predictions.

5. Conclusions

The aim of this thesis is to investigate the ordering of stimuli. To this end, the present chapter has described four models of number processing developed from the neuropsychological literature. It has been postulated that the idea of "order" is based on the number series. Because of this aim, the models will primarily be discussed in their explanation of order. Therefore, key aspects of the models may go undiscussed if they fall outwith this brief.

The McCloskey, et al. (1985) model contains some aspect of order in the abstract internal representation. That is, numbers are translated into semantic representations which encode the digit value and power of ten. This would allow the ordering of ONES, TEENS, and TENS, if the concepts were attached to each other with greater-than/less-than connections. However, it does not address the ordering of the digits. At no time does the model explain how one could solve the problem "is 3 less than 4". As has been shown in Chapter One, postulating a greater-than/less-than connection between digits would be inadequate.

This problem occurs in the Dehaene (1992) model in a different way. He appears to address the issue of order, but seems to say that order is achieved by mapping objects onto a pre-ordered structure (a "number-line") in memory. This begs the question of how that "number-line" developed or how it might be realised in neurological terms. He solves the first of these problems by contending that the "number-line" is innate.

The last two model outlines are added for comparison, but they are not of particular interest for this thesis. In the case of the Ashcraft model (described in Ashcraft, 1992), it address the processing in the section of the number processing system that is

not of interest. In the encoding complex model of Campbell and Clark (1988) however, the definitions are too vague.

E. Implications

The main result of this section is that although number processing mechanisms can be differentiated from letter processing (letters are not generally used in calculations etc.), indicating that they are processed by separate networks, this does not preclude both types of information from being stored in the same *type* of network. The common type of network view is supported by the finding that the mechanisms for the meaning and general use of numbers appear to be located in the left hemisphere, which is also commonly the location of language processing. The general premise of this thesis, that the*sequencing* of numbers and letters is carried out in the same or similar networks, may be tested by determining if degeneration of the two sequences is inextricably linked or, at least, related.

The second result of note is that number processing can be distinguished from calculation. This distinction leads to further distinctions within number processing and calculation. The number processing mechanisms are further divided into arabic/digital processing, verbal/oral processing and writing/reading, with these processes sharing a common abstract representation or operating independently with transcoding algorithms between them. As McCloskey notes, the processing mechanisms for verbal/oral processing and reading/writing may be shared with those used for language. With the results from comparison studies indicating that letters, animal sizes and other stimuli produce the same type of results as numbers, it is equally parsimonious to hypothesise, as Dehaene does, that numbers are treated like other physical quantities - or indeed all sequenced material. The separation out of calculation can be viewed as analogous to higher language functions (i.e., semantic fact retrieval and script or procedural frames), which may also support the view that numbers are not subject to totally unique processing.

The first two models demonstrate the other main points of this chapter i.e. the utility of an abstract internal representation of number and the distribution of processing. The McCloskey model advocates the abstract representation in the form of a semantic translation which attaches digit values to powers of ten (i.e., 20 = {2}10EXP1). This representation is ideal for dividing the sequence into classes such as ONES, TEENS, and TENS, but it does not seem immediately obvious how other physical quantities could be represented using this method, nor how quantity approximations would be made.

This model also runs into difficulty explaining the results of Gonzalez and Kolers (1982 - reported again in 1987) who found that translation between arabic and roman numerals and between languages affected number processing in ways that seem to counter-indicate a common representation.

The Dehaene (1992) model does not have the same difficulty. Although a common representation is postulated, different types of processing are still tied to the representations in which they were learned. This model also provides a mechanism for approximation and the processing of other physical quantities. In fact, any sequence elements could utilise the "number-line".

The problem that this model faces, as far as the purpose of this thesis, is the lack of definition of the "number-line". It appears to be postponing the problem to say that "things are ordered by mapping them onto an ordered list in the mind". The model produced in the next chapter is designed to address the issue of how "an ordered list in the mind" might be constructed and, indeed, what the concept of "ordered" could mean in neurological terms.

In this chapter the number of phenomena which must be accounted for in a model which proposes to explain how the number sequence is stored has been dramatically decreased. The finding that calculation abilities are not inextricably related to the underlying representation of numbers has left open the possibility that this representation is not unique to numbers. It has also been shown

that the most promising model contains an underlying representation of order with various processing mechanisms operating from (or in parallel to) it. A possible implementation of this underlying representation is discussed in the next chapter.

Chapter Three - A New Model of the SDE and Related Effects

This chapter begins the development of a connectionist model of the effects outlined in Chapter One. The evidence from Chapter Two, that number processing occurs separately from arithmetic processing, indicates that it is unnecessary for the model designed here to encompass both. Arithmetic processing will, therefore, not be addressed in the remainder of this thesis.

The initial model development centres on the relative order judgement section of the network. A novel architecture is created and its ability to generate the phenomena of interest is investigated.

Due to their inherent linear nature it seems unlikely that a symbol processing model would be able to generate the SDE. Thus the development of the model utilises the flexibility of the PDP environment.

The underlying idea for this section of the network echoes the theory of 'remote associations' discussed as early as 1935, by Hull. In this theory "..remote forward associations [are used] to account for a symmetrical serial-position curve..(Deese, 1958)" found when a list of items is recalled from short-term memory. That is, when subjects are taught a list of words and asked to recall that list serially, the elements at the beginning of the list are recalled well, while those in the middle and end tend to be lost. Bugelski (1950) reasoned that the ability to recall an element was inversely proportional to the number of associations spanning the portion of the list in which that element occurred. With a fully interconnected list the number of connections spanning the position of an element is higher for elements at the middle of the list and lower for those at the ends. Spanning associations are thought to include those which do not connect to an item itself but pass over it. This higher level of spanning associations was thought to cause an increase in the interference with the emission of a correct response.

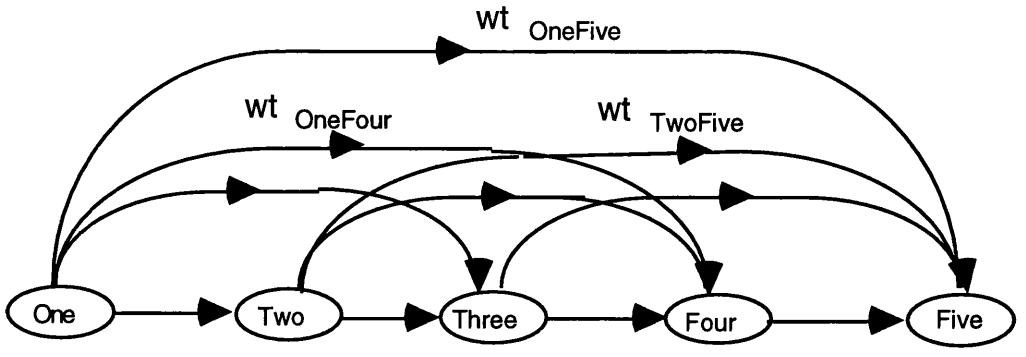


Figure 6. Fragment of the SON model. First five nodes of the SON with the connections between them, Three connections are labelled to indicated the weight of the link (Wt_{ij}).

A. Description of the Network

The network developed here has been named SON, which stands for Serial Order Network. The SON was developed using the MacBrain (3.0) simulation package. It consists of ten nodes, each corresponding to a digit. A fragment is shown in figure 6.

Activation values were computed in all nodes by the same activation function. The activation function used was the continuous sigmoid functions:

$$A_i(t+1) = \left(1/1 + e^{(-w_i(t)/T^*)}\right) - R_i$$

where:

A_j = Activation at node i on the next time step.
$w_i(t)$ = Weighted input to node i at time t.
T = Temperature Value.
R_i = Threshold of node i.

* - since the temperature parameter is the denominator here, it cannot be set to zero without causing the equation to behave strangely.

Each node passes activation to all nodes representing digits after it in the sequence, with these being the only connections in the model, after the fashion of 'remote associations'. These are one

way connections. This configuration causes the activation of a node to increase with its distance from any upstream active node, because all intervening nodes become active and in turn pass on activation. The weights between the nodes dictate how much activation is passed from one to the other, these are labelled wt in figure 6. Thus:

$$w_i = \sum_{j = \text{first\_node\_to\_i}}^{\text{last\_node\_to\_i}} \left(A_j * wt_{ji} \right)$$

where:

w_i = Weighted input to node i at time t.

A_j = Activation at node j.

wt_{ji} = Weight from node j to node i.

If a node receives no input its activation decays in proportion to a parameter specifying the decay rate, which is between 0 and 1. The higher the decay rate the faster the activation value falls. Eventually, if no input is forthcoming, the node's activation will decay to 0.0. The activation values of the nodes are constrained by the Sigmoid activation function to between 0 and 1.

Implementation Options

As a digit is encountered the node associated with it is activated. This is then called the initial node, representing 'the initial node to be activated'. Activating a node consists of either setting its activation value to one, or building its activation to one over time. The node can then either be continuously reactivated or left to decay. If the activation value is continuously reactivated (set to one) the node at separation one is able to build as much activation as possible before the activation at the initial node decays, but the activations of nodes at larger separations increase too quickly, reducing or cancelling the SDE. Thus a compromise in the number of epochs over which the activation should be set to one must be reached. The building of activation in the SON can also be controlled by altering the threshold and decay values.

The decision mechanism is not built into this network, but the ability to make relative order decisions is inherent in the network structure. A decision as to the order of two digits may be made using one of two measures with one of two starting points. There are two measures that can be used to determine if the second digit (hereafter referred to as the comparator) is recognised as larger. The first measure is to determine the first node to reach a pre-set threshold of response within a constrained period of time. The second is to compare the activation values of the nodes corresponding to the two digits, with the digit corresponding to the node with the higher response rate considered larger. To start activation in the network the activation of either the initial node or the node representing the beginning of the sequence (hereafter referred to as the start node) is set to 1.0.

Besides order judgements, it may be possible to determine the distance from one digit to another by the difference in the amount of activation at the corresponding SON nodes, the larger the difference in activation the greater the distance. Distance may also be determined by the difference in the amount of time taken for the nodes to reach criterion activation, with larger time difference equalling greater distance.

The following simulations are designed to investigate the various different implementation options outlined in this description, while also investigating the utility of the model. Experiment One explores how well the simplest form of the SON captures the general SDE and how activation information might used. The second experiment investigates the intuitively appealing idea that connections to adjacent digits should be stronger than connections to distant digits. The compatibility of these connection weights with SDE is then examined. The final experiment investigates the adaptations necessary for the SON to reproduce RT data found in previous research.

B. Initial Obsevarions

The model was first constructed in its simplest form with only 4 nodes. It was immediately noticed that activation values did grow faster for nodes further from the endpoint. Initial tests indicated that these values allowed activation to build slowly in distant nodes, without disappearing from proximal nodes too quickly.

The temperature value was found to have a large impact on the generation of activation in the network. Since it appears as the denominator in a portion of the activation equation, the temperature parameter must be retained, even though it is not necessary as this is not a gradient descent network. It remained at 0.1275, which is the default setting in the MacBrain system, for all experiments, as lowering it slowed the build of activation in the nodes, without any theoretical motivation.

Initial examination found that the nodes at distance one from either the initial node or the start node (the distance one nodes are hereafter called the adjacent nodes) never reached an activation value of 1.0. Its highest activation value was, therefore, taken as the maximum activation for the network.

C. Study One - General SDE and Measures of Order

For this experiment the SON was increased in size to 10 nodes. It was designed to test whether the SON could demonstrate SDE. A subsequent investigation was then carried out on the interpretation of the activation values of the nodes in the SON and how this interpretation is related to the digits they represent.

1. Network and Parameters

The threshold and decay values associated with the nodes were uniform throughout the network, the thresholds were 0.6 and the decay values were 0.1. All weights were set at 0.35.

2. Design

To begin a simulation, the activation of the initial node (or start node) was set to 1.0 for three epochs. Three epochs allowed the activation of the adjacent node to build substantially before the activation in the initial node stopped. Removing this input to the adjacent node caused it too to begin decay. If fewer than three epochs were used, activation in the adjacent node did not grow, before it began to decay, to remain active long enough for a decision to be made. However, if more than three epochs were used the activation values of nodes at higher separations did not remain different for enough epochs for the response mechanism to tell them apart.

For this experiment, node Three was the initial node. This allowed the largest number of distant nodes while maintaining a noticeable difference to the start node condition. The start node condition and the case in which the initial node is One are identical.

3. Results and Discussion

The purpose of this investigation is two-fold. First, to examine the information in the network for its potential to account for the basic relative order judgement effect of SDE. The network is designed for this purpose, if it fails to generate the potential to model this effect, it is useless. The second aspect of this investigation is to determine how the information in the network can be used to perform relative order judgements.

3.1 Ability to Model SDE

Since node Three was chosen as the initial node, all curves should be viewed as starting at node 4. Activation only travelled to the right of the initial node making node 4 the first node to receive activation (the adjacent node). As can be seen in figure 7 the activation values of the nodes furthest away from the initial node were the highest for all epochs after Epoch 2.

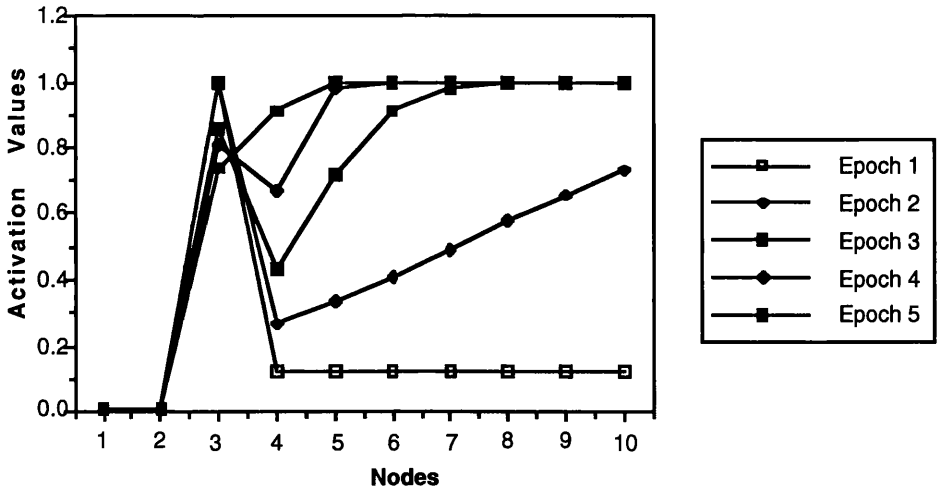


Figure 7. Activation Values from First Five Epochs, Initial Node Three. The activation in node 3 was set to one for the first three epochs. Thus, the curves representing the results of this manipulation should be considered to begin at node 4.

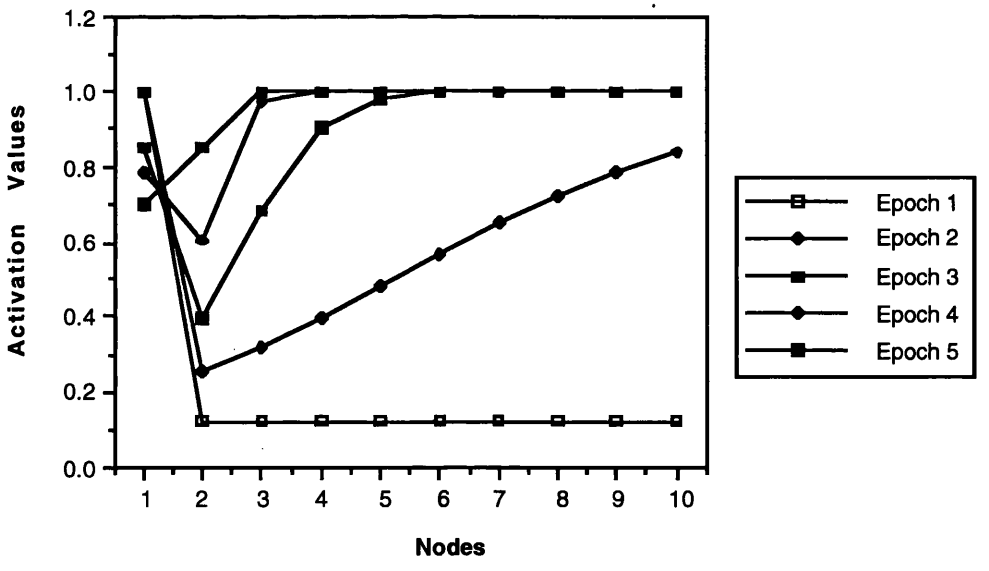


Figure 8. Activation Values from the First Five Epochs, with Initial Node One. In this case curves begin at node 2.

Although only the result for the node 3 as initial node condition are given, all initial nodes showed the same pattern of results. With all weights equal, activation values increased smoothly over time maintaining the relationship between them. This can be seen by examining the curves in figures 7 and 8. The increasing nature of the curves remains until they level off as the later nodes reach a response rate of 1. This demonstrates that the network proposed is capable of generating activation values that should allow modelling of the SDE.

3.2 Measures of Order

The second objective of the experiment was to investigate the information available in the activation values of the nodes in the SON. Two measures of order are investigated in this study; the length of time taken for a node to reach an activation threshold, which can be measured by either the number of epochs or the average change in activation between epochs, and the difference in activation between two nodes of interest. Both measures can be used when the trial is started by activation of the initial node, regardless of where it occurred in the sequence, or when activation of the start node begins the trial.

Activation Threshold Measures

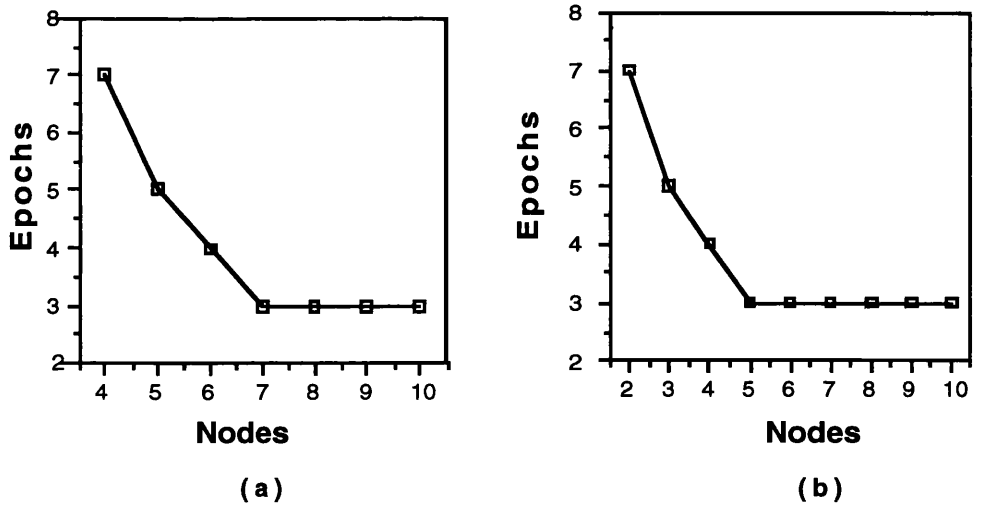
Epochs to Threshold

The first measure of order investigated is based on the relationship between activation values and the number of epochs [cycles of network activation] required to reach them. In both initial and start node conditions the threshold of activation was taken to be the largest activation value attained by the adjacent node, because this value never reached 1. This value was 0.98 in the initial node (Three) condition and 0.97 in the start node condition.

The number of epochs for each node to reach the response threshold after activation of an initial node is displayed in figure 9a. The number of epochs to reach threshold after activation of the start node is displayed in figure 9b. In the initial node condition,

problems arise on nodes 6, 7, 8, and 9, which require the same number of epochs to reach threshold. Using the activation threshold criterion the network would see these digits as having the same magnitude. The same problem occurs in the start node condition for nodes 4, 5, 6, 7, 8, and 9. This may not be considered a problem if one recalls the Compression effect, which states that comparisons between large numbers were slower than those between small numbers at the same separation. In the start node condition, the number of nodes compressed and their proximity to the start node indicates that this system of order judgements is unlikely to be useful. Section E investigates this problem further.

For now, it is important to note that all nodes reached activation values of 1.00 except those at separations 1 and 2 which reached values of .982 and .998, respectively, when the initial node was node 3 and .973 and .999, respectively, when the initial node was node 1 (this is also the start node condition). For the number of epochs to threshold as the measure of order, it appears that activating the start node leads to more difficulty than activating an initial node.



Figures 9a and 9b. Epochs to Maximum Activation Value for Networks with Initial Nodes 3 (a) and 1 (b). Method of determining order #1 - Activation Threshold. (a) shows a network with node 3 as the initial node, while (b) is a network with the start node active.

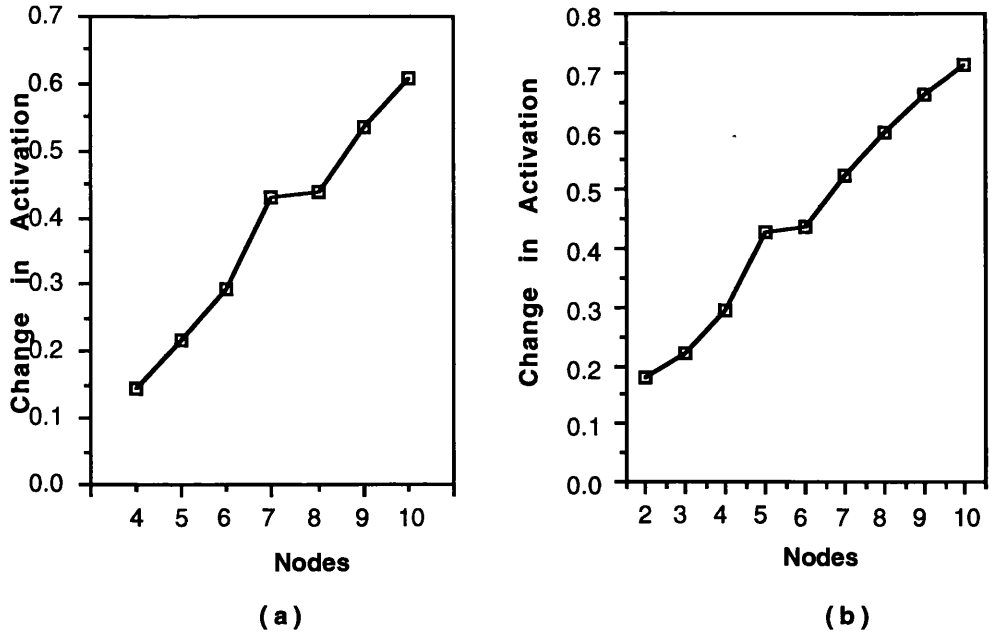


Figure 10a and 10b. Average Activation Differences for Nodes in Networks with Initial Nodes 3 (a) and 1 (b). Method of determining order #2 - The Average Activation Difference. That is, the average amount the activation changes for each node between epochs. The average is taken on epochs before the node's activation reached 1.0. (a) shows a network with node 3 as the initial node, while (b) is a network with the start node active.

Average Activation Difference Measure

The second method of estimating the time taken for activation values to build to threshold, is based on the average difference in the nodes' activation values before they reach 1.0. This is a measure of the acceleration of the activation values, the motivation being that the activation build to threshold over time does not occur in discrete time steps, but is continuous. Therefore, the faster the activation increase, the quicker the order decision should be made. The average change in activation as a measure of the time required to build to threshold is closer to the ideas of cascade expressed in McClelland (1979), than the number of epochs. Due to the sigmoidal activation function, the closer the nodes' activation to the maximum activation value (in this case 1) the slower the change in activation. Therefore, the average increases were taken from only those trials before the activation

value reached 1.0. These activation differences are illustrated in figures 10a and 10b, for the initial node and start node conditions, respectively.

Since the differences between the nodes do not decrease for nodes further along the sequence, both initial node and start node conditions will produce similar results. Comparing values to the start node will not differ from comparing values to any initial node. The average difference increases steadily indicating that each digit was equally likely to respond quicker than its adjacent digit, which were all equally likely to respond quicker than digits at separation 2, 3 etc. This may create difficulty when attempting to model the Compression effect.

One might also, at first glance, see a difficulty for both conditions in comparisons between the initial or start node with any other node, due to the high activation of these nodes. However, the activation values of the start and initial nodes have negative acceleration values and should therefore be easily judged smaller than all other nodes.

For the use of changes in activation values as a measure of the order of two nodes, both initial node and start node conditions seem equally appropriate.

Actual Activation Differences

The remaining measure of order is the difference in the activation values of the nodes of interest. This seems an intuitively appealing measure of order. However, when either the start node or the initial node are activated this measure appears to be ineffective because the activation values in the distant nodes grow too quickly. All nodes at separation values greater than 5 had activation values larger than 0.997 by the third epoch. Nodes at separations of 3 and 4 had activations exceeding 0.999 by the fourth epoch. Therefore, the differences in activation between these nodes and any single previous node would be the same, be this the initial node or any node of interest in the start node condition. In the start node condition the differences between the

activation values of adjacent nodes distant from the start node would be minuscule, making it seemingly impossible for a decision network to recognise them. This can be seen in the curves for epoch 5 in figures 7 and 8. It is for this reason that this measure will not be used throughout the remainder of the study as a measure of the order of a pair of digits.

3.3 Conclusions

The network defined has shown definite potential as an analog representation underlying relative order decisions. It is apparent that the ability to model SDE is available from the activation values of the SON nodes. The only measure of order which seems viable is the build to threshold activation. However, at this point a decision cannot be made between the two ways of measuring this, either the number of epochs to reach the network's maximum activation, or the average change in activation per epoch. If the number of epochs is found to be superior, then the activation of an initial node would generate better results than the activation of the start node. On the other hand, if the average build up of activation is shown to be a better predictor of RT, it will be necessary to choose between activating the start node or an initial node. These methods of order determination will be explored further in the remainer of this chapter where the SON will be compared to previous research in this area.

D. Study 2 - Investigation of Direct Links

When investigating serial order it is necessary to remember that any representation must take into account that the items stored (in this case digits) are learned in a particular order. It seems therefore, unlikely that all weights connecting the digits would be of the same strength. As the digits are more likely to appear in order, especially during training, adjacent digits should become more strongly connected to each other than to distant digits. This experiment initially investigates whether stronger direct links are incompatible with the potential to model SDE with the SON. If this is not the case, the effects of stronger direct links on the results

obtained using various initial nodes are investigated. Finally, the effects of large differences between these weights are examined.

1. Network and Parameters

In this set of experiments the decay and threshold values were set to 0.1 and 0.6, respectively. The weights of the direct sequential links were varied, and all remaining links were given equal values less than the value of the direct links. Throughout the experiment all weights except those on direct links were set to 0.25. In the first phase of this experiment the direct weights were set to 0.30. In the second phase the direct weights were increased to 0.35. Finally, direct weights were increased to 0.50.

2. Design

For this experiment, Initial and start nodes activation values were set to 1.0 before the first and second epochs only, in compensation for the increased connections to adjacent nodes allowing more activation to pass and build before the initial (or start) node's activation began to decay.

3. Results and Discussion

The first simulation of this experiment attempts to show that strong direct connections are compatible with SDE. The results of this simulation are in figures 11 and 12. By comparing these to figures 4 and 5, one can see that the underlying trends in the curves have not changed. However, a dip has developed between the nodes at separations1 and 2, with an overall reduction in the curves due to the decrease in the weights from 0.35 to 0.30 on the direct connections and from 0.35 to 0.25 on the remaining connections, changes which are apparent in both figures. The small dip in the curves between nodes at separations 1 and 2, flattens in epoch five and disappears in epoch six. This indicates that although strengthening the direct connections caused some deviation from the expected SDE graph, the deviation is overcome by the increased input to the node at separation 2. This dip implies that if a

decision network, triggered by the output of the SON, was forced to respond before the activation values had built sufficiently, decisions on elements at separation 1 would be made faster (and perhaps more accurately) than decisions involving elements at separation 2. This is the discontinuity in the SDE curve found by Lovelace and Snodgrass (1971). That the discontinuity does not alter with the initial node chosen can be seen in the start node condition graphed in figure 12.

It is important to note that the results in figures 11 and12 are nearly identical, except that fewer separation values are displayed in figure 11. For this reason remaining graphs will contain only the start node condition.

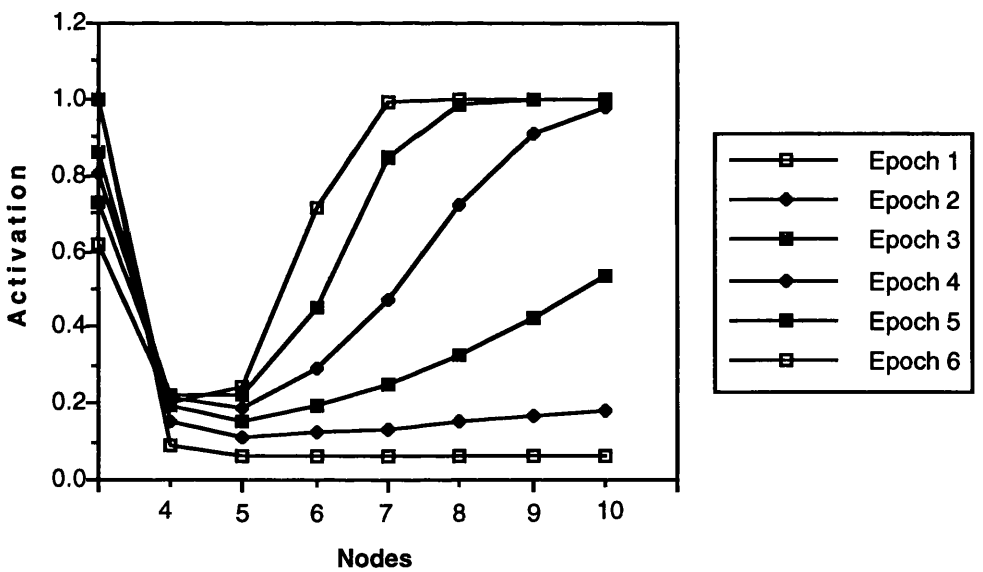


Figure 11. Activation Values Over the First Six Epochs in a Network, with Initial Node 3. Direct weights = 0.30, Remote weights = 0.25. The curve develops a dip between nodes at separations 1 and 2 (nodes 4 and 5) because of the strength of the direct connections.

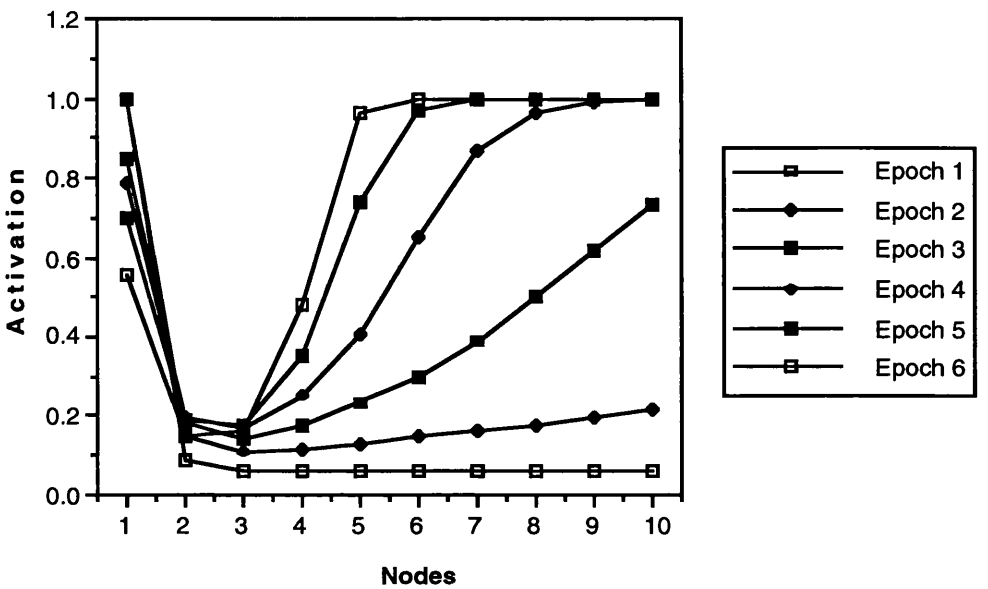


Figure 12. Activation Values Over the First Five Epochs in a Network, with Initial Node 1. Direct weights = 0.30, Remote weights = 0.25, as they change from epoch 1 to epoch 5. The curve develops a dip between nodes at separations 1 and 2 (nodes 2 and 3) because of the strength of the direct connections. Results are almost exactly the same as in figure 11.

Further increasing the weights of the direct connections in the SON model (to 0.35, with all other connections set to 0.25) causes the dip between separation values 1 and 2 to become more pronounced and extend (slightly) over separations 3 and 4 - see figure 13. In this figure the dip between separations 1 and 2 is still apparent at epoch 5, by which time it had all but disappeared in figure 14 (where weights of direct connections are increased to 0.5, all other connections remaining at 0.25). These results predict that if a sensitive enough decision mechanism is used, the higher the proportion of learning dedicated to adjacent items, the greater the discontinuity. This might explain why this discontinuity has been found on alphabet trials not constrained by a deadline, while it is only apparent on number trials if a deadline is imposed. They also imply that, if alphabet trials were constrained by a deadline, one would expect the discontinuity to extend over more than separation 2.

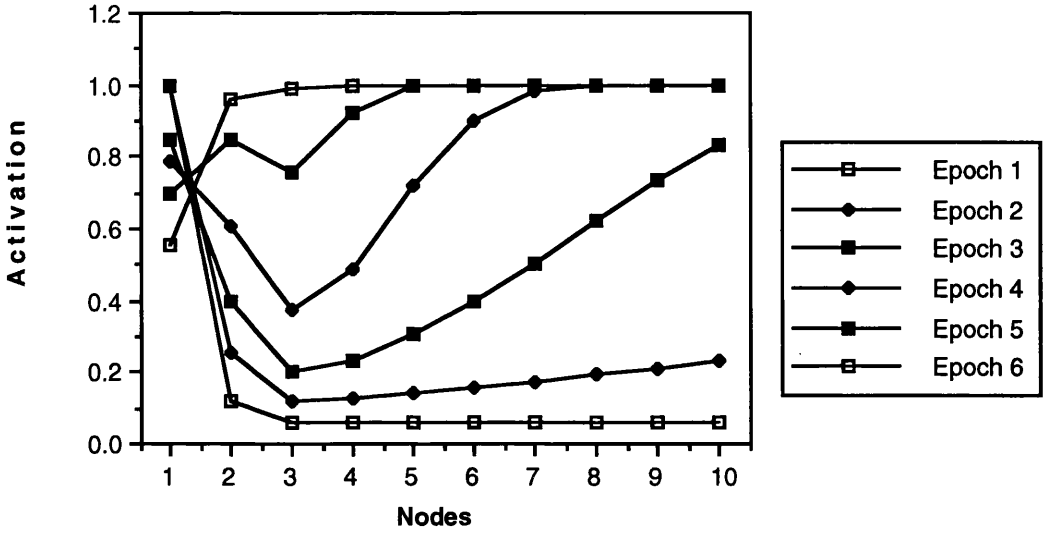


Figure 13. Activation Values Over the First Six Epochs, with Initial Node 1. Direct weights were increased to 0.35, while Remote weights remained at 0.25. The graph shows the activation values as they change from epoch 1 to epoch 6. The dip between nodes 2 and 3 has become more pronounced with the increase in the direct weights.

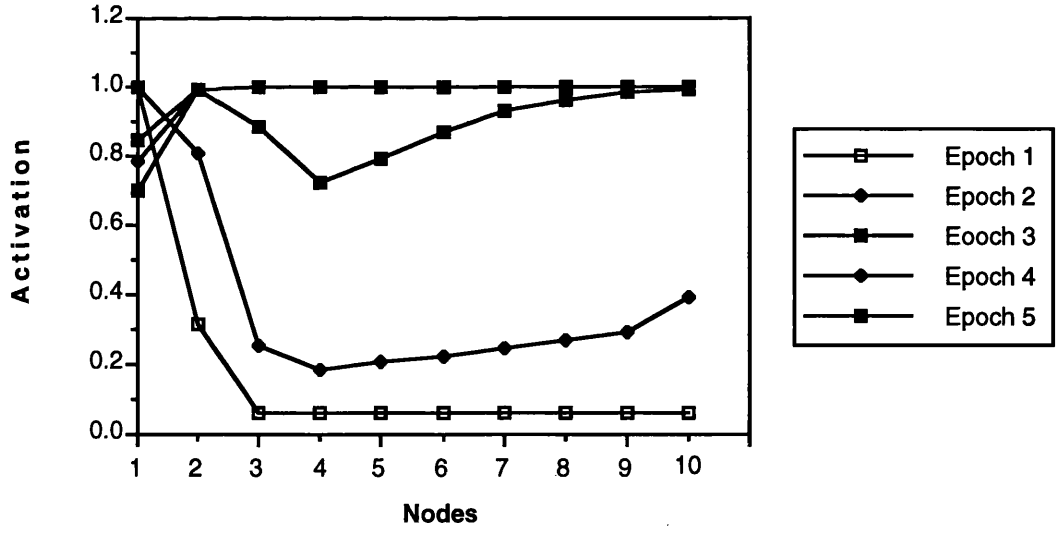


Figure 14. Activation Values Over the First Five Epochs, with Initial Node 1. Direct weights were increased to 0.50, Other weights remain the same at 0.25. The discontinuity is stronger but disappears faster and the expected SDE curve is never really achieved because of the relatively massive direct weights.

Although the dip is greater for the simulation in figure 14, the recovery from it is quicker due to the increase in the direct weights. The expected SDE curve is never actually achieved - by the time the dip has disappeared all activation values are essentially 1.0. The SDE graph might be recovered by a corresponding increase in threshold and/or decay values.

4. Conclusions

The network is further differentiated from other models of sequence representation by its potential to model the discontinuity in the SDE curve. However, the anomaly in the activation curves which would lead to this discontinuity is only displayed in first few epochs. It is possible that these epochs occur too quickly for a decision network to access them, unless the values of the connections between elements are such that the anomaly is sustained across a larger number of epochs. Such weight configurations may occur at different stages of learning and influence the other effects to be modelled.

The value in modelling the discontinuity is defined by the incidence of it in experimental data. The more common a finding is, the more essential it is that a model of the data account for this finding. It is also possible that the occurrence of an effect is influenced by learning. If this is the case a network's ability to model the effect should also be influenced by learning.

The robustness of the discontinuity may be discovered by forced RT experiments, using the alphabet or a novel sequence. If it proves to be a pervasive effect, it will become important to determine how capable the SON is of modelling it. It is also necessary to determine if its occurrence is influenced by learning.

E. Study 3 - Modelling RT Data

So far the SON appears to replicate the SDE in its general sense, but to be a viable model it must be capable of simulating results

as they appear in RT experiments. This section is an attempt to construct representations which appear capable of replicating the results found by Parkman (1971) when subjects were asked to determine if two digits were in the correct order. The subjects responded by pushing a lever to the left of right to indicate if the stimuli were in the correct order. The direction of the lever press response was counterbalanced across subjects. Parkman found results very similar to those of Moyer and Landauer (1967) and Fairbank (1969).

1. Network and Parameters

In this experiment it was necessary to manipulate all of the parameters of the SON. The experiment began with the assumption that the weights at some distance from the first few nodes were as strong as those to proximal nodes. This would stem from the sequence being referred to as the 'numbers One to Ten'. Therefore, in the first simulation the direct weights and the weights to the nodes at distances 8 and 9 (i.e. from One to Nine and Ten and from Two to Ten) were set to 0.35. Remaining weights were set to 0.25. For these simulations, the decay and threshold parameters remained at 0.1 and 0.6, respectively. The initial/start node was set to 1.0 before epochs 1 and 2.

A second simulation was undertaken which was based on the assumption that the connectivity between nodes varied across the sequence, with direct connections the strongest, and remaining weights varying. This weight configuration was an attempt to recreate the stepped pattern of results found in the Parkman experiment (see figure 15). The weight configuration used is outlined in table 8. In this configuration the threshold was increased to 0.67 which slowed the build of activation over the entire network. The decay was decreased to 0.04 which kept the slow building activation from decaying before criterion activation was reached.

Separation	Nodes	Weight
1	1-2, etc.	0.45
2	1-3, 2-4, etc.	0.35
3	1-4, 2-5, etc.	0.35
4	1-5, 2-6, etc.	0.40
5	1-6, 2-7, etc.	0.30/0.33
6	1-7, 2-8, etc.	0.35
7	1-8, 2-9, etc.	0.30
8	1-9 and 2-10	0.35
9	1-10	0.30

Table 8. Weight Configuration Used in Simulation Two. These weights are designed to replicated the stepped pattern of results found by Parkman (1971).

Separation	Nodes	Weight
1	1-2, etc.	0.45
2	1-3, 2-4, etc.	0.35
3	1-4, 2-5, etc.	0.35
4	1-5, 2-6, etc.	0.30
5	1-6, 2-7, etc.	0.30
6	1-7, 2-8, etc.	0.30
7	1-8, 2-9, etc.	0.35
8	1-9 and 2-10	0.35
9	1-10	0.40

Table 9. Weight Configuration Used in Simulation Three. These weights were designed as an improvement over those in table 8 for the replication of the stepped pattern of results found by Parkman (1971).

Finally, the third simulation employed a SON with a weight pattern based on the assumption that near nodes and distant nodes were more highly connected than middle nodes, with near nodes slightly higher than distant. This assumption is based on the serial position curve found in short-term memory experiments. It seems likely that for nodes to be connected to each other they would have to

both be present in any short-term store. Thus, weights are higher between proximal nodes and reduce as the separation increases and then increase again for the largest separations. The weight configuration used is outlined in table 9. In this last network configuration the decay and the threshold parameters remained at 0.04 and 0.67.

2. Design

The aim of this study is to simulate the results found in an RT experiment. Figure 15 shows an approximation of the results Parkman (1971) found in his order decision experiment. It shows RT as a function of the separation between digits. Parkman referred to separation as "split".

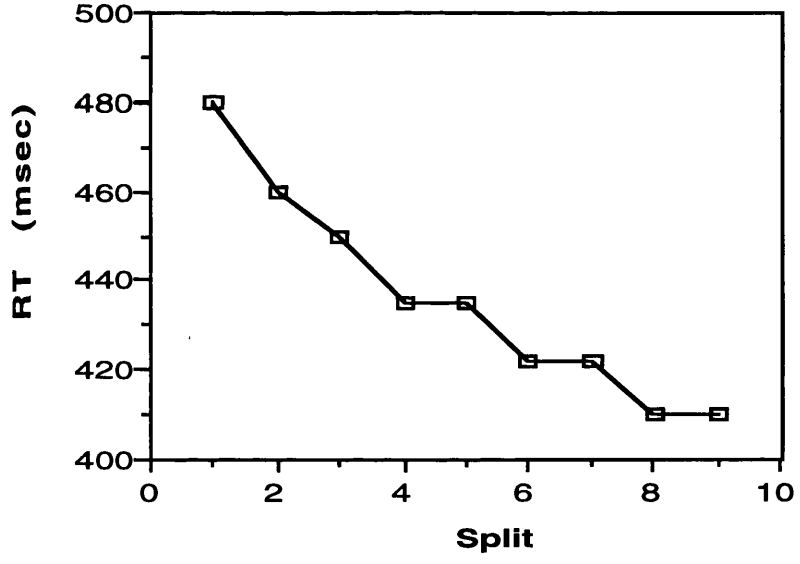
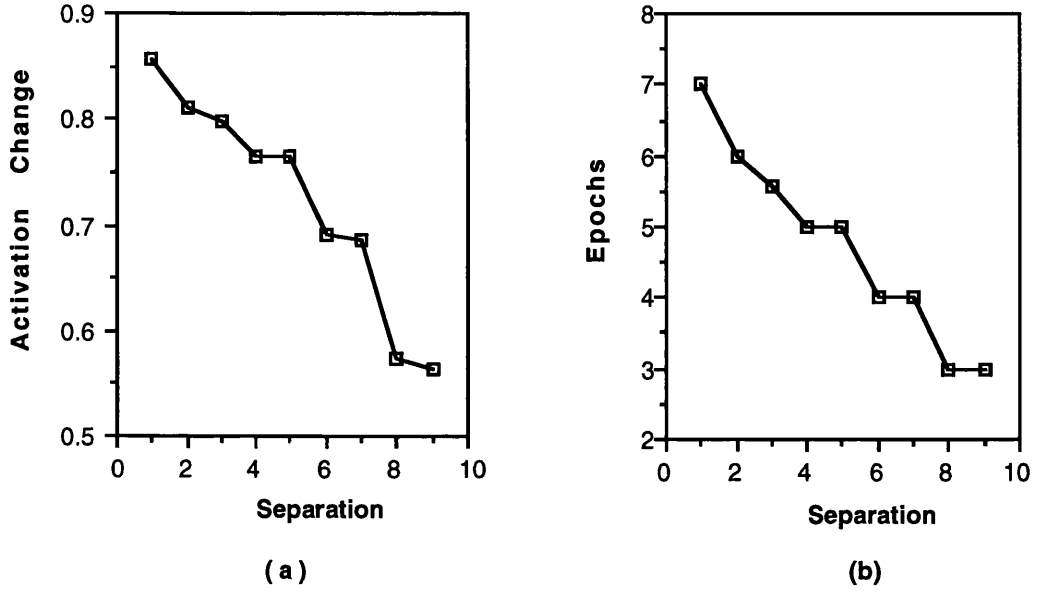


Figure 15. Approximation of Parkman's Data. These results were found when Parkman (1971) asked subjects to judge if two numbers were in the correct order. Split indicates the number of digits intervening between the digits of interest (separation).

3. Results and Discussion

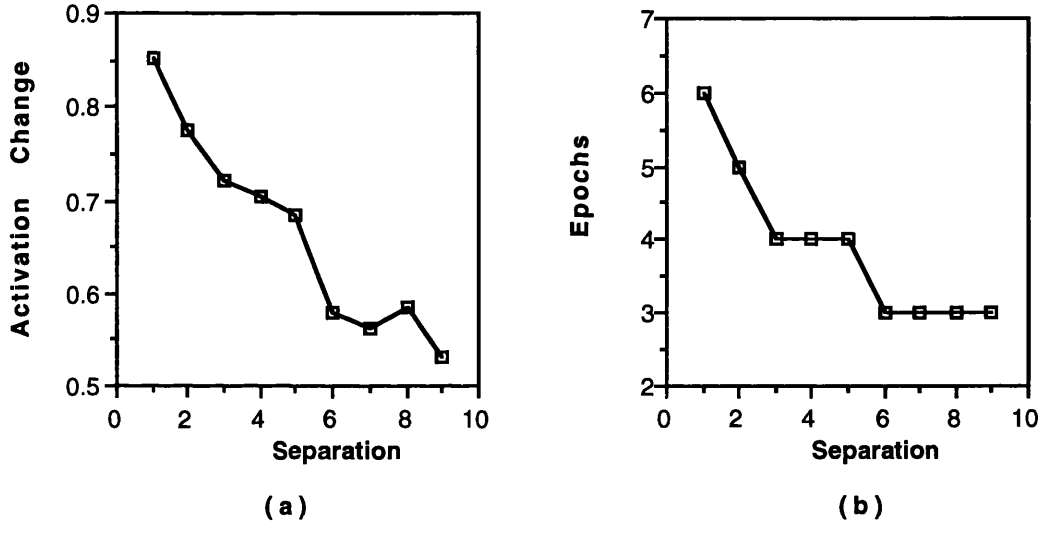
The weight configuration for the first attempt to approximate Parkman's data stemmed from the idea that as someone is taught a sequence it is usually referred to by its first and last members, in this case the numbers 1 to 10. This being true, the learner should strongly associate the first and last items in the sequence. To investigate this the weights between the nodes at distances 8 and 9 were increased to the same value as used for direct weights. The results are measured in the two ways outlined in Study One (section C), see figures 16a and 16b. The results, measured either way, are a close approximation to Parkman's results. Figure 16a shows the average change in activation for the nodes at each separation, graphed as one minus the activation change in order to resemble Parkman's RT graph. Figure 16b illustrates the number of epochs required to reach the maximum activation of the network, which is 0.973. Both figures are the average of the values received when each of the nodes was taken to be the initial node. This was found to produce a slightly better approximation of the RT data than when just the start node was activated. The figures show that when the performance of the network was measured by the average change in activation of the nodes, the network approximates the data well, except at the separation value 3, where the network predicts a higher value than the RT results produce. However, when epochs to maximum activation is used, the results are nearly the same as those obtained by Parkman.



Figures 16a and 16b. The Average Activation Change (a) and Epochs to Maximum Activation (b) in Simulation One. (a): The average change in activation (subtracted from 1) as a function of separation. The activation changes are subtracted from one to facilitate comparison to Parkman's data. (b): The number of epochs to reach maximum activation in this SON (0.973). The data for both figures is the average of the values received by running the simulation with each node in turn as the initial node. The network contained higher weight values on adjacent nodes and those at separations of 9 and 10.

Although the results of the first network approximated the data well, other attempts were made to investigate the effect of varying the weights between the nodes. In the initial network of a second set of configurations, the weight connecting nodes One and Three was lowered in an attempt to reduce the time taken for this node to build activation. This modification would only be useful in the start node activation condition. This network created more problems than it solved. Using both measures a plateau was found at separations of 1, 2, 3 and 4, their values being nearly the same. This led to a total restructuring of the weights in the network. The pattern of weights and parameters after restructuring is listed in table 8. The results for this study are illustrated in figures 17a and 17b. The results for the average activation change measure differed from the RT data in three places, no plateau was found

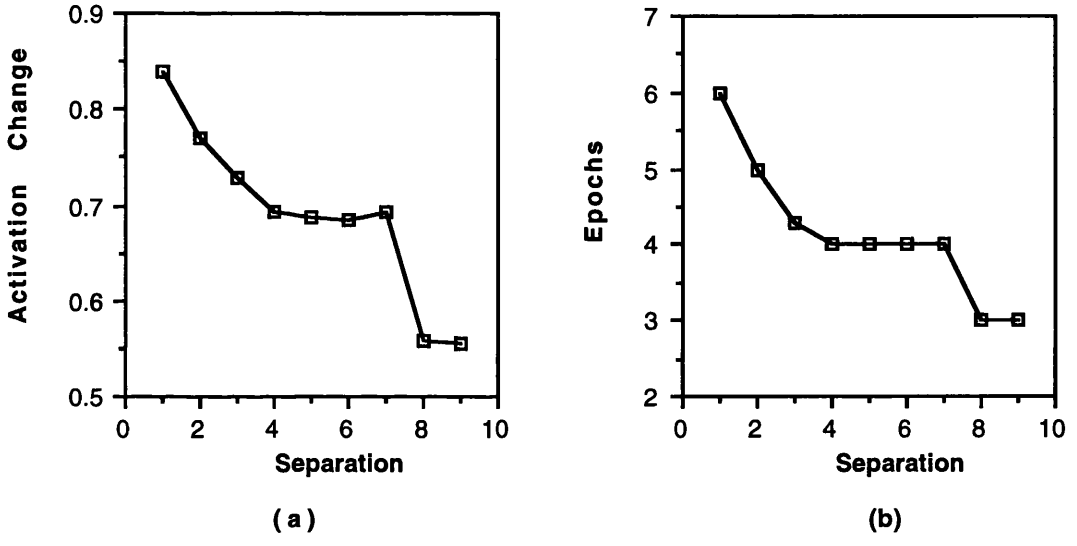
between separation values of 4 and 5 or 6 and 7. Also, the drop between separations 7 and 8 was absent, while the Epochs to maximum activation measure did not find drops between separations 3 and 4 or 7 and 8. The second deviation (between 7 and 8) was found using both measures and remained for all three initial conditions in which these separation values appeared, indicating that results are still consistent across starting conditions. Therefore, only the results obtained in the start node condition are graphed.



Figures 17a and 17b. Average Change in Activation (a) and Epochs to Maximum Activation (b) in Simulation Two. (a) 1 minus the average activation change as a function of separation and (b) the number of epochs to maximum activation (0.891), for a network with high adjacent connections (0.45) as well as various weight and parameter modifications in an attempt to model the RT data found by Parkman (1971).

The results from the second weight configuration are not as promising as those from the first, indicating that it would be useful to employ in the third set of simulations, configurations more like those in the first set. To this end third weight configuration was developed with strong adjacent connections, lower internal connections, and medium high long range connections (see table 9). Connections were especially low for separations 4, 5 and 6 to slow their activation build and allow the

longer range weights more time to build, thus eliminating the plateau in figure 17b for separation values of 6,7,8 and 9. While this tactic was successful in eliminating the plateau at the higher separation values, it exacerbated the plateau found among the medium range values 3, 4 and 5 extending it to 3, 4, 5, 6 and 7. This plateau appears in both measures - the activation value changes and the epochs to maximum activation. The results of this network averaged across all initial node conditions are illustrated in figures 18a and 18b.



Figures 18a and 18b. Average Activation Change (a) and Epochs to Maximum Activation (b) for Simulation Three. The values generated by a SON with strong adjacent connections, slightly lower long range connections, slightly lower yet mid-short range and mid-long range connections, and even lower middle range connections. (a) shows 1 minus the average change in activation, while (b) shows the number of epochs to reach maximum activation for the network (0.891).

Although they did expose the influences of the different network parameters, the results of both the second and third Studies went no further than Study One in modelling the RT data found by Parkman. The results from Study One however, indicate clearly that the SON possesses the capability of modelling Parkman's data with a high degree of accuracy.

F. Conclusions and Implications

The most important conclusion to be made from the results of these experiments is that the relatively simple SON is capable of producing the SDE found by RT studies of number comparisons. The mechanism would consist of a the SON connected to a decision network. The SDE encountered in order judgements is summarised as faster decisions when the digits of interest are farther apart in the sequence. This can be conceptualised as the nodes representing digits father away responding quicker than nodes representing closer digits. This capability may be measured in two ways - either by measuring the number of epochs the node of interest requires to build to the maximum activation value obtained by the node at separation 1, or by measuring the speed with which the activation of the node grows i.e. the faster a node's activation grows the quicker its response will reach a given threshold value. Either of these measures can be used from one of two starting conditions - either the node corresponding to the first digit encountered can be activated and the absolute time taken for the second node's response can be measured (initial node condition), or the start node of the sequence can be activated and the time taken for the second node to respond can be measured relative to that of the first (start node condition). Both of these measures, regardless of starting point, have been shown in this set of experiments to provide roughly equivalent results.

What is there, then, to choose between them? One way such a decision could be made is through comparison with RT data demonstrating effects other than SDE. In the initial node condition, the initial node is activated and the decision network returns a 'larger than' decision when a response is received from the target node. If no response is received, the default decision of 'not larger than' would be given. If time allowed, this decision could then be checked by reversing the order of the digits and attempting another 'larger than' decision. For the start node condition, the start node is activated then the nodes of interest feed activation to the decision network. This network returns a 'larger than' decision as soon as a threshold value is exceeded. If the activation

of the first node is always greater, a 'not larger than' decision is returned. Again, the decision could be verified by reversing the order of the digits and sending them back through.

In this case both proposed mechanisms are supported by RT data which show that decisions requiring the subject to judge stimuli in order opposite to that in which they are normally encountered, take longer than decisions of order in which the stimuli are as they occur in the sequence. This effect, termed the Direction effect, has been found using alphabetic stimuli, by Hamilton and Sanford (1978), Lovelace and Snodgrass (1971) and Parkman (1971), among others. Although this effect is not found with numeric stimuli, it is postulated that two SONs are at work in the numeric case, one for forward decisions and one for backward decisions.

Two effects which might differentiate between the mechanisms, are the SDE discontinuity and the Compression effect. Unfortunately, both effects seem to imply a different starting condition. The SDE discontinuity seems only likely to result from the activation of an initial node. Activation of the start node would allow all nodes to receive input from all downstream nodes. This build up of activation would swamp the effect of the large direct weight, which is thought to cause the discontinuity. It may be possible that a poorly learned sequence is sparsely connected, leading to an increased effect of the large direct weights in the start node condition. However, since this effect is found with numeric stimuli, it is unlikely to be due to sparse connections. Thus the SDE discontinuity would seem to indicate that the initial node condition is more appropriate. On the other hand, the Compression effect is only likely to occur as the result of activating the start node. The amount of activation passing to the nodes later in the sequence causes them to build to near threshold quickly leaving little in their activation values to differentiate between them. Whereas in the initial node condition, if all connection strengths are the same, comparison times for elements at the same separation value should be the same regardless of the elements being compared. However, this effect could be modelled

in the initial node condition if the connections between nodes varied with the position of the element in the sequence. The condition indicated by the Compression effect is therefore unclear.

To confuse matters still further, End-term effects also seem to imply that the start node condition is more appropriate. In this condition the decision mechanism could not use the activation of the start node and therefore an alternative decision strategy would be needed.

Although neither condition is clearly indicated, for simplicity's sake, the simulations in Chapter Four, which attempt to combine the SON with a decision mechanism to fully model the RT effects, will use the start node condition. The ambiguity over starting point and measure of order will be addressed again in the expansion of the SON model to include a decision mechanism in Chapter Four and in further modifications to the network in Chapters Seven and Nine. Chapter Nine introduces a learning mechanism which is necessary since the weights in the network may otherwise provide too many degrees of freedom.

Chapter Four - Extension of the SON Model

This section describes the extension of the PDP mechanisms for storing relative order information, the SON model (described in Chapter Three) to include the generation of responses. The resulting network is demonstrated capable of simulating relative order judgements from serial order information.

A. Network Architecture

The model presented here, pictured in figure 19, has two basic parts: the underlying SON, described in Chapter Two, which has been updated (outside of the MacBrain 3.0 envoronment) to include the ideas of Cascade from McClelland (1979) and augmented by a response generating section which uses a random walk as described by Link (1978).

1. Update of the SON

The underlying SON is made up of ten nodes in the present case, each node representing a number word and labelled One to Ten. Each node is connected to all downstream nodes. The activation flow through the network has been altered in accordance with the principles of Cascade, this allows the network to build activation in its nodes over time, amplifying the slow build due to the design of the network. The activation is now a running average of the net input over time:

$$A_i(t) \equiv \overline{net_i}(t) = \rho net_i(t) + (1-\rho)\overline{net_i}(t-1)$$

where:

$A_i(t)$	= Activation of node i at time t.
$\overline{net_i}(t)$	= Average input to node i at time t.
ρ	= A rate constant.

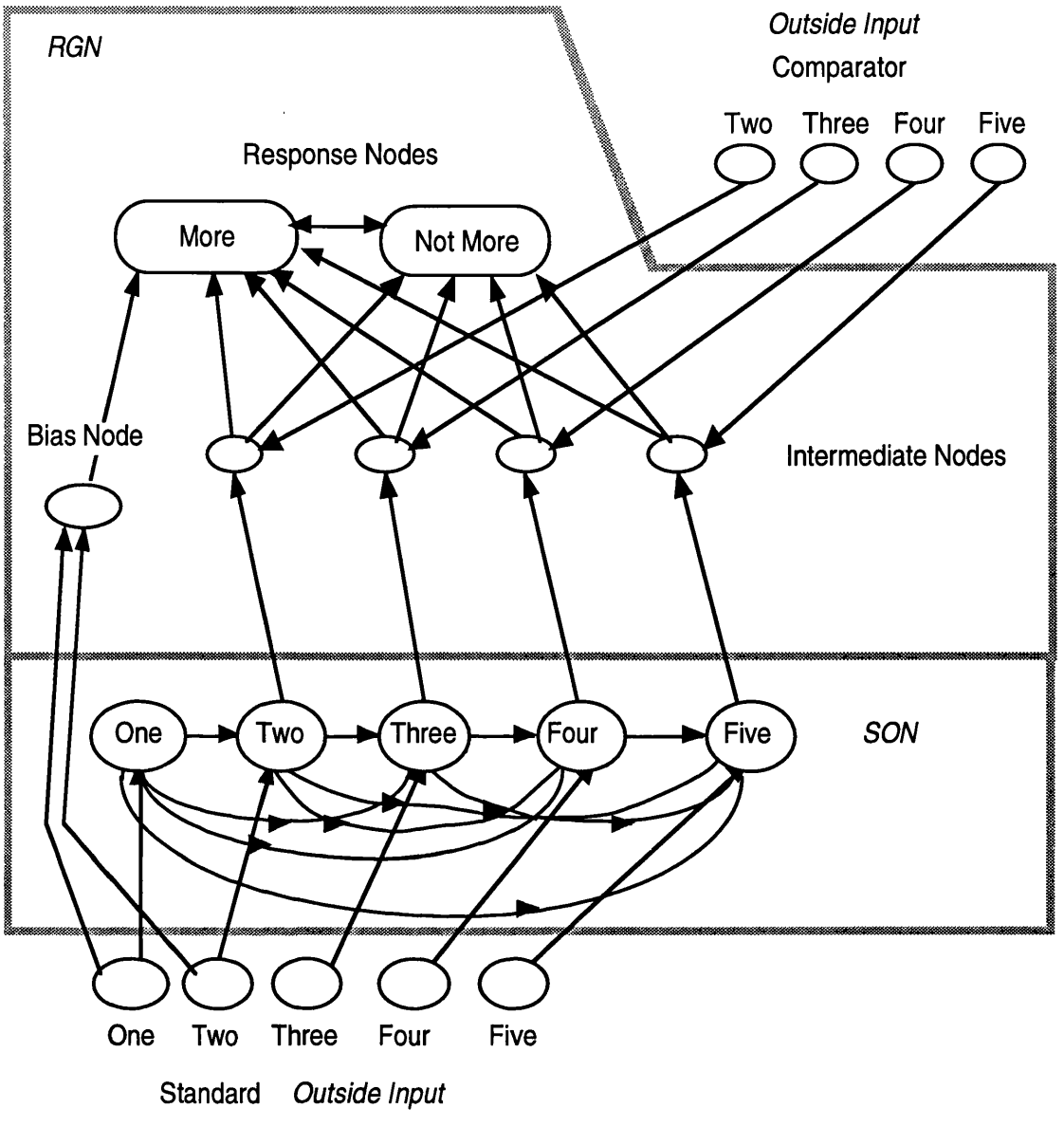


Figure 19. SON Augmented by a Response Generating Network. The Standard node is activated, thus the corresponding SON node is activated and the activation is passed through the SON. All intermediate nodes have a threshold of 1.0. The Comparator node corresponding to the number word on the right is activated and inputs (1 - β) to its intermediate node, whose threshold is then effectively (β). This allows (a_j - β) activation to pass to the response nodes, where j is the SON node corresponding to the Comparator. This value then becomes the mean of the Gaussian distribution from which the next step in the random walk is chosen.

This rate constant defines the time course of the process. A small (ρ) causes the units activation to build slowly while larger values of (ρ) allow quicker activation growth. A feature of this equation is that if a unit's input remains constant its activation will asymptote at a value equal to its input. In order to increase the power of the network it is important to introduce a non-linearity in the update rule. The most common way to do this is using a logistic (sigmoid) function (Cohen, Dunbar and McClelland, 1990) i.e. using the average input to the node as the net input into the sigmoid activation function. This is the method used in the extended SON.

$$A_i(t) \equiv \text{logistic}\left[\overline{net_i}(t)\right] = 1 \Big/ 1 + e\left[\frac{-\overline{net_i}(t)}{T}\right]$$

where:

$A_i(t)$	=	Activation of node i at time t.
$\overline{net_i}(t)$	=	Average input to node i at time t.
T	=	Temperature parameter.

Processing in this model is feed forward only. When the network is given two number words to compare, the standard is given an input of 1.0, which is passed to the corresponding SON node. The activation thus created is passed to all nodes down the network. Since each node in turn passes to all later nodes, the farther a node is from the standard the more active it becomes. All nodes to the left of the standard remain inactive. This restricts the network to **more** or **not more** decisions only, an aspect of the network that will be further addressed later in this report. As in Cohen, Dunbar and McClelland (1990), local representations are used to keep the model simple and interpretable. It is more than likely that a neural implementation of this network would have an ensemble of neurones for each node. See figure 19 for an illustration of the entire network including the SON.

2. Response Generation Network

The response generating network (RGN) is based on the random walk model as explained in Link (1978) and implemented in Cohen, Dunbar and McClelland (1990). The logistic function is not used to calculate activation in the Intermediate, Comparator or Response nodes. The Response node acts as an evidence accumulator. At the onset of each comparison the Response node's activation is cleared. As the network cycles, a small amount of activation is added to the Response node.

$$a_r(t+1) = a_r(t) + a_i(t+1) \qquad \forall \text{ intermediate node}$$

where:

$a_r(t+1)$ = Activation of Response Node at time t+1.
$a_i(t+1)$ = Activation of Intermediate node i at time t+1.

This added activation (a_i) is randomly distributed with mean, (μ_i) and a fixed standard deviation (σ_i). The mean of the distribution (μ_i) is based on the amount of activation passed from the nodes of the SON through the intermediate nodes to the response nodes.

$$a_i > \mu_i = (a_s - \beta_c)$$

where:

a_i = Activation of Intermediate node i.
μ_i = Mean of distribution of activation of node i.
a_s = Activation of SON node s.
β_c = Threshold of Comparator node c.

Thus, the intermediate node is noisy, responding with (a_i) - a more or less corrupted version of the input from the SON node (a_s). These corrupted activation values are evenly distributed around the mean (μ_i). This is implemented in the PDP environment by each intermediate node having a threshold of 1.0. This threshold effectively stops any activation from the corresponding SON node reaching the Response node.

$$a_i > a_s - Th_i + a_c = (a_s - 1) + (1 - \beta_c) = a_s - \beta_c$$

where:

a_s = Activation of SON node s, such that $0 \leq a_s \leq 1$.
Th_i = Threshold of Intermediate node i = 1.
a_c = Activation of Comparator node c = $(1 - \beta_c)$.

A second input received by the Intermediate node from a Comparator node effectively lowers the threshold of the Intermediate node. Since the Comparator node sends activation equal to $(1 - \beta_c)$ on every cycle, the lowered threshold of the intermediate node is equal to (β_c). Thus, the lowered threshold (β_c) determines the rate of evidence accumulation in the Response node. A response is given when the activation in one of the Response nodes reaches that node's threshold (A or -A).

> This response-selection mechanism may seem different from the rest of the network. For example, evidence is accumulated additively in the response-selection mechanism, whereas running averages are used elsewhere in the network. Additionally, the response-selection mechanism is linear, whereas the rest of the net is non-linear and relies on this non linearity. In fact, we can easily show that the additive diffusion process can be mimicked with linear running averages, by assuming that the response criterion gets smaller as processing goes on within a trial. The impact of introducing non-linearity into the evidence accumulator is less obvious. However, it need not exert a strong distorting effect, as long as the threshold is within the linear mid-portion of the accumulation function. (Cohen, Dunbar and McClelland, 1990, p.338)

Response Bias

Link (1990) pointed out that decisions about items at the beginning of the list may be faster than decisions about those at the end, due to the effect of response bias and not to some aspect of the underlying representation (i.e. Compression). To model this, there is a bias node connected to the Response nodes. The bias node is a

slow refractory node, which has the effect of providing a burst of input to the Response nodes when activated. It spends the rest of the time the network requires to deliver the response in recovery. Thus net input into the Response node is:

$$net_R = a_i + a_b$$

where:

net_R = Net input to Response node R.
a_i = Activation in Intermediate node i.
a_b = Activation of response bias node b.

This node has the effect of changing the starting point of the random walk. To more accurately simulate the random walk used by Poltrock (1989) in modelling digit comparisons, this network has two response nodes the **More** node with a positive threshold (A) and the **Not More** node with a negative threshold (-A). An inhibitive connection joins them. In this way when one node reaches threshold it responds and inhibits the other node, allowing only one response to be given on each trial.

B. Investigation of the SON/RGN Model

Now that the SON has been expanded and extended, it is able to generate responses that can be compared directly with RTs found in experiments investigating relative order judgements. This section examines the model's ability to predict these RTs and compares it to the Random Walk implementation described and discussed by Poltrock (1989). A full description and discussion of the equations he used can be found in Chapter One, section C.1.3 of this thesis. This chapter only discusses the ability of his model to simulate the experimental results.

Since the SON/RGN network is being compared to Poltrock, it seems most effective to follow his lead and use the results of his initial human experiment to set the variable parameters for the SON/RGN network. The resulting network configuration is then compared to the results of a digit comparison study run by Parkman (1971) to indicate whether the results of the first

simulation would generalize to other studies. Poltrock is then returned to in order to compare both models' abilities to simulate the speed/accuracy trade-offs associated with digit comparison tasks.

1. Update of SON Parameters

The description of the SON in Chapter Three looked at the type of weight structure that is likely to lead to the type of results Parkman found. For the purposes of this Chapter, the most promising weight structure (from section E, simulation one) is taken as fixed. This structure had relatively large direct and long-range connections (0.35), with the remaining connections smaller (0.25). However, to slow the build of avtivation the weights were lowered to (0.25) for direct and long-range connections and (0.15) for the remainder. This fixed weight structure is presumed to correspond to a fully trained SON, or at least to be representative of the amount of training the network has received up to the point of the relative order task.

The Temperature parameter of the SON is also fixed. The necessity of this parameter was found in Chapter One i.e. as it is in the denominator of one of the terms of the logistic function, its absence causes the equation to behave oddly. Since this network is no longer implemented using the MacBrain simulator, the Temperature has been increased slightly from the default of (T = 0.1275) to (T = 0.1503), thus increasing the amount of activation in a node with the same amount of input. This parameter did not vary.

With the addition of the Cascade of activation in the SON nodes, it is necessary to set the rate parameter (ρ), to a value between zero and one. This parameter controls the build of activation in the nodes. The aim is for activation to build slowly to allow access to the network's potential to model the non-linearity in the SDE curve, which appears within the first couple of epochs of activation flow through the SON. Initial network runs showed that a low value of (ρ =0.07) provided a suitably slow build up of activation. It was set to that and held fixed.

The slowing of the build of activation in the SON is designed to allow the random walk to operate before all nodes reach maximum activation.

The only variables left are the parameters of the RGN. One of these parameters is also fixed i.e. the standard deviation of the noise added to the activation passed to the Response node (of the random walk parameter (σ_i)). This parameter is fixed as there is no theoretical reason for it to vary. As in Cohen, et al. (1990) the standard deviation is set to ($\sigma_i = 0.1$).

As the subjects modelled read from left to right, the left number word is always taken as the Standard and the right the Comparator.

The variable parameters are all random walk variables. They include the start of the random walk which corresponds to the activation of response bias node (a_b). Variable parameters also include the threshold required to be reached before a response is given (A and -A). Also variable is the effective threshold of the intermediate nodes (β), which controls size of the steps of the walk. The higher this value in relation to the mean (μ), the smaller the steps. The values for these parameters will be discussed in association with each simulation.

2. Setting of RGN Parameters

The results from Poltrock's (1989) experiment are used to set the variable parameters of the RGN, and an equation used to translate epochs to decision into RT.

2.1 Estimation of Variable Parameters

The first parameter investigated was the effective threshold of the Intermediate nodes (β). This parameter determines the size of step taken at each epoch. To begin, one must look at the pattern of activation in the SON nodes. Figure 20 shows how activation in the network develops over the first 5 epochs when the standard is Two. It is apparent that activation stays near 0.50. Trials began

with effective threshold (β) values near 0.50 to keep the mean value for the steps in the random walk (μ) near zero, which allows mistakes to be made by the network. The further the mean of the steps from zero the fewer mistakes are made, because all steps are positive and lead to (A), which is always the correct answer. Negative steps occur only if the noise added by the Intermediate node causes a number of negative steps, the higher the mean of the distribution, the less are likely negative responses to be produced. The exact percentages of errors reported by Poltrock were between 0.000 and 0.183. This percentage increased as the distance between the two number words decreased, the lowest proportion of errors for distance 1 occurred at pairs (1,2) and (2,3). The value of (β) was manipulated in an attempt to keep as close as possible to these error proportions. A comparison of the error rates obtained with (β) values of 0.52 and 0.513 and those obtained by Poltrock is displayed in figure 21. This graph shows that both (β) values of 0.52 and 0.513 generate too few errors for separation values of 2 or more, while (β) of 0.52 generates too many errors for separation 1 and 0.513 too few. Because of the way in which (β) interacts with the other variable parameters, the value of 0.513 was chosen as the most effective. Larger values caused too large an increase in the number of errors with too small a change in threshold (A).

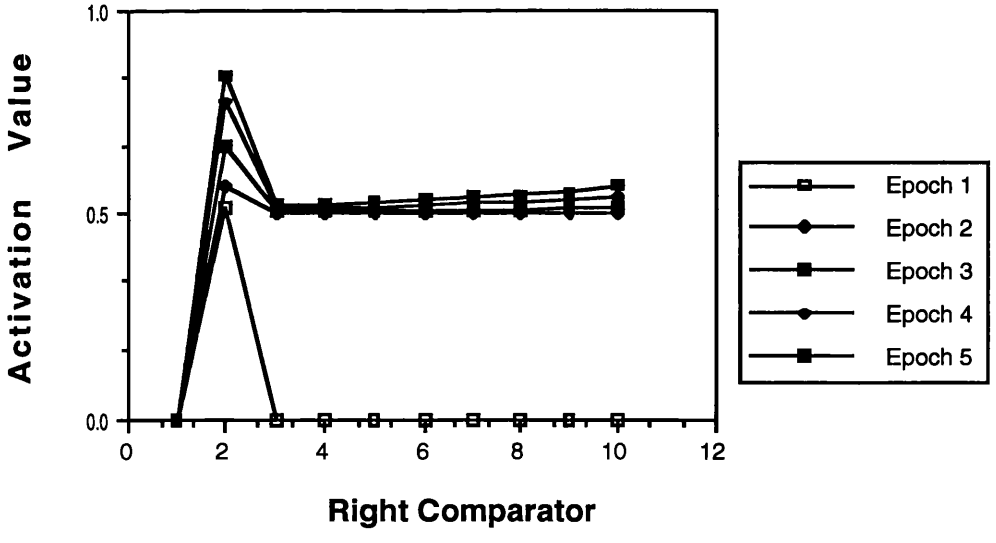


Figure 20. Activation Values in the SON/RGN for the First Five Epochs. The activation values that develop over the first five epochs in the SON network when the standard is Two. Note that all values remain near 0.5.

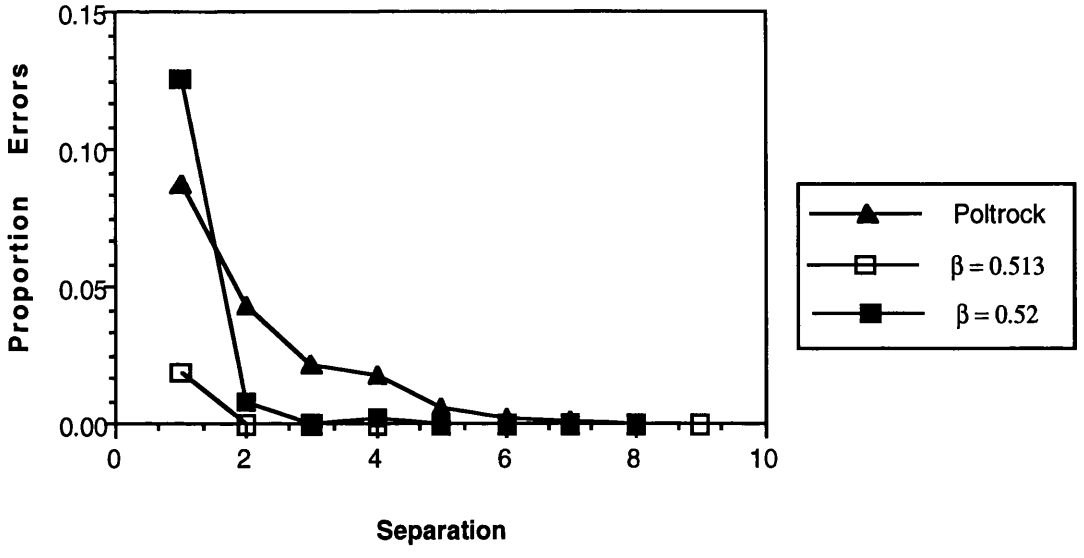


Figure 21. Comparison of Error Rates. Comparison of error rates in a SON/RGN network with β value 0.52, 0.513 and the error rates found for human subjects by Poltrock (1989).

The second parameter to be investigated was the response bias (a_b) for each standard. Poltrock reported the RTs and error rates obtained from his subjects for each digit combination and separated trials in which digits occurred in reverse order i.e. (4, 5) and (5, 4). This data can be used to compare the change in percentage RT for each digit pair, excluding pairs in which the Standard was greater than the Comparator. This RT change represents a measure of response bias. To simplify the process the pairs at separation 1 were looked at first. The pair (5, 6) is taken to represent an unbiased RT, since (5) represents the central number and therefore neither answer was more statistically likely to be correct. In Poltrock's data, from (5, 6) there is a reduction of 14% in reaction time for the pair (1, 2) and 4% for pair (2, 3). All other reductions were below 2% except for pair (4, 5) where a seemingly anomalous reduction of 7% occurred. This 7% reduction did not conform to any pattern of response bias and may have been due to the random element in human responses. The response biases responsible for these decreases in RT can be simulated by increasing the starting point of the random walk. In our model, RT

is represented by the number of epochs to solution. This value reduces with the distance between the start of the random walk and the threshold to be reached.

Starting Point	% decrease in Epochs
0.01	0.00%
0.02	14%
0.03	8%
0.04	7%
0.05	11%
0.06	11%
0.07	15%

Table 10. Effect of Changing the Starting Point of the Random Walk in the RGN. The reduction in the number of epochs to decision generated by increasing the starting point of the walk from 0.000.

Table 10 shows some starting points and the subsequent percentage reduction in epochs to solution. From this table a starting point of 0.07 was chosen for comparisons in which the Standard was (1), leading to the 14% reduction in RTs. A starting point of 0.02 was chosen for comparisons in which the Standard was (2), to account for the 4% reduction in RTs. The second choice may seem odd, but it was decided that the 14% reduction in epochs to decision was not due only to the change in starting point, but was an artefact created by the random component of the walk. Table 1 is based on 6 trials with 84 comparisons per trial, by looking at the raw data one can see that the trials with the starting point of 0.02 had a much larger number of error trials which seem to cause an unnecessary reduction in the number of epochs to solution per trial, which in turn causes an abnormally large decrease to show up in the table.

The thresholds (A and -A) for this simulation were +1 and -1, these corresponding to **More** and **Not More**, respectively. Only

More decisions were correct. **Less** decisions are discussed in the network modifications section.

2.2 Estimation of RTs from SON/RGN

To compare the results of the SON/RGN with those of Poltrock it is first necessary to calculate predicted RTs for the SON/RGN network. Since the number of epochs to solution is believed to represent the amount of time taken to reach a conclusion, it is necessary to determine the relationship between number of epochs and RT. Initially, it is noted that the number of epochs is highly correlated with the RT. Thus, a relationship can be determined by linear regression of the number of epochs to solution against RT. Although hypothesising a linear relationship between epochs and RT is a strong claim, it is necessary for a prediction to be made without which no interesting comparisons between the SON/RGN and other Response Estimation systems can be made. This claim has been made by previous researchers for the same reason - see Cohen, Dunbar and McClelland (1990).

Number Pair	Reaction Time	Average # of Epochs to soln
(1,2)	379	22.33
(2,3)	427	28.96
(3,4)	442	34.78
(4,5)	412	29.14
(5,6)	443	28.82
(6,7)	447	33.90
(7,8)	439	32.27
(8,9)	435	40.45

Table 11. Comparison of RTs and Epochs to Solution in SON/RGN. This table shows the reaction time and number of epochs to solution for each number word pair at separation 1. Reaction times are averaged over all six of Poltrock's subjects (14 trials per subject = 84 trials). Epochs to solution are also averaged over 84 trials. r = 0.72 indicating that these values are highly correlated.

Since number pairs with separation one were used as guides for determining the network parameters, they were chosen to determine the regression equation for predicting RT based on number of epochs to solution. Table 11 shows the RTs to each number pair at separation 1, averaged over Poltrock's six subjects (14 trials each = 84 trials) and the average number of epochs to solution in the SON/RGN network over 84 trials. These values have a correlation of (r = 0.72), which is a high correlation, this makes them suitable for regression. The equation to predict RT from the number of epochs to solution is:

$$Y_{RT} = 332.25 + (3.056)X_{ES}$$

where:
RT = reaction time.
ES = epochs to solution.

This equation implies that each epoch requires 3.056 msec on top of a constant 332.25 msec for underlying processes. These underlying processes may include percieving and encoding the stimulus, and initiating and performing the response.

3. Simulation One

In this simulation Poltrock's (1989) results are compared to the SON/RGN configuration that they have been used to create.

3.1 Procedure

The stimuli used were the digits (1 - 9) paired such that all separation values (1 - 8) were represented. Each stimulus pair was presented to the network eighty-four times, to match the number of trials which Poltrock recorded for each subject. The number of errors and average number of epochs to solution for non-error trials were recorded.

The regression equation generated in section 2.2 was then used to calculate the predicted RTs for each separation value, which were then compared to Poltrock's findings. This comparison was restricted to forward trials only.

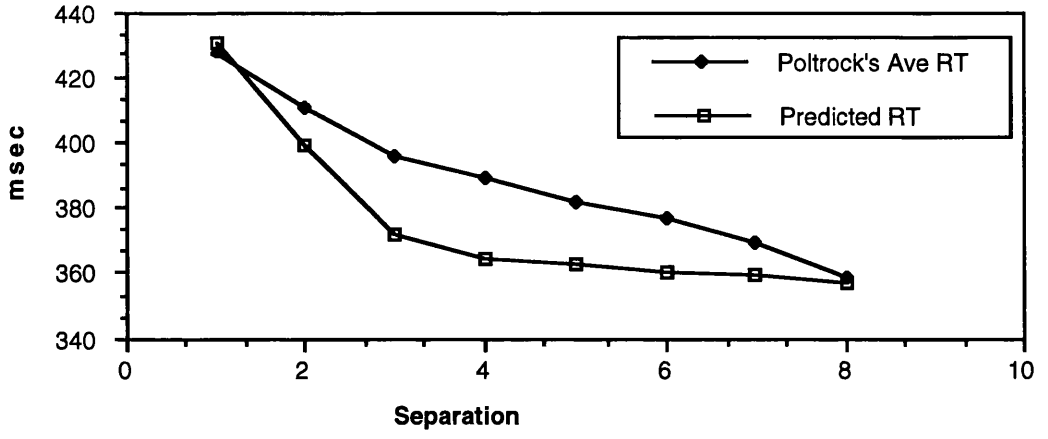


Figure 22. RTs Predicted by SON/RGN v RTs Observed by Poltrock. Subjects were allowed to take as long as desired, thus, error rates were low (0% - .183%). The SON/RGN error rates was slightly lower (0% - .019%). Poltrock did not report his predicted RTs, therefore they are not graphed.

3.2 Results

A comparison of the RTs predicted from the SON/RGN to the RTs found by Poltrock is graphed in Figure 22. Poltrock used the data from individual subjects to tailor his model. This tailoring involved changing the starting point of the walk, threshold and a constant non decision time for each subject, along with a value for the mean analog magnitude for each digit. The differences in mean analog magnitude for each digit would be equivalent to changing the weights in the SON network for each subject. Despite this advantage, Poltrock's model had difficulty simulating the data due to the extreme rapidity with which subjects responded. For pairs containing (1), four of his subjects seem to reach response threshold in a single time step, indicating that the difference in mean analog magnitudes for the digits in the pair (μ), which he calls the mean drift rate, for these digits exceeds threshold (A). This violates the principal that ($A >> \mu$), which is necessary for the validity of one of his networks' equations. To overcome this problem Poltrock analysed only trials with number words (2) through (9). With this restriction, the highest correlation he achieved between predicted and actual RTs was $r = 0.95$. The

correlation for the mean of his subjects' data was $r = 0.92$. He does not record the error rate achieved by his model.

For the SON/RGN, predicted and actual RTs for the mean data, as access to individual data was unavailable, have a slightly higher correlation than Poltrock achieved ($r = 0.93$). This is achieved without restructuring the network for individual subjects or restricting the pairs modelled. As only activation from one node is used to calculate stepsize, the mean drift rate is simulated by the equation ($\mu = a_j - \beta$), which remained much less than threshold. The SON/RGN produced (0% - 0.018%) errors in comparison to (0% - 0.183%) produced by Poltrock's subjects. Thus, the model is doing slightly better than it should.

3.3 Conclusions

The results indicate that the SON/RGN is capable of modelling the same data as the Poltrock model, with the same degree of success, without the difficulties that his model encountered.

For future simulations it would be more effective to model the results of individual subjects, as Poltrock did, instead of the mean results of several subjects. Due to the data available for these and the remaining simulations in this study, it is perhaps more accurate to view the SON/RGN as a prototypical representation, a general purpose serial order judgement machine, and not an attempt to simulate any individual's, or set of individuals', representation of the number word sequence.

4. Simulation Two

Simulation Two is a comparison of the network described in Simulation One to the RTs found in the experiment by Parkman (1971). This simulation is designed to indicate whether the results in Simulation One generalize to other sets of data.

4.1 Procedure

The variable parameters in the network and the stimuli used remained as they were set in Simulation One. Each stimulus pair

was presented to the network fifty times (the number of observations in the Parkman study was less than in Poltrock's).

The regression equation generated in section 2.2 of Simulation One was again used to determine the predicted average RT at each separation value for the network.

Estimates of the average RTs at each separation value from the Parkman paper are made as the actual RTs are not reported. The estimates are taken from the graph of his results.

4.2 Results

The relationship between the average number of epochs to solution for each letter separation value in the SON/RGN and an estimation of the average RTs obtained by Parkman (actual RT values were not reported in his paper) is illustrated in figure 23. The correlation between these values is very high (r = 0.95).

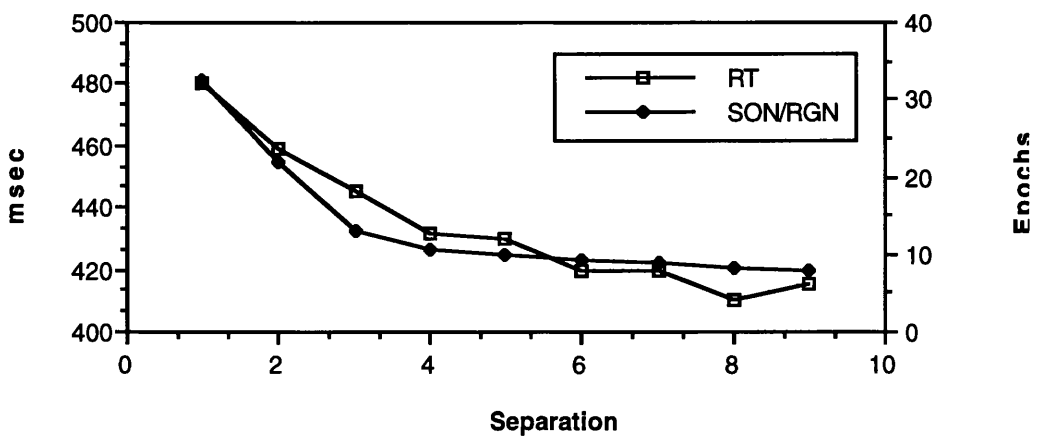


Figure 23. Average RT Observed by Parkman v Average Epochs to Decision in SON/RGN. The relationship between average epochs to decision and estimation of RTs obtained in Parkman (1971). These values are very highly correlated (r = .95).

The regression equation is then used to generate predicted RTs for the purpose of comparison. The correlation between the RTs predicted and an estimation of the RTs obtained by Parkman is

demonstrated in figure 24. The estimations and predicted RTs differ by a constant amount. This constant difference may be due, amongst other experimental variables, to an increase in the non-decision time for Parkman's subjects or a difference in the degree of learning between the subjects in the two experiments.

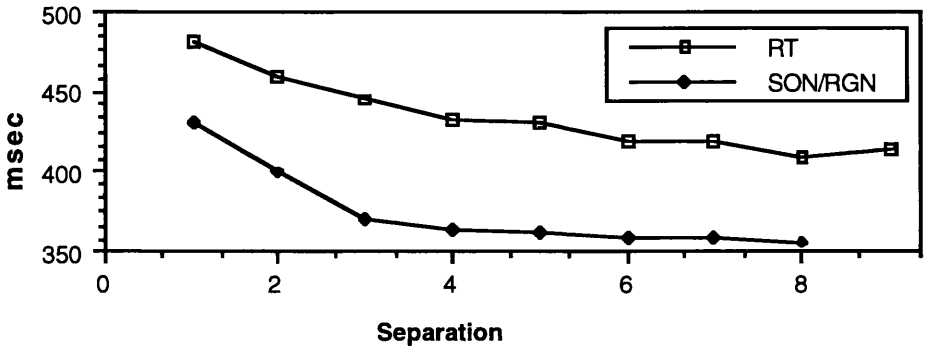


Figure 24. RTs Predicted by SON/RGN v RTs Obtained by Parkman. These lines differ by a constant amount. This constant difference may be due to an increase in non-decision time or difference in learning between subjects or other experimental variable.

4.3 Conclusions

This simulation confirms that the model's ability to replicate Poltrock's (1989) data is not idiosyncratic. However, it does indicate that either the model's parameters or the equation relating epochs to RT should be updated when modelling a new set of data. The regression equation which would allow better predictions to be made in Simulation Two is:

$$Y_{RT} = 398.84 + (2.64)X_{ES}$$

not

$$Y_{RT} = 332.25 + (3.056)X_{ES}$$

The largest difference between these two equations is in the Y-intercept term, leading to the assumption that the non-decision processes were longer for the subjects in Parkman's study. The non-decision processes are expected to remain constant across comparisons and thus be represented in the Y-intercept, while decision processes are expected to change and therefore be

represented by the slope. Non-decision processes may be related to the response required in the experiment or other subject variables. It is unlikely that this difference is in the weights of the SON or parameters of the RGN as changing these would lead to fewer epochs to decision, which does not appear to be required.

5. Simulation Three

Simulation Three is a further comparison between the SON/RGN network, Poltrock's random walk model and the RTs found by Poltrock in human experiments. In this experiment subjects were instructed to respond within a set amount of time. This time limit was designed to cause the subjects to produce 10% - 20% errors, based on their performance in Experiment One.

5.1 Procedure

This simulation used the same stimuli and procedure as in Simulation One (Section B.3).

However the threshold of the random walk was lowered to generate the decrease in RT and increase in errors that Poltrock found. Reducing the threshold of the walk increases the size (relative to threshold) and importance of each step, since fewer steps will lead to a response. This increases the likelihood of errors, without the need to increase the likelihood of missteps being made, limiting the number of free parameters that must be varied.

5.2 Results

Poltrock's model continued to have difficulties in this simulation. As well as the continuing problem with pairs containing (1) appearing to cross the response threshold in a single time step, the effects of distance and minimum digit were smaller, therefore more difficult to model. The random walk model could account for a smaller proportion of RT variance and could consequently not be fitted to the results of each subject. He found that although the random walk model should account for the increase in errors and decrease in reaction time by only a change in threshold, his model

was unable to do this - he also needed to change the non-decision time and starting point of the walk.

Other aspects of the results his model could not explain, were the faster and more accurate decisions made when the Standard was less than the Comparator (direction effect) and the increase in the effect of response bias when both digits were greater than five. The first of these findings appears to necessitate a non-symmetric random walk, while the other requires the postulation of two response biases for a single number word. Poltrock's overall results were a correlation of (r = 0.74) for number words (1) - (9) and a correlation of (r = 0.63) for number words (2) - (9).

The SON/RGN network had an easier time than the random walk with this simulation. Since Poltrock did not report the individual data, one cannot say whether the SON/RGN network would be able to model it, but it seems that this network would have the capability of modelling subtle differences due to the variance available in the weights of the SON.

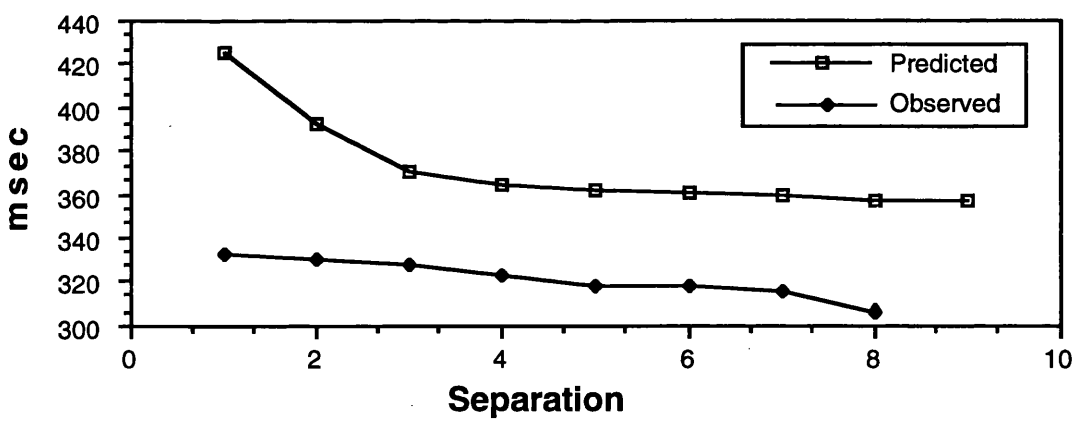


Figure 25. RTs Predicted by SON/RGN v RTs Observed with 10% to 20% Errors. Comparison of the RTs predicted by the SON/RGN network and the RTs observed by Poltrock in his 1989 study. Subjects and network were allowed to make 10% to 20% errors.

The results of the SON/RGN, illustrated in figure 25, were gained through changes in the threshold of the response node only. Although a high correlation was achieved (r = 0.80), it is evident in

the graph that the predicted RTs differ by a constant from the RTs of Poltrock's subjects. This could be explained either by decrease in what Poltrock refers to as non-decision time (the Y-intercept in the regression equation), or by a change in the β value in the SON network. Changes in non-decision time may relflect the subjects voluntarily increasing the speed of their response initiation and/or execution. Alternatively, a change in the β value could be brought about by increasing the activation provided by the Comparator to the Intermediate node, the motivation being that the threshold in the Comparator nodes is lowered due to general excitation of the system.

In theory, the SON/RGN model is also capable of simulating the faster and more accurate decisions made when the Standard was less than the Comparator, without postulating a non-symmetric random walk. It is able to do this because decisions in which the Comparator is larger would be taken by a different network. It is also able to explain greater effects of response bias when both number words are greater than five because there are independent connections from the Standards and Comparators to the bias node. If the connections are weak then they would only be noticed if more than one of the nodes was active. Mean analog magnitude differences are not explicitly calculated in this model and there are no assumptions made as to their relationship to the random walk boundary. But, again, on no trial was the boundary reached in a single step.

5.3 Conclusions

This simulation, while causing the SON/RGN network some difficulty, does not cause as much as was experienced by the traditional random walk model as implemented by Poltrock. Indicating that the SON/RGN network is a step forward.

6. Simulation Four

The purpose of this simulation was to indicate whether the SON/RGN network was capable of simulating varying degrees of error and RT length. Poltrock designed this experiment to cause his

subjects to make 5%, 15% and 25% errors, by asking them to react within shorter and shorter periods of time. Poltrock found that (1) error rates increased as the RT target decreased, (2) error rates increased as the separation between the number words decreased and (3) error rates were greater when the standard was greater than the comparator. In Poltrock's model these effects are attributed to changes in different variables. (1) is attributed to a change in response threshold, (2) to changes in the difference between mean analog values of the different number words (drift rate) and (3) to changes in response bias (starting point of walk).

6.1 Procedure

Stimuli and procedure were the same as used in Simulation One (Section B.3) and Simulation Three (Section B.5).

Changes were made to the threshold of the response nodes in an attempt to decrease RTs and increase errors to reproduce Poltrock's results. The SON/RGN network produced results by changing only this boundary as this is the change which is considered to produce a reduction in RT. The difference between mean analog values, which are a direct result of the weights between the SON nodes in the SON/RGN model, must remain constant across all trials and are therefore designed to simulate error rate changes without being altered. Postulating changes in these values implies postulating a change in a subject's underlying representation of a sequence during testing, which is not plausible. Poltrock too, rejected this as a possibility.

It is plausible that response biases from some of Standards and Comparators are quite small and only become apparent when the boundary is lowered. However, this change is best considered in the light of individual data which was not available.

6.2 Results

As in the previous simulation Poltrock had to separate comparisons with (1) as the boundary appeared to be reached in a single step. For the remaining comparisons with (2) - (9), he achieves correlations between (r = 0.36 and r = 0.79) when 5%

trials are errors, correlations between (r = 0.31 and r = 0.67) when 15% of trials are errors and (r = 0.11 and r = 0.43) when 25% of trials are errors.

The problem most apparent in the SON/RGN network was the lack of reduction in RTs. The correlations between predicted and obtained RTs were high on the 5% and the 15% error conditions, but much lower on the 25% error conditions (r = 0.85, r = 0.88 and r = 0.24, respectively), but the predicted RTs were substantially higher than observed. Predicted RTs ranged from 425 msec to 355 msec in the 5% condition, while observed Rtswere between 380 msec and 350 msec. This discrepancy increased in the other two conditions because predicted RTs did not vary to any significant degree, while observed RTs reduced to between 290 msec and 275 msec in the 25% error condition.

The SON/RGN model also failed to exhibit the flattening of the RT by separation curve found by Poltrock and illustrated in figure 22. This flattening does not seem to be beyond the SON/RGN network's abilities to model, because the early activation levels of the SON nodes are very similar. To tap the information in these early activation levels it may be necessary to reduce the β value to allow the response threshold to be reached in fewer steps, or increase the background level of activation in the network, which might also cause the step sizes to increase.

In figure 26, one can also see the differences in the RTs obtained between trials in which the larger number word was the Comparator or the Standard (direction effect). This is strong evidence for a non-symmetric random walk and, thus, lends further support to the SON/RGN model over the traditional random walk models.

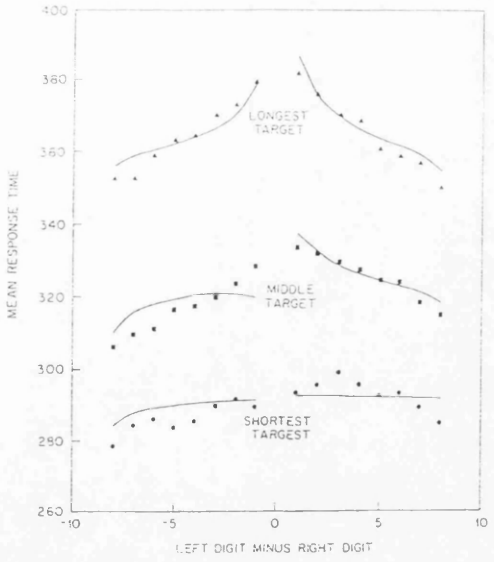


Figure 26. RTs by Separation when Errors are Made on 5%, 15% and 25% of Trials. Taken from Poltrock (1989) shows the pattern of RTs by separation when errors are made on 5%, 15% and 25% of trials. Separation is denoted by value of number word on the left minus value of number word on the right.

6.3 Conclusions

Although both models experience difficulty in explaining the effects of subjects' voluntarily decreasing their RTs, the SON/RGN appears better able to address these effects within the constructs of the random walk. That is, the SON/RGN approaches an explanation of these effects which requires only a change in the threshold of the random walk, and perhaps the non-decision processes, which is how the random walk was designed to operate. The Poltrock model, however, requires changes be made to the response bias and drift rate as well. With these changes Poltrock's model achieves a better prediction of performance in the condition in which 25% of trials result in errors, but the SON/RGN outperforms it on the 5% and 15% conditions, as well as producing a correlation in the 25% condition, which is higher than the lowest correlation produced by Poltrock's model.

Difficulties experienced in the 25% error condition may indicate that the β value of the walk is also under conscious control or that an increase in background activation in the network diminishes the differences between pairs leading to the flattening of the SDE curve.

C. General Conclusions

Initially, Poltrock's results were used to set the variable parameters of the SON/RGN. The configuration thus developed was shown in Simulations One and Two to be capable of modelling Poltrock's data and generalising to model data recorded by Parkman. Simulations were then pursued to investigate whether the SON/RGN was capable of modelling data in which subjects were believed to be varying the parameters of the random walk. In this instance the theory behind the random walk states that the only variable which is under conscious control, and therefore capable of varying between trials, is the random walk boundary. That is, the subject can decide to respond even though the evidence that their response is correct has not reached one hundred percent certainty. The results of the SON/RGN predictions on these simulations were also compared to those of Poltrock to determine if a purely mathematical random walk would produce a more accurate simulation of the data.

It was found in both Simulations Three and Four that the SON/RGN did not experience the same degree of difficulty as Poltrock's random walk. When using the mathematical model, it was found that for very small separation values subjects appeared to be reaching the random walk threshold in a single step. This violates the assumption behind the equations used to implement the walk, which is that step size is much smaller than the threshold. This problem was not experienced by the SON/RGN as step size always remained smaller than threshold. However, if step size had exceeded threshold this would not have invalidated the equations in the SON/RGN.

The mathematical random walk also experienced difficulty in comparisons involving (1) as these comparisons did not appear to show the SDE. Since this model requires that all digits have different mean analog magnitudes and all step sizes be much less than threshold, it is unable to produce consistent RTs for all comparisons involving (1). The SON/RGN, however, can use the bias

node to ensure that all comparisons with (1) produce roughly the same number of epochs to solution.

Despite Poltrock only modelling comparisons between the digits (2) to (9), the SON/RGN produced similar results in Simulation Three, and better results in the 5% and 15% error conditions in Simulation Four. In Simulation Four the SON/RGN also managed to produce these results while only varying the threshold of the walk, which was postulated to be the only aspect under conscious control. Poltrock found that he also had to vary drift rate and response bias. The 25% error condition in Simulation Four caused both models to produce very low correlations. These may indicate that a second parameter is also under conscious control or it may be a result of general anxiety to respond quickly, increasing the background activation in the network, and causing differences between pairs to be diminished.

In general, it appears that the SON/RGN model is more promising than the purely mathematically random walk. The SON/RGN contains more latitude in which to be altered to represent other types of sequences. It also has the ability to learn, which the mathematical random walk is incapable of doing. The possible changes are addressed in the next section and learning is addressed in Chapter Six.

D. Modifications to the SON/RGN

In this section extensions to the network are motivated and implementation discussed.

1. Relationship Between Forward and Backward Sequences

In examining the order judgement literature, one often finds that stimuli produce direction effects. These direction effects are believed to be due to the functioning of two separate order judgement networks, one for forward comparisons and one for backward. Evidence for the existence of separate forward and backward judgement networks can be found in the research on the

acquisition and elaboration of the number sequences. Pike and Olson (1977) found that immature children seem to have a representation of numbers that is in terms of **more**. For these children to answer questions about the relationships between numbers, they could answer **more** questions quickest, followed by **not more**, with **less** questions proving the most difficult. Pike and Olsen hypothesised that this difference occurred because in order to answer **less** questions the immature children have to recode **not more** answers. That is, they are only able to judge if one digit is **more** or **not more,** all other answers must be derived from these. Mature children appear to have two representations - one for **more** and the other for **less**. Addition appears to activate the **more** net and subtraction the **less** net.

The Congruence effect may also indicate the presence of more than one network. It is found in research into relational judgements i.e. preferred colours are selected more quickly when subjects are asked which colour is preferred and more slowly when subjects are asked which colour is less preferred (Jamieson and Petrusic, 1975). This difference may be due to the different instructions accessing a different network, whose elements are structured in the opposite order.

Further evidence of distinct representations for forward and backward orderings has also been found in the acquisition of sequences themselves. Subjects do not appear to have access to previous elements of a sequence unless they purposely learn the sequence in reverse order. The contention being that learning a sequence in reverse order generates the second network. Evidence of this is found with the alphabet and number series. When subjects are asked questions like "Which letter comes before Q ?" they report run-through proceeding in the forward direction only, for example, "L M N O P Q", the subject then reports the answer "P" (Hamilton and Sanford, 1979). In contrast, if asked "Which number comes before 9?" at some point children begin to say "10, 9, 8" and report the answer "8". This leads to the hypothesis that these differences are found because two networks are used to represent forward and backward relationships between sequential

stimuli, in this case number words but not letters. A possible relationship between the two networks is shown in figure 27.

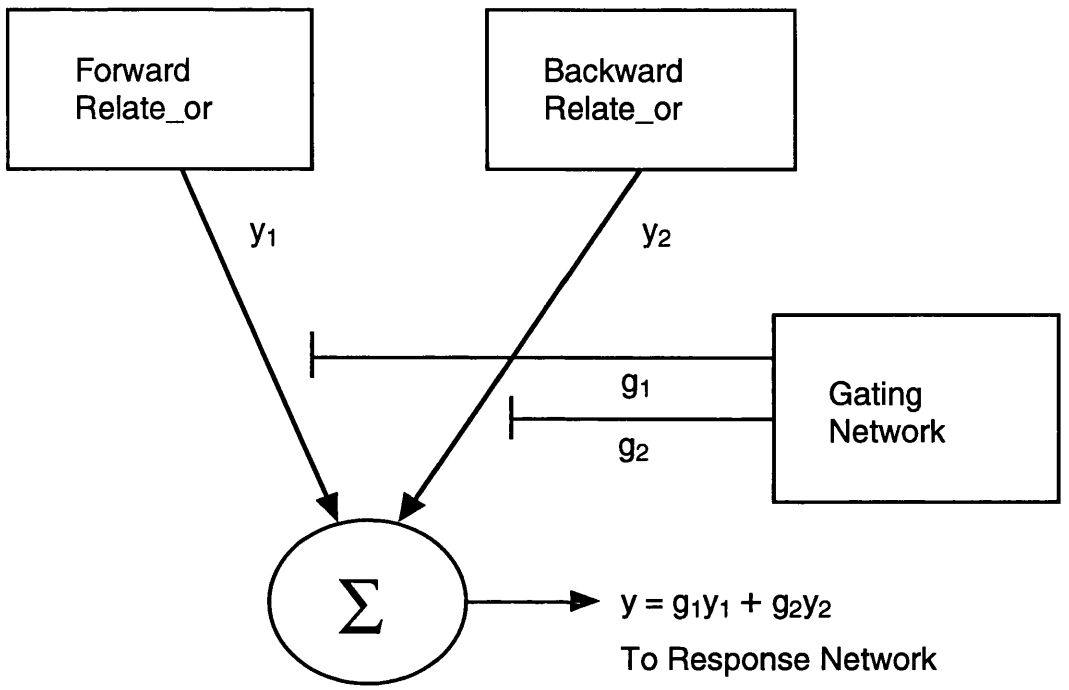


Figure 27. Forward and Backward Networks Connected Through a Gating Network. The networks compete to learn the training patterns or to send activation to the response network and the gating network mediates their competition. From Jacobs, Jordan and Barto (1991).

As the sequence is encountered in reverse order it is treated as a new sequence and a new SON network is developed. Since only a single response is required all networks must compete to provide it. They must develop inhibitory inter-links to ensure that only the response from one network reaches consciousness, otherwise responses would be lost due to confusion. Jacobs, Jordan and Barto's (1991) Modular Connectionist Architecture provides an elegant way of implementing this inter-linking of networks.

The idea behind the modular architecture is that the networks compete to learn the training patterns and the gating network mediates their competition. After training they compute different functions that are useful in different regions of the input space. The intuition behind the training function is that for any sequence,

presentation of the initial element causes activation in the subsequent elements. If there is no activation of the subsequent elements a network is chosen for training. All trained networks are labelled, with these labels being strengthened with use and decaying when left idle. In this way networks which are unused can be retrained. If one or more networks show activation as expected in downstream sequence elements, the network with the strongest activation is chosen for further training. The system works to reduce the sum of squared error between the output of the system for each sequence element, (y_i), and its desired output, (y_i*). This error, denoted (J_y), is summed over all sequence elements presented and given by:

$$J_y = \sum_{i=1}^{n} \frac{1}{2} \left(y_i^* - y_i \right)^T \left(y_i^* - y_i \right)$$

where:

J_y = summed squared error.
n = the number of sequence elements.
y_i = sequence element i.
y_i^* = desired output when y_i is input.

If one network comes closer than the others to producing the desired output, for the sequence, it is termed the "winner" and all others are termed "losers". The weights on the gating network are adjusted to make the output corresponding to the winning network increase toward (1) and the output corresponding to the losing networks decrease toward (0). Alternatively, if no network responds the gating network is adjusted to move all of its outputs toward some neutral value. To decide if a network is responding the performance of the entire system is compared to its previous performance. Where past performance is an exponentially weighted average of (J_y) over earlier time steps, this is given by:

$$\overline{J_y}(t) = \alpha J_y(t) + (1-\alpha)\overline{J_y}(t-1)$$

where

$\overline{J_y}(t)$ = weighted average of SS error at time t.
α = weighting term, such that $0 \leq \alpha \leq 1$.
$J_y(t)$ = summed squared error at time t.

They use binary values λ_{wta} (wta = "winner take all") and λ_{nt} (nt = "neutral") to indicate whether the system's performance has improved. Specifically:

IF $J_y(t) < \gamma \overline{J_y}(t-1)$ THEN $\lambda_{wta} = 1$ and $\lambda_{nt} = 0$

ELSE $\lambda_{wta} = 0$ and $\lambda_{nt} = 1$

where:

$J_y(t)$ = summed squared error at time t.
γ = weighting term.
$\overline{J_y}(t)$ = weighted average of SS error at time t.
λ_{wta} = indicates (=1) if performance has improved.
λ_{nt} = indicates (=1) if performance has remained same.

(γ) is a multiplicative factor that determines how much less the current error must be than the measure of past error in order for a network to be responding (i.e. network performance is significantly improved). If there is significant improvement ($\lambda_{wta} = 1$) the system must determine which network is responding closest to the desired output. To do this they define the error for network (n) to be the sum of the squared error between the network's output, (y_n), and the desired output, (y_n^*). Calculated by:

$$J_{y_n} = \frac{1}{2}\left(y_n^* - y_n\right)^T\left(y_n^* - y_n\right)$$

where:

J_{y_n} = error for network n.
y_n = the output of network n.
y_n^* = the output desired from network n.

The winning network is the network with the smallest error. If network (m) is the winner, then the desired value of the m^{th} output of the gating network, denoted (g_m*), is set to (1). Otherwise, if net (m) is a loser, (g_m*) is set to 0. If the architecture's performance has not significantly improved (λ_{nt} = 1), then the weights of the gating network are adjusted so all outputs move toward a neutral value. This value is ($^1/_N$), where N is the number of networks.

The gating network's error function, denoted J_G, is given by:

$$J_G = \lambda_{wta}\frac{1}{2}\sum\left(g_i^* - g_i\right)^2 + \lambda_{wta}\frac{1}{2}\left(1-\sum g_i^2\right)$$

$$+\lambda_{wta}\sum g_i\left(1-g_i\right) + \lambda_{nt}\frac{1}{2}\sum\left(\frac{1}{N} - g_i\right)^2$$

where:

J_G	=	error for the gating network.
λ_{wta}	=	indicator of improved performance.
g_i^*	=	desired output of gating function.
g_i	=	output of the gating function
λ_{nt}	=	indicator that performance has not improved.

The first three terms contribute to the error when the architecture's performance has significantly improved and the final term when it has not. In a change from the model described in Jacobs, Jordan and Barto (1991), error vectors for each network are not back propagated from the gating network to the individual networks, but the gating network does decide which network will be exposed to incoming data. The gating network does this by only opening the input connections to either the most responsive network or the network next in line to train. The gating network does this by using the values of g_1 and g_2, which are also used to determine which network should respond to a given input. The network that responds to the data also uses that data to train or if no network responds a new network is trained. The training scheme used for the SON networks is described in Chapter Nine.

2. Decade structure of Number Word Sequence

A second modification within the multiple sequence system is designed to simulate the decade structure of the number word sequence. This modification is the addition of further networks. The idea behind it is motivated by findings from data on acquisition of the number word sequence greater than ten. Fuson, et al., (1982) found that children made a large shift in the length of the number word sequence they could report when they understood that the number words for the tens digit followed a sequence much like the number words one to nine i.e.. tens, twenties, thirties, forties, fifties, etc. This finding along with the finding of Itsukushima, et al., (1990) that symbolic distance effects are found between numbers within a decade (i.e. 34 - 39) and between two decades (i.e. 20 - 50), but not between different numbers within a decade not containing the standard (i.e. 34 - 41 is not slower than 34 - 49). These finding indicate that the number word for the decades are stored in a separate SON network which is accessed only when they differ. Decisions within one decade could be made by the SON network for the number words One to Nine.

3. Alphabetic Sequence

A third modification to the network is for sequences which are not as fully elaborated as the number word sequence e.g. the alphabet. In relational judgement experiments carried out on this sequence, researchers have found that subjects perform what has already been referred to as run-through (Hamilton and Sanford, 1979). Sub-sequences are accessed to respond to questions about the relationships between letter words. The SON network as already discussed cannot account for this phenomenon. However, after some modifications it may (see figure 28). This figure contains a connection structure which may develop if a SON network was trained on sub-sequences only, rather than extensive work with individual sequence element pairs. It is likely that this type of training difference occurs between learning the alphabet and

learning the number word sequence. RT studies have found evidence of preferred entry points into the alphabet (Klahr, Chase and Lovelace, 1983). These would be at A and E in figure 28. The preferred entry points would be a by-product of training on sub-sequences. If the sequence was not routinely entered at B, C or D and individual element pairs were not routinely tested, the standard would not become associated with its corresponding element. Instead, if a sub-sequence is normally pointed to, it would become associated with the first element of that sub-sequence. For example, "Where is C in the alphabet ?", answer : "A B C, C comes after B." To test this hypothetical configuration of the SON network it would be necessary to find out if this is the type of training children receive for the alphabet. Once the type of training received is analysed one can intuit the type of SON network configuration that would result and test this configuration in the response network comparing the results to studies of RTs to alphabetic stimuli. This type of structure is developed further in Chapter Nine.

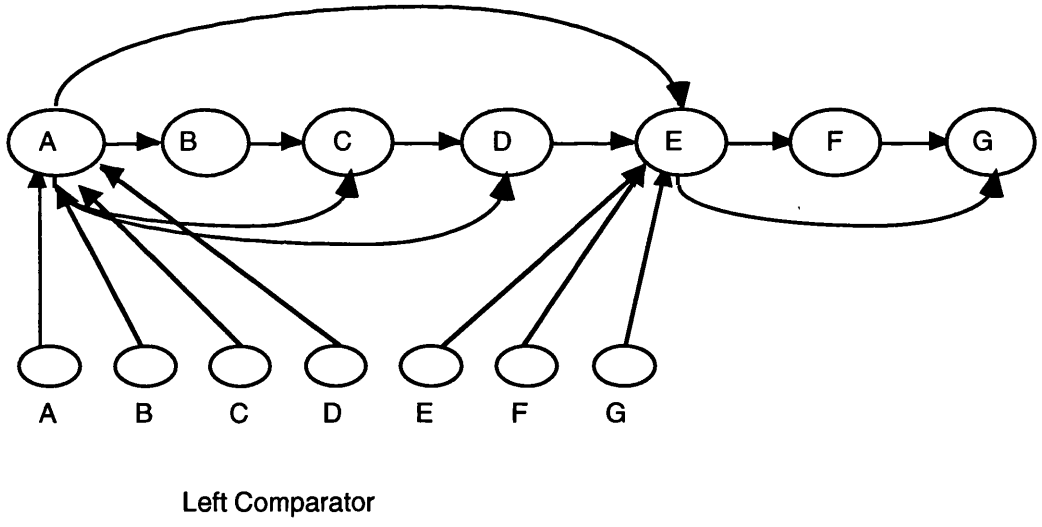


Figure 28. SON Designed to Explicate Structure of Alphabet. A SON network that has only trained on sub-sequences and not individual pairs of sequence elements is likely to develop a structure similar to this with only a few elements becoming fully interconnected. This is a hypothesised connection system. Connection systems which model people's knowledge about relationships between alphabetic stimuli will have to be developed through training of the network or RT simulations.

Chapter Five - Processing of Sequences

This chapter describes ways proposed for the processing of sequences of elements. Initially, the theories of sequence processing are discussed, then the implementations they have generated. These models are examined in the light of the three basic requirements for an implementation of long-term sequence learning, representation and relative order judgements. The requirements are (1) sequential learning beyond a simple 'next' operator, (2) incremental learning and (3) entry into the sequence at points other than the first element. Three implementations are found which have the potential to fulfil these requirements. Two are modified training techniques for the Simple Recurrent Network and the third is an extension of the SON. These three implementations are further investigated in Chapters Six and Seven.

A. Theories

Theories of sequence learning and representation can be divided into two classes. These are centred on paired-associate and serial models or based on the hierarchic encoding of sequences. These two types of theory have both been superseded by or encompassed in the implementations which are discussed in the next section.

1. Paired-associate and Serial Models

One of the first theories was proposed by Ebbinghaus (1913), it was termed the theory of pairwise associations. This was a chaining theory of serial recall, with each element attached to only the following element. Lashley (1951) pointed out that this theory could not account for sentence comprehension or skilled movements, because in both of these phenomena distant elements are related to each other too quickly to have passed through all intervening elements - a criticism that can be extended to relative order judgement

findings also. Therefore, this theory does not provide an explanation which is relevant to the current task.

Bewley (1972) reports that a variation on the basic chaining hypothesis is the cluster hypothesis of Horowitz and Izawa (1963), which states that a cluster of preceding items serves as the cue for each following item. According to this hypothesis, in the list ABCDEFGHI, ABC or BC might be the stimulus for D.

Further hypothesising was carried out by Young (1962) and Young, Patterson and Benson (1963), who held that serial position is the cue for recall of items in the middle of the list, with preceding items the cues at the two ends of the list. Ebenholtz (1963) suggested the opposite relation, i.e. serial position cues at the list ends with preceding items as cues in the middle of the list. Bewley also points out that another set of parameters thought to influence the relative importance of serial position and preceding items as cues to recall, are material and task variables. Battig, Brown and Schild (1964) suggested that serial position cues are important with lists of relatively low task difficulty, such as with short lists and meaningful, low similarity items. Similarly, associations between adjacent items are thought to occur in lists of high difficulty e.g. in long lists or with material of low meaningfulness and high similarity.

One test of these different theories is the transfer of learning from paired-associates to serial learning tasks (as mentioned above). Bewley reports, in contrast to Lewandowsky and Murdock (1989), that the general finding is a slight but significant positive transfer which supports the chaining hypotheses. However, in going to a paired-associate (PA) from a serial learning (SL) task it is generally found that no transfer occurs. When transfer is found it is restricted to early trials of PA. Thus, Stark (1968) found that increasing the level of SL learning increased the transfer to PA in later trials. However, data reported by Breckinridge and Dixon (1970), Horton and Turnage

(1970) and Young (1961) found no evidence for an increase in SL-to-PA transfer as a function of degree of SL learning.

To explore the importance of serial position as a cue to item recall, Bewley reports the findings of Young, et al. (1963), who found that when subjects learned two lists, with the second the reverse of the first, the serial position curve was flatter at the centre (indicating better performance) than when a list of different items was learned as list two. Since the positions of the items in list two were closer to the positions of the items in list one at the centre of list two for the experimental group, it seems that serial position is more important as a cue in the centre of the list. However, Bewley points out that these data can be explained in nonassociative terms by noting that the middle of the list has relatively more to gain from positive transfer that the list end. That is, the list ends will be learned rapidly even without positive transfer and will therefore be less benefited by such transfer than the middle of the list.

In conclusion, Bewley states that the conflicting evidence regarding the functional stimulus in serial learning can be explained by viewing the learning task as a problem in strategy selection rather than stimulus selection. The choice of strategy depends upon the mind set of subjects, and upon the nature of the list. The subjects' mind set depends on instructions and prior experience. The nature of the list can be defined either in terms of how the list is presented or in terms of relationships among list items. Presentation method also influences strategy. Relationships among items which are not dependent upon the presentation method are normative associations among items, conceptual relationships among items, and the existence of sequences within a list which can be described by a rule. The research which he reviewed seems to indicate that naïve (uninstructed) subjects may choose strategies in a particular order. He points out that Keppel (1964) and Saufly (1967) have shown that subjects prefer to use a serial position strategy unless it is made impossible by removal of serial position cues. When this happens

chaining is used. Hence, for Bewley (1966, 1969), subjects attempt to use serial position cues, but are unable to in the middle of the list due to a lack of distinctiveness, and must resort to chaining. This is essentially the view expressed by Ebenholtz.

Lewandowsky and Murdock (1989) report that further serial position theorising was carried out by Conrad (1965) and Broadbent (1958,1971), who postulated that each item was kept in a separate bin in memory, with each bin corresponding to a position in the list. They did not believe that any associations existed between the bins. Some errors were considered a consequence of information fading over time, with others arising when information in a bin was misinterpreted. Waugh and Norman (1965) explained that this theory did not account for errors due to interference, which are more common than errors due to decay. Murdock and vom Saal (1967) also noted that this theory cannot explain transpositions in the storage of information, which are also common.

A compromise between serial (as typified by paired associates theory) and relational models (i.e. the bin theory) was developed by Shiffrin and Cook (1978). They postulated a relational model in which elements are chained together, but with information about the item stored separately from its order (relational) information. Order information was represented through a chain of nodes, each connected to only the following node by a relational bond. Item information was attached to each node via an item bond. Thus, the knowledge about what an item is can be lost without loss of position information for adjacent nodes in the chain. Boundary nodes were assumed to precede and follow the first and last elements in the list. These boundary nodes were attached ad hoc to get bowing of the serial position curve, found in short-term memory experiments, where items in the first and last parts of the curve are recalled better than items in the middle. Boundary nodes also provided the ability to integrate the list into the continuous stream of memories.

Unlike most serial models, the Shiffrin and Cook model relied on a distributed memory representation, which allowed elements to be retrieved by a direct access mechanism avoiding serial memory scanning. Distributed representations also led to some redundancy. Thus, local damage would not lead to complete failure. Most importantly, distributed representation has been shown to make more appropriate predictions about retention of serial order information than localised storage models (Murdock, 1983).

2. Hierarchical Organisation

The hierarchical organisation of list elements can be seen as an extension of the cluster hypothesis of Horowitz and Izawa (1963), where items in the list are grouped together and these groups act as recall cues for the following groups.

Cohen, Ivry and Keele (1990) noted that "substantial evidence argues that humans often represent sequences by hierarchical structures". They say that in these representations a sequence is coded as a series of groups, with each group being more finely divided at a lower level. This type of representation is employed in the Klahr, Chase and Lovelace (1983) model of the alphabet outlined in Chapter One. As Hamilton (1978) noted, pause boundaries in alphabet recitation were good indicators of subjects' underlying grouping, so Cohen et al. report that groupings are also sometimes revealed by the pauses between subgroups of a motor sequence.

In an attempt to understand how these hierarchical codes develop, Cohen, et al. investigated subjects' ability to learn sequences under distraction. They found that structured sequences could be learned even if the subjects are not aware of the sequence. They contend that although subjects could not attend to individual elements in distraction they could attend to relations between elements. They believe that this result, combined with those of Nissen, et al. (1987), shows a dissociation between neural systems responsible for

structured sequence learning and those responsible for declarative memory.

In further experiments using unique (each element followed by a unique successor, i.e. a b c d), hybrid (most elements are followed by a unique successor, but sub-sequence are repeated within the sequence, i.e. a b c a b) and ambiguous (no unique successors, i.e. a b a c b c) sequences, Cohen, et al. found that unique and hybrid sequences could be learned while subjects were distracted, but that ambiguous sequences could not. They hypothesise that the ambiguous condition forces hierarchic coding and that such a code requires attention for its formulation. They point out that this result is consistent with the hypothesis that frequent repetition of events in differing contexts favours the establishment of a hierarchic codes and such coding requires attention. On the other hand, because structured sequences with at least some unique associations are learnable under distraction, there also appears to be a second kind of learning system that is based on sequential associations, and that this type of coding may not require attention for learning relations between successive events.

They propose a second theory that might explain the present results. It is based on the premise that all structured sequences, whether learned under distraction or not, are coded in hierarchic form. Parsing is based on recognition, not necessarily conscious, of passages that recur in a structure. The presence of a unique association provides a salient cue that the sequence will follow in the same order as before i.e. may serve as a sort of flag that defines the start of a structure. When a structure has only ambiguous associations then the parsing mechanism must note that a series of events occur in the same order on different occasions. They point out that this theory is supported by the outcome of a priori predictions, but it lacks direct test.

They relate that a critical issue concerns how the sequence becomes parsed and why the distraction task interferes with parsing. The

parsing, they explain, may depend on a short-term memory process that temporarily retains the preceding item[s] in a sequence. This short-term memory may provide a basis for recognition of sequence parts that occur in the same order on other occasions. Such recognition would constitute a parse of the structured sequence, because the recognition partition has a start and end that separate it from adjacent parts. The parse also allows a control code to be established for the subsection of the structure. They suggest that if this conjecture is correct, a possible reason that a secondary task interferes with learning the ambiguous structure is that distraction interferes with the short-term memory that is responsible for recognising recurring subsections. The system that recognises repetitions of identified structure parts need not involve explicit awareness. Removal of the distraction task allows learning of ambiguous structures. The subject does not necessarily become aware of the structures learned, but such awareness becomes more likely.

Thus, a sequence learning system's applicability would be enhanced by the ability to address the issues highlighted by subjects' learning of unique and hybrid, but not ambiguous, sequences under distraction.

Along the same general line Mueller and Overcast (1975) postulated that recall of sequences increases over trials partly because the subject's organisational units increase in size and integrity over trials. They point out that Bower, et al. (1969) found that increasing chunk size presented over trials led to greater recall than decreasing chunk size. However, Mueller and Overcast doubt the validity of Bower, et al.'s conclusions as they did not run the necessary neutral reference group. Mueller and Overcast performed a new series of experiments and found that differences in the way simultaneous presentation of items is accomplished (whether groupings are larger, smaller or constant size over trials) are not as important as the fact of simultaneous presentation per se.

They state further that single-item presentation did not lead to inferior recall compared to the grouped presentation conditions. Single-item presentation did not produce less subjective organisation than all of the grouped presentation conditions as if the groupings helped subdivide the list, but other processes under the single-item condition compensated for that advantage. They conclude overall, that apparently the subject's memory skills are sufficiently flexible that they can more or less overcome most variations in presentation method, unless perhaps the intrinsic level of organisation in the list is quite high.

Laughery and Spector (1972) related the findings of Winzenz (1970) who cites two possibilities for organising sequences into groups. The first was suggested by Mandler (1967). Items of a group are recoded into a single unit, and then these units are recoded into a higher-order unit (e.g. B-U-P stored as "bup"). These must be decoded at recall, passing down through the layers of the hierarchy. The second method was suggested by Neisser (1967). Reading and rehearsing a series produces a rhythmic pattern, and successive groups are attached to accented beats. This preserves information about each item while combining the series of items into a different organisational format. Input and internal representation map one to one. Thus, the memory trace need not be decoded to retrieve the elements of the group. Neisser called this regrouping.

Laughery and Spector's (1972) findings indicate that subjects did not recode. Subjects were not aware of recoding and error probabilities showed that improvements in recall occurred mostly within groups. They concluded that individual items are learned and the links between them established over repetitions, as opposed to the links between recoded units being established on succeeding repetitions.

They find that these results do not seem consistent with a strong emphasis on the idea that rhythm provides the basic structure for memory. However, superior performance on grouped noise sequences

does indicate rhythm as a factor. They explain that if rhythm operates in primary memory, the effect of grouping on the noise sequence makes sense, because these sequences are to some extent retrieved from primary memory. On the other hand, they point out, the assumption that rhythm affects secondary memory is contradicted by the results on the repeated sequences. They prefer to think of the effect of rhythm in secondary memory as providing the cues and the opportunity for other processes to take place that do affect performance. Operations such as recoding and rehearsal are examples of operations that rhythm would influence by providing cues to determine what to recode or rehearse as well as the time (pauses) to do it.

B. Implementations

In looking at the sequence learning networks available, one finds a considerable number and diversity. Some of these networks are designed to recognise a sequence that changes over time, while others are designed to predict the following element given any element. A third set may have neither of these as the specified goal but still relies on relative order information. A brief outline of some of these networks is found below.

1. Sequence Recognition Systems

All of these networks are designed to parse a moving input pattern into recognisable bits, or to figure out that they are seeing the same thing despite differing temporal characteristics. Most of the sequence learning networks are designed to interpret language. They take a string of words as input and produce a representation of their meaning. The major difference between the task of interest and the task performed by these networks is in the output. Many of the recognition networks cannot output a single input stimulus, but only a general interpretation of all the input combined. These networks are

not of particular interest as this thesis is concerned with the reproduction of single input stimuli.

One of the first recognition networks was developed by Gaudiano (1988). This is a network designed to recognise dynamic patterns based on limit cycles with Hebbian learning. A limit cycle is the falling of the nodes in the network into a cyclic, periodic firing pattern. Hebbian learning is the strengthening of connections between two active nodes, combined with weakening of connections between nodes when only one of the pair is active on a given time step.

Learning occurs in the Gaudiano network when the synaptic connections are modified by Hebbian learning while nodes in specific spatial locations are externally stimulated. After training the network is able to recognise a pattern with the same spatial characteristics, even if its temporal characteristics differ. The same nodes will fire, but the network will enter a new limit cycle. This allows the network to recognise the same word regardless of whether it was said fast or slow, as long as the duration of the word was long enough for the network to enter a limit cycle.

Gaudiano notes that if it were possible to differentiate between limit cycles quantitatively, they could be considered to represent different memory traces. Then the network would be considered to have recognised the stimulus when it attains the corresponding limit cycle. Since a system can fall into a limit cycle in tens of milliseconds, it would be fast enough to encode a string of phonemes in continuous speech. Gaudiano speculates as to the construction of a language comprehension network using limit cycle recognisers as building blocks. This network would be designed to recognise patterns as they move past its inputs.

A recognition system has also been developed by Tank and Hopfield (1987). They delay the responses of some of the input level units in the network so that units encountering different parts of the

temporal sequence will all respond at the same time. Then a higher level unit recognises this concentrated pattern of activation. Time delays can be designed such that the order of input is important or not depending on the task.

Using the idea of time delay, but extending it over the entire input layer, Stornetta, Hogg and Huberman (1988) describe a system that is part way between the recognition systems and the recurrent systems detailed below. In their network input, nodes have added amplitude (*a*) and decay (*d*) parameters. The decay parameter is equivalent to a non-modifiable recurrent link with weight (*d*), which is tied to the temporal characteristics of the problem to be learned. This overcomes the need to replicate portions of the network at different time steps, which recurrent networks must do, but it specifically determines the length of the temporal dependencies to be learned.

While tasks performed in these networks obviously must occur to process sequences, they are to some extent at a lower level than the processing of interest. To try to address this issue would increase the scale of this project considerably. Therefore, the focus of this chapter will be on networks which are intended to predict the next element of already parsed strings i.e. prediction as opposed to recognition networks.

2. Prediction Networks

The networks which are best suited to the task of interest to this thesis are those which are based around prediction of following elements. Prediction systems are those that given an input item produces the item that generally follows it. For most, the response of the network at a given point in time depends not only on the current input, but potentially all previous inputs (Mozer, 1993). Cleermans and McClelland (1991) note that most early models of sequence processing, of this type, have typically assumed that subjects somehow base their performance on an estimation of the conditional

probabilities characterising the transitions between sequence elements (e.g. Laming, 1969; Restle, 1970). These networks failed to show how subjects might come to represent or compute the conditional probabilities.

More recently, Mozer (1993) has pointed out that prediction involves two conceptually distinct components. The first is to construct a short-term memory that retains aspects of the input sequence relevant to making predictions. The second makes a prediction based on the short-term memory. He concludes that in a neural net framework the predictor will always be a feedforward component of the net, while the short-term memory will often have internal recurrent connections. The recurrent portion of the network uses activation recycled from previous patterns to encode the sequential dependencies in the input. The length of the dependencies encoded can be discovered by the network or manipulated by the experimenter.

One of the first recurrent networks was developed by Jordan (1986). In which the output activation was recycled to become part of the input for the next pattern. This input consisted of a 'plan' component and a 'context' component. The 'plan' component remained static across all input patterns associated with that plan. Thus, the instantiation of one plan should lead to a sequence of outputs being generated, much like a sequence of movements is employed with the goal of catching a ball. The 'context' portion of the input was the recycled output from the previous pattern. In this way the network learns to link each pattern with the pattern that follows it. Problems arise for this network when patterns are repeated within the sequence. In this situation the network cannot produce the appropriate following item.

Jennings and Keele (1990) propose two methods of parsing sequences within a Jordan network. In the first method they reset the context units at each run through the sequence, marking its beginning and end. They considered this to act much like a working memory system. In

the second, different plan values are assigned to each subpart of the sequence. When a sub-sequence is finished and a new one is to start, the activation on the plan units changes accordingly. This is thought to be closer to a hierarchic coding scheme.

Each of these methods had the same effect on sequence learning. The learning of ambiguous sequences was facilitated. Their findings are that the simulations replicated the empirical findings and suggest that imposing hierarchical structure on sequences with ambiguous associations significantly improves the model's ability to learn the sequence.

Elman (1990) noted that some problems change their nature when expressed as temporal events, that the time varying error signal can be used as a clue to temporal structure. He further pointed out that the representation of time (and memory) is highly task dependent. There is no separate "representation of time". To incorporate a representation of time into his network, Elman recycled the activation from the hidden layer as part of the input. In this way the previous state of the entire network (as opposed to only the previous pattern) is incorporated into the representation of the present element. This allows the network to generate longer-term sequence dependencies. This type of network has to come to be referred to as a Simple Recurrent Network (SRN).

Jordan and SRN networks were investigated by Servan-Schreiber, Cleermans and McClelland (1988). SRNs show an advantage because they use hidden layer representations, which Rumelhart, Hinton and Williams (1986) point out have already been partially encoded into features relevant to the task. As Servan-Schreiber, et al. note "..In these nets internal representations encode not only the prior event but also relevant aspects of the representation that was constructed in predicting the prior event from its predecessor".

Prediction machines are also used in language studies. Instead of abstracting a meaning for the stimuli input to the network, they are used to understand how the grammatical structure underlying a language might be built from experience with that language. The grammar can loosely be described as the rules governing which types of utterances make sense in the language. Servan-Schreiber et al., determined that SRNs were capable of differentiating between grammatical and non-grammatical strings from experience with a language.

They also note that the ability of the network to retain temporal dependencies is tied to the size of the hidden layer - when resources are scarce, dependencies are lost. If the network is to retain long range dependencies it may be necessary to have a large amount of redundancy in the hidden layer representations.

Cleermans and McClelland (1991) again examined the use of SRNs for sequence learning. They point out that as an "encoding" of the relevant feature of the input patterns, the hidden unit patterns, and thus the context layer patterns, can allow the network to learn to maintain production-relevant features of an entire sequence of events. They describe the properties that make SRNs attractive for sequence learning as: the network's ability to only develop sensitivity to temporal context if it is relevant in optimising performance on the current element of the sequence, and the network's ability to make minimal assumptions regarding processing resources. They point out that the second of these stems from the network's tendency to become sensitive to temporal context in a very gradual way, and its tendency to fail to discriminate between the successors of identical sub-sequences with disambiguating predecessors, when the embedded material is not itself dependent on the preceding information.

Cleermans and McClelland's investigations stem from the finding that the ability to learn sequential material under attentional distraction interacts with sequence complexity (see section 1b). They propose

extensions to the basic SRN based on a series of observations. Firstly, that it appears that a response that is executed remains primed for a number of subsequent trials. The second factor, which may be related, is that responses that are acceptable at trial t, but do not actually occur, remain primed at trial t+1. The third factor is a priming, not of a particular response, but of a particular sequential pairing of responses. This could be due to a rapidly decaying component of the increment to the connection weights mediating the associative activation of an element by its predecessor. They note that such "fast" weights have been proposed by a number of investigators (Hinton and Plant, 1987; McClelland and Rumelhart, 1985).

Cleermans and McClelland propose two changes to the SRN. Firstly, the pre-activation of a particular response based not only on activation coming from the net, but also on a decaying trace of the previous activation. Secondly, the changes imposed on the connection weights by the bp learning procedure are to have two components - a small, but effectively permanent change, and a slightly larger change which has a half-life of only a single time step.

Their results suggest that the model now captures key aspects of acquisition and, at almost every point, the best prediction of the human data is the simulation of the corresponding point in training. The ability of the network to simultaneously represent similarities and differences led them to refer to the model as an instantiation of a graded state machine. This is designed to emphasise the fact that, although there is no explicit representation of the hierarchical nature of the material, the model nevertheless develops internal representations that are shaded by previous elements of the sequence.

In relation to Cohen, et al.'s (1990)(see section 1b) theory about the possible differences in representations between unique and ambiguous sequences, Cleermans and McClelland note that in this model the representations of sequence elements that are uniquely associated with their successors are not different in kind from those elements

that can be followed by different successors as a function of their own predecessors. They found that by adding normally distributed random noise to the net input of each unit in the network, they were able to reproduce the pattern of results found by Cohen, et al. (1990). Therefore, a distinction between associative and hierarchical representations does not appear to be necessary to explain the interaction between sequence structure and attention.

Another type of recurrent network was designed by Plate (1992). He termed this a Holographic Recurrent Network. In this network pairs of vectors are associated by circular convolution, with multiple associations summed together. The sequence is decoded by convolving the resulting representation with the inverse of one of the encoding vectors. Plate notes that circular convolution is useful for representing hierarchical structure in that the circular convolution of two vectors is another vector of the same dimension, which can be used for further associations.

The particular method used by Plate for storing sequences using circular convolution is to associate elements of the sequence with points along a predetermined trajectory. Elements of the sequence and points (loci) on the trajectory are represented by n-dimensional vectors, with the loci being the successive convolutions of a single vector K (i.e. K^0, K^1, K^2, .. such that $K^{i+1} = K^i \oplus K$). Decoding is then done by repeatedly convolving the resulting representation with the inverse of the loci of the trajectory.

Plate notes that since the decoding of the sequence is noisy, the decoded vectors must be compared to all possible sequence elements in an error-correcting associative memory. Thus, with 512 element vectors and 1000 items in the error-correcting memory, five pairs of vectors can be stored with a one percent chance of an error in decoding. The scaling is nearly linear in n: with 1024 element vectors ten pairs can be stored with about a one percent chance of an error.

In contrast to the SRNs, but in common with the unexpanded Jordan networks, the target of this network is the generation of target sequences at the output units, with inputs that do not vary in time. One of the motivations for this work was to find recurrent networks with high generative capacity i.e. networks which, after training on just a few sequences, could generate many others without further modification of recurrent or output weights. New sequences are trained incrementally since the generation of each sequence involves an exclusive set of modifiable weights, as only one input unit is active for any sequence.

These prediction networks, as has been noted, are best suited to the task at hand. However, it is important to review a third set of networks which use relative order information in a different way in order to investigate if this new approach has anything to offer.

3. Other Networks

Some of the networks outlined in Chapter One, section D are also sequence production mechanisms. These include TODAM (Lewandowsky and Murdock, 1989), MVBP (Lacouture and Marley, 1991) and the Alphabet model (Klahr, Chase and Lovelace, 1983). The TODAM model involves the convolution of vectors representing sequence elements and can be extended to include hierarchical information. The MVBP model is an extension of a non-recurrent backpropagation network. And the Klahr, et al. model is a hierarchical model which does not include a learning mechanism.

Mareschal and Shultz (1993) investigated the use of cascade-correlation networks in the development of seriation. Cascade-correlation nets are those in which the activation in the network is averaged through time instead of each time step being treated as a separate entity. This is another extension of the backpropagation training algorithm. The network they developed is one which learns to sort.

The Mareschal and Shultz network begins with a minimal topology of an input layer fully connected to an output layer. Weights are trained using the quickprop algorithm (a minimal version of backpropagation), to minimise the difference between observed and desired activation across the output units. If the error is not reduced to within an acceptable criterion, several candidate hidden units are constructed and inserted into the network, with connections from all units in the network other than the outputs. The weights on the formed connections are trained so as to maximise the correlation between each of the candidate hidden unit's activations and the residual error at the outputs. Once the correlations reach asymptote, the hidden unit whose activations correlate best with the error is installed in the network with trainable connections to the outputs. Once installed, the weights leading to the new unit are frozen. The network then reverts to the error minimisation phase, but with the added power of a new hidden unit tuned to the residual error. If necessary, the cycle is repeated until success is achieved.

The Mareschal and Shultz model contains a component devoted to the processing of seriation information. This component receives information about the present state of the series, processes this information, and outputs a move. A move is defined as the identification of a sequence element and of the position to which that element should be moved. The move is carried out by auxiliary software that maintains the array. The seriation component is composed of two distinct modules, each processing the same input array but responding independently. Processing occurs in parallel, with the output of the element identification (*which*) module and the placement (*where*) module integrated in the output units. The modules are trained to move the smallest element that is out of order to the correct place. The *which* and *where* modules are each individual cascade-correlation networks.

The different type of sequencing network was developed by Houghton (1990). It is called the Competitive Queuing model [CQ]. The plan is to use aspects of the CQ models in connection with the SON model outlined in Chapters Three and Four to recite the sequence.

Houghton designed the CQ model based on the intuition that sequencing can be affected in iac models by the activation of a node representing, for example, a word, leading to the parallel activation of components of the word (phonemes in the case of this model). The sequential selection of successive elements in then made on the basis of their level of activation. Houghton began with two goals:

1. Defining a learning algorithm so that sequences can be taught: the resulting coding scheme is stable and sequence-sharing sub-elements do not interfere with each other.
2. Defining an effective sequencing mechanism so that recall of a sequence is genuinely a temporal phenomenon: the mechanism proposed avoids the use of position-specific encodings. (Houghton, 1990, p. 314).

The second of the above is the goal of interest, and therefore the aspects of the model designed to reach the first goal will only be touched on and not fully explained.

Along with these goals Houghton described several criteria which he felt must be met by the model. The criterion particularly relevant here was that the network should learn sequences through exposure to them followed by simple associative weight change rules based on activation levels.

In its simplest form the CQ model consists of a set of three layers of **on-centre, off-surround** nodes. In such a layer each unit inhibits and is inhibited by the other units in the layer (off-surround), and feeds back positively to itself (on-centre).

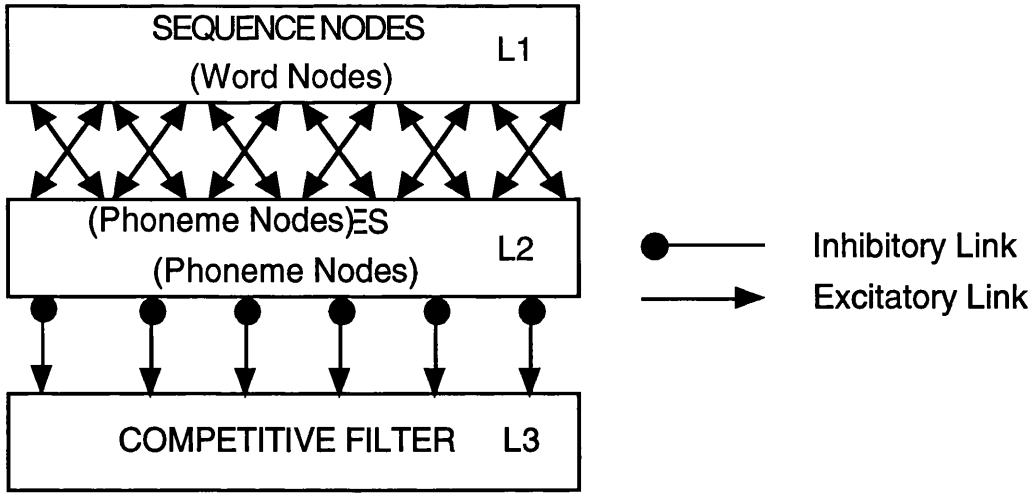


Figure 29. The Competitive Queue Model Described by Houghton (1990 As an extension to the SON the number of layers would remain the same, but L1 and L2 would be replaced. Figure taken from Houghton (1990).

The representations of nodes are defined by a local scheme whereby each node or unit represents some behavioural item at some linguistic level (for example, a phoneme or phonetic feature). Nodes stand for putative 'cell-assemblies' i.e. collections of excitatorily connected neurones whose firing is positively correlated. The lateral inhibition between assemblies (nodes) would be mediated by inhibitory inter-neurones, not by direct inhibitory connections. In these two respects the CQ model and the SON correspond directly.

The CQ architecture, as defined by Houghton, consists of three layers (L1, L2, L3). This is depicted in figure 29. The first two layers are specific to the word-phoneme task implemented by Houghton. The aspects of the model which are more general are the connections from L2 to L3 which are one-to-one and essentially copy activation from L2 to L3. L2 and L3 are structurally identical but L3 is winner-take-all i.e. the inhibition has been increased in L3, such that only one node remains active after activation has run its course. The feedback from L3 to L2 is one-to-one inhibitory, and once a node at L3 has won the competition it inhibits its corresponding node at L2. L3 thus acts as a **competitive filter.** All activation values in the model vary in the

range [-1,1], with an activation value of 0 representing background (spontaneous) levels of activation. Negative activation values are representative of below-background levels of activation and positive values represent excited states. Negative activations do not spread. Figure 30 shows the construction of the competitive filter.

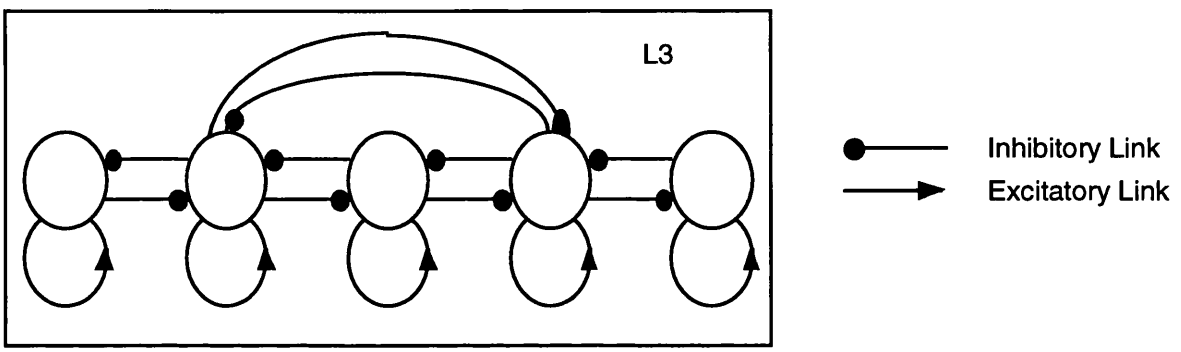


Figure 30. Configuration of the Competitive Filter in the CQ Mode
Excitatory and Inhibitory connections are balanced so that only one node is active after processing (winner-take-all). Figure taken from Houghton (1990).

This CQ design can be used to generate a representation of the sequence and recite that sequence from the representation. The L2 and L3 layers could also be used to take a sequence representation and produce a recitation of that sequence. The effects produced at recitation would depend on the representation of the sequence that had been developed.

4. Summary

The suitability of the predictor models has not been superseded by any of the other models outlined. Although the CQ model may also prove suitable. It therefore seems appropriate to examine the predictor models to determine if they are likely to offer a valuable model of sequence learning and production of the type outlined in this thesis. The two main networks presented are the Jordan network and the SRN. Of these the SRN seems better suited to the task as it allows more

complicated time-dependencies to develop. These time-dependencies are important in the construction of the comparison network, and therefore it would seem more appropriate to have them represented in the sequence learning and production network as well. The third network presented the Holographic Recurrent Network suffers from the same shortcoming as the Jordan network.

Based on this summary the following section will examine the possibility of generating an SRN to learn and produce a sequence of elements to model the results reported thus far in this thesis.

C. Predictor Network for the Present Problem

By outlining the requirements of a network to suit the task at hand, it becomes possible to create criteria on which the networks described can be judged. There are three basic requirements: (1) sequential learning beyond a simple 'next' operator, (2) incremental learning, (3) entry into the sequence at points other than the first element. Networks meeting the first two requirements will be investigated in Chapter Six, with those meeting the third examined in Chapter Seven.

The necessity of sequential learning beyond a simple 'next' operator is discussed in detail in Chapter One and above. Incremental learning is required because, besides being intuitively obvious that sequences can be learned a bit at a time, Fuson (1988), in her study of children's acquisition of the number series, found that the children can be at different levels of learning with different parts of the sequence at the same time. Along the same lines Hamilton (1979) found that subjects were able to learn novel sequences incrementally in sub-groups.

The arbitrary (but consistent) sequence entry points requirement, is discussed in Chapter One section A. As noted by Hamilton and Sanford (1978), subjects show preferences for the letters at which they start run-through. These letters have been found to correspond to the

beginnings of subgroups encountered by subjects while being taught the alphabet, plus a few intra-subgroup entry points. Fuson also found the same type of consistent entry points in the Breakable Chain level of number-word sequence learning (discussed further in Chapter Seven). The fact must also be addressed that with some amount of learning these entry points remain, but as more learning occurs the subject is able to enter at any element .

Section B concludes that the most appropriate of the networks outlined is the SRN as it is designed to develop temporal dependencies beyond the simple 'next' operator. The ability of this type of network to perform the tasks outlined above has already been investigated by Morrison (1990).

1. Morrison's Investigations

Morrison (1990) investigated the ability of a 15x10x5 Elman network to model the blocked learning of a sequence. The patterns she used were 5-bit binary patterns (i.e. a = 0 0 0 0 1, b = 0 0 0 1 0,.., e = 1 0 0 0 0, f = 1 0 0 0 1). The network, when presented with the entire sequence, learned to a *tss* value of 1.08 in 450 epochs, at which point all elements of the sequence were recognised (*tss* < 0.01), except one (*o*, *pss* = 1.004). When she tested entry into the sequence at an arbitrary point (i), the network was able to predict the next element (p ≈ i+1), but predictions from this point became worse i.e. p+2 was a worse approximation to i+3 than p+1 was an approximation to i+2.

The sequence was then divided into 5 blocks with a 3-bit change between the last pattern of block n and the first pattern of block n+1 (i.e. *f* = 1 0 0 0 1, *g* = 0 0 1 1 0). Again, the network was able to recognise all sequence elements after 450 training epochs. The network was able to predict the next element across a group boundary with the same degree of success as predictions within groups, but entry at any element other than the start element produced *pss* values greater than 1.0 for the following element. This large a *pss* value

indicates that the network would not be able to continue recitation of the sequence.

In her next simulation the input codes for all elements of a block were the same, while target outputs varied. The network required 1500 epochs to learn to a *tss* value of 1.93. It was capable of predicting elements across group boundaries (learning these mappings appeared to be responsible for the large number of epochs to train). Testing the network on arbitrary entry points still gave poor results as did tests of prediction of elements within blocks. Again, predictions became worse as n in p+n increased.

In Morrison's fourth simulation she set context unit inputs to zero between blocks in an attempt to keep the network from using this information. The network learned to *tss* = 1.0000 in 300 epochs and *tss* = 0.0842 in 650 epochs. All sequence elements were highly predictable with block-initial elements particularly high, but the networks' ability to continue the sequence once entered was low, even when initial entry was at the start of a block. When blocks were presented individually, the network showed difficulty i.e. it took 540 epochs to learn block one alone to *tss* = 0.0260. The network also showed catastrophic forgetting.

In a final simulation Morrison trained block one, then blocks one and two, followed by blocks one, two and three, etc. She found it took 3376 epochs to train to a *tss* of 1.0000, when the target *tss* was reduced to 2.0000. Learning time was reduced to 1205 epochs which was 755 more epochs then it took to train the entire sequence together.

The problems of recall and continuation of the sequence after recall in this network may have been due to the size of the hidden layer with respect to the size of the input patterns for the stimuli. As noted above, Servan-Schreiber, et. al. (1988) found that the more processing capacity there is in the network, the more it will use the time

dependency information. A second reason the network may have had difficulty in continuing the sequence is in the attempt to clear the context by setting context units to zero. By doing this they will have been treated as ten extra input units. The zero and one input values are designed to drive the activation of the hidden layer nodes to the asymptotes of the activation function (see Elman's training theories below). Elman has shown that replacing the context with 0.5 is a better option as it should keep the activation function within its dynamic portion. Thirdly, context activation was set to zero but momentum remained at (0.9) and mu at (0.5), both of which will cause the previous weight change and activation values to effect learning. Because of these terms, context is still partially active.

The simulations presented in Chapter Six will make an effort to re-investigate the Elman network's potential to model a sequence, with the effects in which we are interested, taking into account the findings and shortcomings of Morrison's experiments.

2. Incremental Learning and Catastrophic Forgetting

In Morrison's experiments she found that training patterns incrementally caused the network to overwrite the initial patterns with the subsequent ones. This phenomenon is known as "catastrophic forgetting". Although the following section describing the phenomenon and some proposed solutions centres on feed-forward BP networks and not SRNs it is considered that solutions which are valid for FFBP networks may be extended for use with SRNs.

2.1 Description

The consensus is that catastrophic forgetting occurs because the learning in the network is distributed throughout the network with the weight between each two nodes representing part of the "knowledge". When new items are learned and each weight is changed part of the previous "knowledge" is lost. The more weights

the new learning changes, the less of the previous "knowledge" is retained. This being the case when the material to be learned is represented in a localised way (with only a small set of the available input nodes set to one), the less likely it is that subsequent material will activate the same nodes and therefore the weights representing the stored knowledge are less likely to be disrupted.

Catastrophic forgetting is not always a negative phenomenon, some researchers have found it to be helpful in creating models of human memory. Rumelhart and McClelland (1986) used this feature of the network to model an aspect of children's language development. When children are learning to use the past tense of verbs they appear to pass through three stages. In stage one they use the past tense of a small number of verbs, many of which are irregular, in their proper form (i.e. went, came). During stage two they appear to learn the rule "in the past tense verbs require an -ed ending", and so they use verbs incorrectly that they were previously using correctly (i.e. "goed", "comed", "camed"). Stage three sees the return of correct usage of previously learned irregular verbs and correct use of the "-ed" ending. By training a FFBP network in stages which correspond to a child's early experience of language, it appears to learn in the same stages as the child. If the network's tendency to overwrite previous learning was not there the initially learned irregular verbs would never be incorrectly used.

Most researchers, however, have found catastrophic forgetting a major failing in SRN and BP networks. Ratcliff (1990) examined catastrophic forgetting with a view to "...investigating the limitations on the multi-layer connectionist architecture, and in so doing provide results that will help develop an understanding of the general constraints on the model". He performed several manipulations in pursuit of the goal of incremental learning, on both a small and a large network. The training patterns used were three initial vectors and one subsequent learning vector for the small

network, and a group of four vectors initially and eight individual vectors subsequently on the large network.

The first manipulation allowed all weights in the networks to change with no repetition of old material during training of new. The result was a loss of ability to reproduce old material. In the second manipulation, only weights which were small after initial training, indicating that they may be unimportant to the memory for this material, were allowed to change during subsequent learning. This also resulted in failure to retain. It appeared that although the weights were small, they were vital to the distributed representation. Next, new hidden units were added to the network as subsequent training took place. All weights were allowed to change during subsequent training. This configuration resulted in better retention of previously learned material, but still catastrophic forgetting occurred. If only the weights to the new hidden units were allowed to change during subsequent learning, perhaps catastrophic forgetting would not occur. While this resulted in better reproduction of initial items, it also resulted in poorer recall of the subsequently trained items. It appeared to be a trade-off. The weights which could not update could not be overwritten but also could not be used to generate representations of new material.

Ratcliff's investigations found that the more items subsequently trained, the larger the decline in ability to recall, regardless of the changes made to the fully distributed network. He also found that items learned in groups were more resistant to forgetting than items learned individually. McCloskey and Cohen (quoted in Ratcliff, 1990) obtained similar results for two different tasks - learning arithmetic facts and the Barnes and Underwood A-B, A-C interference paradigm.

2.2 Semi-Distributed Representations

French (1991) detailed two networks which circumvented the catastrophic forgetting problem. These were unlike Ratcliff's in that they had semi-distributed representations as opposed to fully-distributed representations.

The first was a Sparse Distributed Memory (hereafter SDM). SDM networks were made of ten million, one thousand bit memory locations. Each memory location contained a "counter". The system learned vectors corresponding to one thousand bit words. When a word was being learned the system selected all memory locations within a Hamming distance of four hundred and fifty places (approximately one thousand locations). At each location, if the corresponding bit of the one thousand bit word to be learned was one, the "counter" was incremented; if the bit was zero the "counter" was decremented. SDM networks were capable of learning new words without disrupting those already stored, as long as there was not too much overlap in their representations. When the memory was full, (approximately ten thousand words), interference began and no new words could be stored without disrupting old ones.

The second of the semi-distributed networks French (1991) outlined, was Kruschke's ALCOVE network. This is a three-layer network, but in contrast to those outlined earlier, the hidden units were not fully interconnected. Instead, the activation of each hidden unit was inversely exponentially proportional to its distance from the stimulus position. Thus, if only positions four, five and six in the input vector were active (set to one) then position five in the hidden layer would be highly active, and position fifteen would be less active. This is referred to mathematically as the hidden layer "covering" the input-layer, and neurophysiologically as each hidden layer unit having a localised receptive field in the input layer. In this way patterns which had activation in disparate areas of the input layer would not interfere with or overwrite each other. As receptive field size increases and the network becomes more

distributed, more interference is encountered. These networks did not exhibit catastrophic forgetting to the extent Ratcliff found because they were not fully distributed.

Lack of distribution can create difficulties because a network's ability to generalise lies in the distribution of representations throughout it. The more distribution, the greater the distance allowed between vectors that can be classed as similar or the same. This does not appear to be a desirable quality until one examines the ability of human beings to recognise a "table" as both a three-legged object topped with a hexagonal piece of Plexiglas, and a single-legged object topped with a round piece of Formica. While it is obvious that these objects share some qualities and therefore their representations will in some ways overlap, it is assuming unavailable knowledge to build networks whose success depends upon recognising some similarities and ignoring others. Therefore, the most promising path forward is to attempt to overcome catastrophic forgetting while retaining as much of the distribution as possible.

2.3 "Sharpening" In FFBP Networks

To this end French (1991) has proposed modifications to the standard FFBP network much the way Ratcliff did, but has met with greater success. Originally, French and Jones (quoted in French, 1991) proposed a method of differentially modifying the learning rates to connections in the network. This method, however, failed to scale up to larger networks. The ability of any solution to be successful on large networks is an important one. The human brain contains approximately one-billion neurones. With 0.01% of the brain's capacity equal to ten million neurones, networks containing three hundred neurones seem unlikely, those containing any less seem ludicrous.

French has now proposed a modification which has met with the most success to date. He describes a system which is still fully-

distributed, but in which hidden layer activation is concentrated in a few highly active units. When the activation associated with an input pattern is "sharpened" in this way it is thought less likely to interfere with that of other patterns, provided the patterns do not sharpen the same hidden layer nodes. This system works because the change in the weights of the network is dependent on the activations of the nodes. If a node is not highly active the weights leading to and from it are not greatly changed.

French's idea was to create a system which determined the most active hidden layer nodes and quickly set the others as close to zero as possible. This is accomplished by determining the number of nodes to sharpen and augmenting their activation by the value called the "sharpening factor". The higher the sharpening factor the higher the nodes' activation. All other hidden unit activations are decremented by their activation multiplied by the same sharpening factor.

By keeping the learning rate low and the sharpening factor relatively high, little damage is done to weights used in previous learning before nodes to be sharpened for subsequent learning are chosen. A high momentum value helps ensure that weights which are increasing continue to do so while not significantly affecting weights which are changing little. When French used this system he found that although the initial material was not well recalled immediately after the subsequent patterns were trained, it was significantly easier for the network to retrain the earlier patterns.

French found that the amount of memory refresh varied directly with the amount of activation overlap in the initial and subsequent patterns representations. This modification does reduce the system's ability to generalise, but it does not make explicit assumptions as to where in the representation similarities must lie for items to be classed together. French was unable to state with certainty the number of nodes to sharpen in the hidden layer. He

suggested k, where k is the smallest integer such that nCk is greater than the number of inputs or $\log_n$ if the number of inputs is unknown.

Although it is an advance on catastrophic forgetting for the network to retrain previously learned patterns in fewer epochs than required for original training, this does not constitute true incremental learning of the type desired in this thesis.

2.4 Sharkey and Sharkey (1993)

Despite the apparent success of French's sharpening it contains a major flaw. If a "sharpening" network is designed with the number of hidden layer nodes equivalent to the number of different classes of input pattern, Sharkey and Sharkey (1993) explains that it will orthogonalise the different classes of input pattern across the hidden layer. This causes all input patterns which are associated with the same output pattern to develop the same hidden layer representation. Thus, the training in the network is all being done between the input and hidden layers. With the hidden to output layer connections acting as a look-up table. This type of learning (look-up tables) has been long discredited as the number of experiences would quickly overload the capacity of the brain.

This type of learning ensures that once the network has been trained on the basis vectors for the input pattern set (i.e. the set of vectors from which all of the input pattern can be derived) the network is able to correctly classify all of the input patterns. However, this leads to a second problem. The network is unable to distinguish between input patterns on which it has and has not been trained. This is the problem of lack of discrimination.

Due to these shortcomings this method of overcoming catastrophic forgetting will not be pursued further. Instead, two other methods will be proposed and examined.

3. Pretraining (McRae and Hetherington, 1993)

Greater success in the pursuit of incremental learning has been achieved by McRae and Hetherington (1993). Their work toward incremental learning in backpropagation networks focuses on pretraining the network. Their rationale is that subjects do not come to a task totally naïve and unable to create associations between the task in the study and previous knowledge. Even if one were able to create a totally novel and unrelated task, it would not reflect "normal" processing capacities.

To investigate the utility of pretraining the network, they used an encoder network to learn two lists of CVC (Consonant Vowel Consonant trigrams) incrementally. They found that pretraining the network with Wickel orthographic representations of monosyllabic words or orthographic stimulus-response pairs, significantly reduced the number of hidden layer nodes used to encode new patterns over a naïve network. This reduction of the hidden layer overlap between patterns reduced the interference between the first and second lists learned.

They then attempted a serial list learning task. Items were learned and recalled in order. Hidden layer overlap was again reduced, but they found the pretrained network produced more interference for similar items over non-similar items, then did the naïve network. This indicates that the network should still be able to generalise. The pretrained network again demonstrated less interference between the two lists then the naïve network and even less then humans do on such tasks.

The third task they simulated was a paired-associated task, were two lists of stimulus-response pairs were learned. This task required an increase in the size of the hidden layer. The reduction in hidden layer overlap remained as did the decrease in interference between lists learned by the pretrained network. This interference may again be

lower than that found in humans. Overall, they have shown that for these tasks, pretraining the network allows incremental learning to take place.

The networks McRae and Hetherington used are not sufficient for the purposes of this thesis because they did not use each item to predict its successor. The serial task they used was simply to look at the patterns as trained and recalled in a specific order, not to look at the ability to produce the next item in the list. In Chapter Six their ideas are adapted for use with an SRN.

4. Increasing Recurrence (Elman, 1993)

Some recent work by Elman may also succeed in extending the SRN to allow incremental learning of the type desired. In Elman's (1993) investigation, he began by examining learning in the network. He explained that there are three limits or constraints on the ability of any backpropagation network (including the SRN) to learn.

Firstly, these networks rely on the degree of representation of their data sets. If a small sample is drawn from the set, a network may fail to make generalisations that apply to the larger set. This is a problem most likely to occur early in learning i.e. when sample sizes are small.

Secondly, these networks are at their most sensitive in early learning because the closer each node's input is to zero, the more readily changes in input are translated to changes in weight. This is due to the inclusion of the first derivative (or slope) of the activation function in the weight change equation. As demonstrated in figure 31 the activation functions slope is steepest near inputs of zero and asymptotes as input approaches either ∞ or $-\infty$.

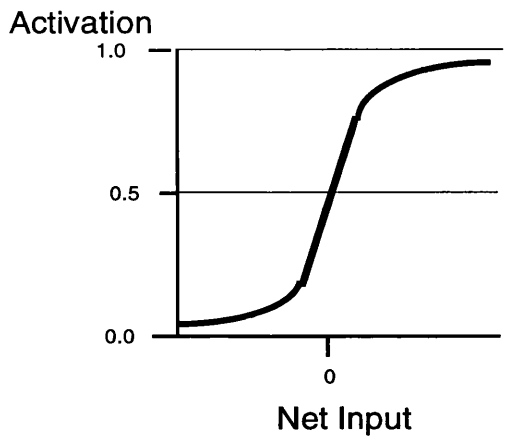


Figure 31. Sigmoid Activation Function used in Most Backpropagati Networks. The slope of the curve (and hence its first derivative) is largest when net input to the unit is 0 and flattens out as net input approaches ∞ or -∞.

Examining the first and second limitations together, it is apparent that the network is making the largest modifications at a time when its data is least likely to be representative.

Thirdly, the gradient descent learning method does not allow the network to make radical alterations in hypothesis. In these networks, solution hypotheses are represented by the weights of the connections between units. The network makes changes to these weights in response to the difference between the solution produced and the desired solution in such a way that the solution produced becomes closer to the desired solution in every time step i.e. the error in the network is always reduced. In mathematical terms, the error in the network descends the gradient of the surface representing all possible error values.

One by-product of this is that the network can become trapped in partial solutions (local minima). These partial solutions reduce the error in the network (the difference between the produced solution and the desired solution), but they do not fully solve the problem i.e. the network still produces some wrong answers. To move away from this partial solution, the network would have to rearrange its

weights (move to a new hypothesis). However, the network is restricted to making incremental weight changes, which means that it could not go to a completely new arrangement in one time step. The time steps between the present arrangement and the new arrangement would be characterised by an increase in the error in the network, but the network can only allow the error to decrease, so the network cannot rearrange - it is stuck. Symbolic and process models do not have the limitation of incremental changes to hypotheses.

The third limitation combined with the second limitation implies that since weight changes decrease in size as learning progresses, the problem of incremental changes to the hypothesis increases as learning progresses. The more the network learns, the less likely it is to escape a local minimum.

Elman proposes two methods of overcoming these limitations: providing better initial data, or worse initial data. Better initial data may allow the network to learn the correct generalisations while it is making its largest weight changes. The proposition is that in some cases the entire data set is necessary for the network to learn the correct generalisations (Harris, 1991 [no relation]) and remain free of local minima (Elman, 1993). This could also be achieved by training on the basis vectors for the set of input patterns, as pointed out by Sharkey and Sharkey (1993).

Elman also discusses another type of 'better initial data' solution. In this solution the network effectively 'chooses' initial data which allows it to generate a basic structure on which the remainder of the data set can build. To do this the network is forced to function with limited reference to previous events. The recurrent feedback is eliminated randomly after every third or fourth pattern (context units set to 0.5). As training progresses, recurrence is increased through five phases, after which the network is fully recurrent. This has the same effect as choosing the first training examples from the simplest in the data set and gradually increasing complexity as

training progresses (in Elman's task the more complex events were characterised by longer range time dependencies). Elman sees this type of mechanism as mirrored by the human infant, whose processing limitations may be necessary for basic concepts to be learned and then built upon.

The second method of overcoming this problem is to present the network with worse initial data. Worse initial data may keep the network in a state of flux until the data set is more representative. This can be achieved by adding random (Gaussian) noise to the initial training sets.

The task of interest is one that becomes more difficult as the range of dependencies decreases. Initially these sequences can be produced only initial element to end, but as training progresses they can be entered at any point and continued. Thus, the simulations in Chapter Six take the form of presenting the entire data set and gradually reducing it, or starting with a fully recurrent network and gradually reducing the recurrence. The interaction between these solutions and incremental learning is also examined .

5. Summary

This section has examined the type of network necessary to model the task of learning and producing a series of elements in the way that people are able to learn and produce sequences that are to be, or have been, stored in their memory for a protracted period of time. Three basic aspects of this task were outlined as requirements for such a network. Firstly, relations between elements should be more complex than a 'next' operator. Secondly, the series should be learned incrementally. And finally, the end result should allow entry into the sequence at points other than the start element.

Three possible models have been outlined. The first two are modified training techniques for the SRN. These networks are appealing in that

their structure allows the development of sequence dependencies beyond those of the previous element, thus fulfilling criterion one. The necessity of modified training techniques stems from the network's inability to learn in stages, thus leaving criterion two unfulfilled. These training techniques are believed to allow the network to overcome this difficulty, a claim which will be examined in the next chapter. If either or both techniques prove successful, they will be investigated further to determine the resultant network(s) ability to fulfil criterion three.

The third possible model is an extension to the SON model which is described in Chapters Three and Four. This model fulfils criterion one by the explicit postulation of connections beyond those between adjacent elements. The model's ability to fulfil criterion two is also in its basic design. As new elements to the sequence are encountered new nodes are allocated to represent them. As this does not involve previously represented elements, their representations are not lost. The only remaining difficulty for criterion two is extending the model to produce the elements of the sequence in order. This extension is examined in Chapter Seven. The ability of the network to learn the sequence and enter it at points other than the start element (criterion three) are explored in Chapter Nine.

Chapter Six - Sequence Simulations

This chapter examines the possibility of a network dedicated to learning the sequence being used to augment the SON. The last chapter indicated a reasonable choice for such a network is the SRN. The SRN structure is taken as the architecture of choice for much of the research into sequence learning, because the hidden layer patterns are partially encoded into features relevant to the task (Servan-Schreiber, Cleermans and McClelland, 1988 - discussed in Chapter Five). One problem of this type of network is that it is incapable of incremental learning because it suffers from catastrophic forgetting. Thus, two methods designed to overcome the problem of catastrophic forgetting in the SRN structure are examined. Neither of the two methods is found to overcome the problem to the extent necessary to model the task at hand. It is therefore necessary to investigate the third option of an augmented SON which was outlined in Chapter Five.

A. Network Implementation for Methods One and Two

The underlying network for training methods one and two was an SRN implemented in the McClelland-Rumelhart simulation package. It contained 16 input units (including 8 context units), 8 hidden layer units, and 8 output units (16x8x8). The learning rate was set at 0.5, with a momentum of 0.0. For learning in which context was irrelevant, context units' activation values were set to 0.5 instead of copying back the activation from the hidden layer, to encourage the network to ignore their input. This technique was also used by Elman (1990).

Training Patterns

The first two methods are tested using the same training patterns, although these patterns are organised differently across the two methods. The sequence elements are represented with arbitrary codes, as the interest lies simply in the network's ability to overcome catastrophic forgetting. More complicated encodings, such as transcription of the phonemes coding for voiced/unvoiced

etc., would only needlessly complicate the network and can be considered to have occurred earlier in processing.

To generate a sufficient amount of output, twenty-six patterns were used. For ease of identification these patterns were given names roughly approximating the letters of the alphabet. Each pattern was eight units long and contained two active units. This encoding allowed each pattern to contain the same number of active units, thus conferring no training advantage to later patterns, which generally contain more active elements.

To check incremental learning the patterns were divided into four subgroups, based roughly on the experimenter's alphabetic grouping (*aee -gee*, *haitch -pea*, *que -vee*, *dbl-u -zee*). The patterns may also have been grouped by the network because of the coding structure used. The first seven vectors (*aee - gee*) had their two active units adjacent to each other, followed by a set of six vectors (*haitch - eem*) with active units one unit apart. Then came a set of five vectors (*een - are*), with their active units two units apart, a set of four (*ees - vee*), with active units three units apart, three (*dbl-u*, *eex*, *why*), with active units four units apart and one (*zee*) with active units five units apart. This arrangement may have afforded the network a training advantage, in that most patterns differed from their immediate successor by only one active unit. The set arrangement, as outlined, did not correspond to the group arrangement.

B. Method One - Hetherington and McRae (1993)

For this method of overcoming catastrophic forgetting, the network is pretrained, so should be building on a knowledge-base which allows new training to occur without overwriting previously learned material. For the purpose of learning a sequence of information the Hetherington-McRae method has been adapted to use with a recurrent network. This adaptation entails pretraining the network on an auto-associative version of the pattern set, but removing the recurrence. Without recurrence the patterns should be learned as individual entities and not as a list. This pretraining is

considered to correspond to learning the individual elements of the sequence in isolation. For example, being taught that the syllable "ae" represents the letter A, without any reference to its relationship with B.

1. Procedure

The pretraining patterns employed were auto-associations of the training patterns, with context units set to 0.5. A list of the patterns in the first two subgroups is in table 12, with a full listing in Appendix 1. On the recurrent trials an additional *start* pattern was added for the convenience of the pattern name displayed being that of the input pattern. The *start* pattern was not of the same form as the other patterns, it consisted of four active units. For a full list of recurrent training files see Appendix 1.

Pattern Name	**Subgroup 1 Pattern**	**Pattern Name**	**Subgroup 2 Pattern**
start	1 1 1 1 0 0 0 0	haitch	1 0 1 0 0 0 0 0
aee	1 1 0 0 0 0 0 0	aie	0 1 0 1 0 0 0 0
bee	0 1 1 0 0 0 0 0	jay	0 0 1 0 1 0 0 0
cee	0 0 1 1 0 0 0 0	kay	0 0 0 1 0 1 0 0
dee	0 0 0 1 1 0 0 0	eel	0 0 0 0 1 0 1 0
eee	0 0 0 0 1 1 0 0	eem	0 0 0 0 0 1 0 1
eef	0 0 0 0 0 1 1 0	een	1 0 0 1 0 0 0 0
gee	0 0 0 0 0 0 1 1	oow	0 1 0 0 1 0 0 0
		pea	0 0 1 0 0 1 0 0

Table 12. Patterns for First Two Subgroups Used in Simulations. Including the start pattern. The patterns are arranged in sets, with most patterns overlapping one unit with their immediate successor. The set structure does not necessarily correspond to the group structure. Patterns were given names reminiscent of the alphabet for ease of identification.

2. Preliminary Tests

2.1 Design

Each pattern file was trained individually to a *tss* of 1.0000, in order to assess the ability of the network to learn the information the file contained. The value of 1.0000 was chosen as representative of all patterns in the file trained to the extent that the most active units in the network output correspond to the active units in the training pattern. This was necessary to compare with the pretrained network to determine if the pretraining gave the network any advantage.

2.2 Results

Table 13 shows the average number of epochs required to train each pattern file over four trials. On some trials the files fell into local minima, (*tss* values above 1.0000 after 1000 epochs), these trials are excluded from the analysis. Patterns for which local minimum trials were found are marked in the table with an (*).

The *tss* varies across files because of differences in the number of patterns the files contained. The *tss* being the summed error values across all patterns in the file, it increases with the number of patterns in the file.

The attempt at training the four subgroup file incrementally without pretraining, showed that the network encountered a local minimum on two of the four training runs. The file containing the first and second subgroups trained normally, but the third file had not trained to a *tss* below 1.0000 after 1000 epochs, and the fourth file again had no difficulty. The runs which did not encounter the local minima are reported in the table, but marked by an (*) to indicate the result is not as straightforward as it appears.

Pattern File	Epochs
Pretraining	97
Sub One	107
Sub Two	121
Sub Three	102
Sub Four	61
Entire File	318*
Subs One & Two & Three & Four	215*

Table 13. Average Number of Epochs for Test Files to Train. Preliminary investigation of the average number of epochs (over four trials) required to train the files used in the simulations, on a naïve network. Pretraining is the file containing the auto-associative pretraining patterns. The Entire file contained all twenty-seven patterns. All files were trained to a *tss* of 1.0000. Names linked by an & indicates the files were trained incrementally. * indicates that on some trials the network became trapped in a local minimum - these trials are not included in the average.

3. Tests

3.1 Test One

Design

The first test is of the validity of the assumption that network pretraining would lower the number of hidden units involved in encoding the target patterns. A naïve network and a pretrained network were trained on the first subgroup of the data set. The first subgroup was chosen as this group trained to the criterion value on both networks. Each network was trained four times and the activation values across the hidden layer for each pattern were recorded.

Results

Figure 32 shows a histogram of the percentage of the hidden layer units within each activation band. It is apparent that the pre-trained network has more nodes with activation values between 0

and 10 and between 91 and 100. An increase which is reflected by a decrease in nodes displaying activation values from 11 to 90 (with a small exception at activation values 31 - 40). This increase and reflective decrease may indicate that the network is polarising the hidden layer units as McRae and Hetherington predict.

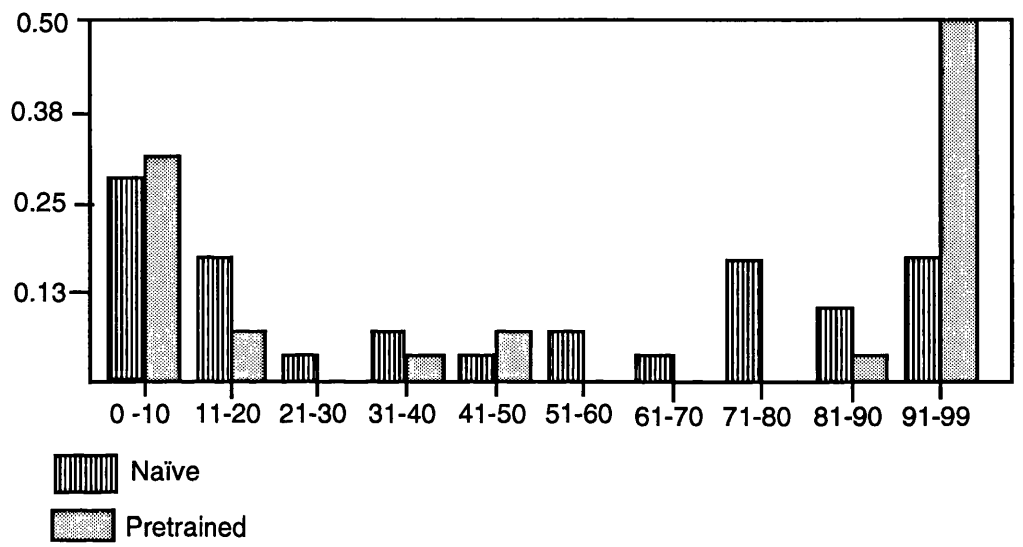


Figure 32. Activation Values Across Hidden Layer Units. The percentage of hidden layer units displaying activation within each band for four naïve and four pretrained networks after training subgroup one of the pattern file (to tss < 1.0). The pretrained networks have more nodes within bands 0 to 10 and 91 to 99, and due to a small inconsistency at 31 to 40. The naïve networks have more nodes in all other bands.

3.2 Test Two

Design

Initially, it is also important to determine if the pretraining gives the network a simple advantage over a naïve network. To examine this a network was pretrained (to a *tss* < 1.0000) and then trained on the file containing the entire data set (also to a *tss* < 1.0000). The entire file had become trapped in local minima on fifteen preliminary tests before four successful trials were completed on a naïve network.

Results

After eight trials on a network with experience of the auto-associative patterns, no successful trials had been completed. This means that after 1000 epochs the *tss* for the entire files had not fallen below 1.0000 and appeared unlikely to do so. This represents an increase in the number of local minima encountered over the naïve networks. This indicates that far from giving the network an advantage in learning the new material, the pretraining actually hampers subsequent learning of the material.

3.3 Test Three

Design

The third training regime used the approximation of McRae and Hetherington's training scheme outlined above. The auto-associative pretraining was followed by the incremental training of the four recurrent subgroup files, to determine if pretraining allowed incremental learning to occur. After training on the four subgroups, the network was tested on a file containing the entire pattern file with separations between the subgroups, to determine if the network had learned the sequence.

Results

Four training trials were completed (N_1, N_2, N_3, N_4) the pretraining file required 101, 93, 98 and 96 epochs to train to the required *tss*. After the initial training attempts were made to train the network to a *tss* < 1.0000 on the four subgroup files. On one of the trials the networks encountered a local minima while training the first sub-group file. The other three networks reached the target *tss* after 179 (N_1), 84 (N_2), and 65 (N_3) training epochs on the first subgroup. Three of the networks became trapped in local minima while training the second subgroup file. Only N_2 trained the second file, taking 141 epochs to do so. The same result occurred when the file containing the third subgroup was trained. Again, only N_2 learned the file to a *tss* < 1.0000, doing so in 103 epochs. The network which succeeded in learning the final subgroup file was not N_2. N_4 recovered from the local minimum, encountered in training the third subgroup file, allowing it to learn the fourth in 64 epochs. All other networks encountered local minima in training this file.

However, in N2 the tss for the fourth file after 2000 epochs was 1.0061 indicating that the file may have eventually trained.

As N_2 was the only network which consistently avoided falling into a local minimum, it was examined to determine if incremental learning had occurred. Table 14 shows the *tss* values for the four subgroup files and the entire test file of the pretrained network compared with those of naïve network which succeeded in training all four files. This comparison shows that the naïve network required fewer epochs to train all files except the one containing subgroup three. This result mirrors the results of the first tests, pretraining does not appear to aid in the subsequent learning when used in this way.

A second comparison to determine if the pretraining enables the network to recall the previously learned patterns and therefore overcome catastrophic forgetting, is to test the network on the entire pattern file. In this comparison the pretrained network produced a *tss* of 22.8335 while the naïve network produced a *tss* of 16.0224. This result again indicates that pretraining does not allow the type of incremental learning in which we are interested.

	Trial One	
	Pretrained	Naïve
Sub 1	179	115
Sub 2	141	60
Sub 3	103	425
Sub 4	2000+	12
Total Epochs	2516	612

Table 14. Number of Training Epochs for Test Networks. The number of training epochs for the pretrained and naïve networks, which succeeded in training all four sub-sequence files (although the pretrained network experienced difficulty with subgroup four). Note that the naïve network was able to learn in fewer epochs.

4. Discussion

Although it does appear that the activation in the hidden layer units has gravitated to the extremes of between 0 and 10 and 90 and 99, pretraining did not allow the network to overcome catastrophic forgetting, nor did it create any training advantages for the network. In fact, training was less successful in pretrained networks than it had been in naïve networks. The measures of success used were the number of epochs to train or the *tss* if training required more than 1000 epochs.

It is possible that the failure of this method was due to the network architecture used. McRae and Hetherington's work was done on a standard backpropagation network (without recurrence). It may be that the addition of recurrence to the network after pre-training causes the pretraining to be counter-productive, as opposed to a failure of the pretraining system in general. It is also possible that increasing the size of the hidden layer would allow the network more flexibility in encoding patterns, thus leaving more scope for the pretraining advantage the be demonstrated.

Regardless, it does not appear the McRae-Hetherington training method used in conjunction with the SRN architecture is sufficient to produce a network capable of modelling the task at hand.

C. Method Two - Elman (1993)

The approach used by Elman to overcome catastrophic forgetting was designed especially for SRNs. Based on the principle that as a child develops their attention span increases, this technique is one in which recurrence in the network increases as the pattern file is trained. This technique can also be thought to be relevant to sequence learning. As sequences are learned, the development appears to proceed in the opposite direction. Initially, subjects may be able to report individual elements, but they cannot repeat the entire sequence. Gradually, they are able to recite larger portions of the string until the entire sequence can be produced.

However, as the subject's familiarity with the sequence increases, the process seems to go in reverse. After training, the subject is able to report the sequence start to finish only. As training continues, the subject is able to start reciting the sequence from elements other than the first element of the sequence. This continues until the subject is able to enter the sequence at any point. Based on this observation tests are undertaken in which recurrence is both increased and decreased as training of the sequence continues.

1. Procedure

1.1 Network structure

The network structure used was the same as that described in section two.

1.2 Training Patterns

The training patterns used were based on the Entire pattern set from Simulation One. This set represented the fully recurrent condition. Three other conditions were added, two partially recurrent and one with no recurrence. The two partially recurrent conditions divided the sequence into subgroups roughly approximating the grouping structure used by the experimenter. Thus, for the file Entire2 the recurrent connections were cleared between patterns *gee* and *haitch, pea* and *que* and *vee* and *dbl-u* (described as the group boundaries in the previous section). For the Entire3 file, recurrent connections were cleared at the same points as in Entire2, with the inclusion of clearances for all patterns *aee* through *dee*, and *eex* through *zee*, and between patterns *kay* and *eel* and *tea* and *you*. The Entire3 set of patterns was designed to represent the state of learning most people achieve with the alphabet. The file which contains no recurrence is thought to represent the state of learning most people achieve with the number series. Full file listings can be found in Appendix 1.

2. Preliminary Tests

2.1 Design

As a result of the similarities with Simulation One, the preliminary tests carried out are valid for this investigation. The result from the first set of preliminary tests that remains relevant is the training of the entire file on a naïve network. In all preliminary tests a proportion of trials resulted in the network falling into a local minimum. As in the previous section these trials are excluded when the average number of epochs to train is calculated.

2.2 Results

To achieve four successful training runs, the Entire pattern file produced fifteen local minima, the Entire2 pattern file produced four, the Entire3 pattern file produced three and the Entire4 pattern file produced seven. Table 15 shows the average number of epochs to train each file over the four successful training runs. As the amount of recurrence decreases the number of epochs to train increases, although not greatly.

Pattern File	Epochs to Train
Entire	318
Entire2	306
Entire3	365
Entire4	397

Table 15. Average Number of Epochs to Train the Second Set of Test Files. Across four training runs (rounded to the nearest whole number).

3. Tests

3.1 Test One

Design

The first test followed Elman's training regime. Thus, the four files were trained in increasing order of recurrence. Entire4 was trained first followed by Entire3, Entire2 and Entire.

Results

Table 16 lists the number of epochs required to train each file. These trials were selected for their ability to remain clear of local minima while training the Entire4 pattern file. Looking at the table it is apparent that the files trained after Entire4 have been given a definite training advantage. These results were found when the patterns in file Entire4 were trained in order and when they were trained in random order.

Pattern	Trial 1	Trial 2	Trial 3	Trial 4
Entire4	467	548	466	360
Entire3	8	8	5	6
Entire2	7	8	5	6
Entire	1	1	1	1

Table 16. Training with Increasing Recurrence. The number of epochs to train the files in order of increasing recurrence. As is obvious the files trained after the Entire4 pattern have a significant advantage over training them on their own.

3.2 Test Two

Design

In this set of tests the files were trained in order of decreasing recurrence. This constituted an effort to model the effect of further learning of the sequence after the elements can be produced in the proper order.

Results

Table 17 shows the number of epochs to train each file. As in the previous test the number of epochs is reduced after the training of the initial pattern file, although this advantage is not as great as in section 3.1.

Pattern	Trial 1	Trial 2	Trial 3	Trial 4
Entire	332	341	377	225
Entire2	18	9	12	25
Entire3	36	46	30	20
Entire4	52	45	34	40

Table 17. Training with Decreasing Recurrence. The number of epochs to train the files in order of decreasing recurrence. As is obvious, the files trained after the Entire pattern have a significant advantage over training them on their own, although less of an advantage than in the previous test.

3.3 Test Three

Design

As both of the previous tests have produced results indicating that some form of incremental learning is possible, it was decided to combine the Elman method and the McRae-Hetherington method. To do this, the network is given limited experience of the entire training set (with recurrence), then an attempt is made to train the network on the separate groups incrementally. The network is then tested on the entire pattern file with breaks in the recurrence after each group (Entire2 file).

Preliminary results of this type of training are found in section A(2). It is shown that the four subgroup files, if trained incrementally, require 215 epochs to train, as long as a local minimum is not encountered. Local minima were encountered on two of the four training runs, with difficulties occurring as the third subgroup file was trained.

For this experiment two further recordings based on these files were made. After the four files had been trained, the *tss* for the

network tested with the file Entire2, and the number of epochs to train the file Entire2 were recorded. The Entire2 file was used as it contained the full pattern file with breaks in recurrence between the subgroup files. The average tss was 23.4820. This value did not seen to differ between files in which the training of the four patterns was successful, and those for which it was not. The number of epochs to train file Entire2 also did not differ - the average was 116. However, the only local minimum encountered was on a trial which did not train the four subgroup files successfully.

Tests were done with pretraining of file Entire (full recurrence) and Entire4 (no recurrence), with both files trained either to the criterion tss of less than 1.0000 or for 50 epochs. The first of these was designed to determine if pretraining offered an improvement over naïve networks, and the second because it is unlikely that knowing the sequence subjects would then try to learn it in parts.

Results

The results with full pretraining show an obvious advantage over the results on a naïve network, as is demonstrated in table 18. For both pretraining files the number of epochs to train the four subgroup files was greatly reduced as were the *tss* and number of epoch to train the Entire2 file at testing. None of these networks encountered local minima in training the subgroup files or the Entire2 file. This advantage is likely to be due to the fact that the network had learned the entire pattern file and was given no patterns outside that file to overwrite it. This is not likely to be a useful method of training as it postulates that the subject learns the entire sequence and then goes back and learns it in bits, which is patently not the case.

The more likely occurrence is that the subject has some limited experience of the entire sequence before attempting to learn it piecemeal. Whether or not an advantage was gained by training the network on the pretraining file for 50 epochs only is unclear. Local minima were still encountered when training the third subgroup

file on two of the four networks pretrained with the Entire file and one of the four pretrained with the Entire4 file. There does appear to be a slight reduction in the number of epochs to train each of the subgroup files and in the *tss* to the Entire2 file at test. However, these reductions do not seem to translate to a reduction in the number of epochs to train the Entire2 file. The table appears to show a reduction for pretraining with the Entire file but this may be because the relatively few trials on which the Entire2 file was learned to criterion. For each pretraining file, three of the four networks encountered local minima while training the Entire2 file. This is in contrast to only one of the four naïve networks. One of the naïve networks to successfully learn the Entire2 file did so in 50 epochs, which is the same number as the successful network pretrained with the Entire file.

Pre-train File	Sub-group1 Epochs	Sub-group2 Epochs	Sub-group3 Epochs	Sub-group4 Epochs	Test *tss*	Test File Epochs
none	91	58	68*	256	23.4820	116*
Entire	1	1	1	1	1.2654	13
Entire4	1	1	1	1	1.1023	7
Entire#	41	17	50*	73	11.4183	50*
Entire4#	29	18	90*	113	12.0927	228*

Table 18. Results Using a Pretrained Network. Indicates if the network was pretrained with a fully recurrent pattern file (Entire) or the full pattern file with no recurrence (Entire4), the epochs to train the four subgroup files, the *tss* value for the test file (Entire2) and finally the epochs to train the test file from the *tss* given. # indicates that the pretraining file was trained for only 50 epochs, not to the criterion of less than 1.0000. * indicates that some of the trials in these conditions encountered local minima, such trials are not included in this summary.

From these results it is unclear as to how useful Elman's method of training will be in allowing an SRN to incrementally learn the task at hand. The only success achieved here requires an intuitively unlikely training scheme to be employed.

4. Discussion

The overall result of this section is that neither of the training procedures allowed the network to overcome catastrophic forgetting to the extent required for the task. Specifically, the results of this section are mixed. In the first two tests it was found that training an SRN on files where recurrence either increased or decreased across files, resulted in subsequent files training in fewer epochs then had they been trained on a naïve network. However, when the attempt was made in the third test to adapt this method using a training scheme which more closely resembles that of human subjects, the results were not as promising. There appeared to be no obvious advantage for pretrained networks.

The SRN was being investigated as a mechanism for generating the sequence. This would have been augmented by a SON system which would generate a sequence representation for use in relative order judgements. However, the SRN appears to be inappropriate for its task. This is surprising as generally SRNs are held to be the best type of network for sequence learning.

These findings suggest that an alternative method for generating the sequence must be found. Therefore, a third alternative is needed. The alternative produced consists of combining the SON model with a CQ module. The learning mechanism hypothesised for the SON is dynamically allocating nodes as sequence elements are encountered, and connecting these nodes in one-direction only using simple Hebbian learning. This learning mechanism is inherently incremental. The sequence representation generated, in the form of the SON, can then be recited through the CQ module, thus fulfilling the requirement of incremental learning of the sequence. The CQ extension and the SON learning mechanism are described and investigated in Chapter Seven.

Chapter Seven - SON Extensions

Since the previous chapter has determined that a separate network dedicated to learning the sequence is unlikely to produce the desired results, the alternative of using the representation generated by the SON to produce the sequence must be investigated. This representation is used by extending the SON to include a sequence recitation module. The extension takes the form of Houghton's (1990) Competitive Queue architecture, as described in Chapter Five. This chapter describes the construction of the extended SON/CQ system and the learning mechanism proposed for that system. The ability of the SON/CQ to produce a recitation of the sequence is established in preparation for a further exploration of the learning mechanism over the next two chapters.

A. SON/CQ System

Several changes would need to be made to the CQ model to combine it with the SON. The result of these changes is illustrated in figure 33. The first would be to replace L1 with the SON. In this way the activation values used to generate the sequence elements would be those from the SON. The activation of the SON nodes would proceed in the same way as outlined in Chapter Three, by activation of either the start node of the sequence or by a Standard node within the sequence. Activation in the SON would then be cycled so that the comparative relations among the sequence elements could be established.

The second change would be to replace L2 with a copy of each node from the SON having an inhibitory connection of value 1 between each node in L1 and its corresponding node in L2. This would be necessary as the activation pattern in the SON is such that the earlier an element is in the sequence, the lower its activation. In order to reverse this activation pattern to allow the competitive filter to function properly, an intermediate level must be added. This level would also allow the inhibition of the nodes as they are articulated without disrupting the activation in the SON - a step which may prove unnecessary. After the activation values in the

SON are transposed between layers L1 and L2 the remainder of the model functions in the same way as Houghton described.

The parameters of the Competitive Filter layer (i.e. the strength of the inhibitory connections between units and the excitatory connections within units) would be the only ones to remain as the CQ model was combined with the SON. However, the value returned to L2 by the node in L3 which wins the competition may need to be altered.

This system's ability to recite the sequence is explored Section D.

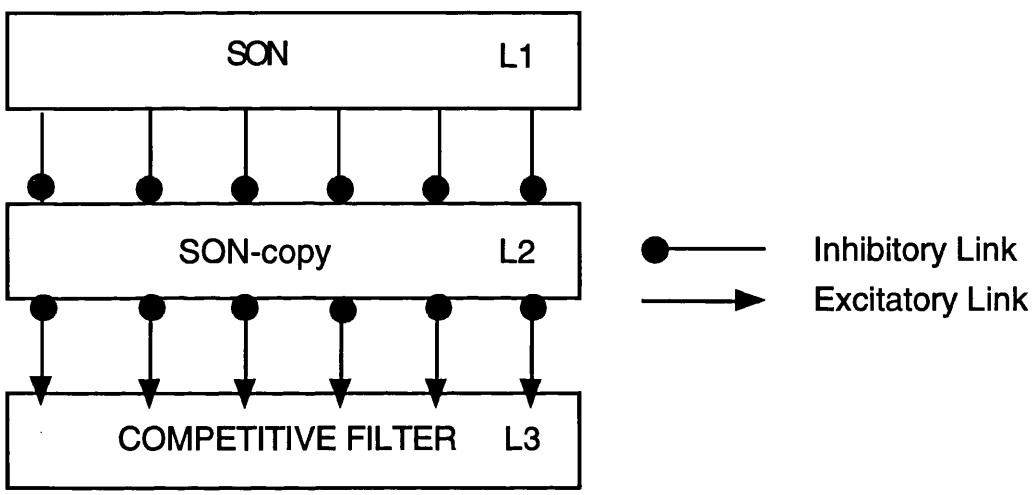


Figure 33. SON/CQ Extension of the SON Model. The changes necessary to allow recitation of the sequence using the representation developed in the SON.

B. System Parameters

By combining the SON and CQ models, the number of parameters in the resultant network increases. These parameters are delineated below.

SON Parameters
1. SON weights.
2. The number of epochs over which outside input is received (*in_eps*).
3. Rate of decay in nodes without input (*decay*).

4. Input value to SON node as activated from outside the network (*input*).

CQ Parameters
1. Weights within and to the CQ portion of the system
2. Inhibition returned to L2 from winning node in the CQ (*rval*).

Combination Parameter
1. Epochs of cycled activation in the SON before the CQ is formed (*eps*).
2. Threshold of activation before a node can join the CQ (*cut-off*).

The influence of each of these parameters must be investigated to determine if the network's ability to recite the sequence is dependent on the tweaking of certain parameters, and not on the architecture itself.

The influence of the number of epochs of outside input (*in_eps*) was investigated in Chapter Three, and will not be replicated. The value of 3 epochs determined at that time will remain over this set of simulations.

The weights within, to, and from the CQ have been designed and investigated in the literature by Houghton (1990), and so will not be examined here. The rate of decay in nodes without input will also be ignored as the only node without input is the *start* node which does not contribute to recitation. This parameter will be explored in Chapter Nine. The remaining parameters are investigated below.

C. Initial Observation

An initial observation is made to determine if the new system is capable of reciting the sequence. Parameters are set to give the system the maximum possibility of success. The influence of the various parameters on this success is investigated in the following sections.

1. SON Parameters

The SON used in this simulation was configured as described in Chapter Four. The only exception is that for this test the node names were shifted down, effectively adding a *start* node to allow the activation values in the net to rise smoothly across all of the number nodes (zero - nine). The weight configuration used was the one found to best replicate Poltrock's (1989) findings of RTs generated in response to relative order judgements. The weight configuration used is illustrated in table 19, the weight pattern repeated across nodes.

To → From ↓	Two	Three	Four	Five	Six	Seven	Eight	Nine
One	0.25	0.15	0.15	0.20	0.10	0.15	0.10	0.15
Two	XXXX	0.25	0.15	0.15	0.20	0.10	0.15	0.10
Three	XXXX	XXXX	0.25	0.15	0.15	0.20	0.10	0.15
Four	XXXX	XXXX	XXXX	0.25	0.15	0.15	0.20	0.10
Five	XXXX	XXXX	XXXX	XXXX	0.25	0.15	0.15	0.20
Six	XXXX	XXXX	XXXX	XXXX	XXXX	0.25	0.15	0.15
Seven	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	0.25	0.15
Eight	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	0.25

Table 19. Weight Configuration Used in Simulation One. Weights from each node to all following nodes. This weight configuration was found in Chapter Four to best approximate the RT data found by Poltrock (1989).

The decay value for nodes which are not receiving input was set to a relatively low value of 0.20. The outside input to the SON nodes was set to 1.0 to indicate activation of the number at which recitation should start.

2. CQ Parameters

The CQ extension to the SON was configured as described by Houghton (1990). The parameters within the CQ were set to 0.09 for negative connections between the nodes in the Competitive

Filter, and 0.5 for the positive connections from each node to itself in the Competitive Filter. The connections to the Competitive Filter was 1.0 from the CQ and -1.0 from the SON to the CQ. The value returned to L2 by the node that wins the competition was set to -1.0.

3. Combination Parameters

The remaining parameters were the epochs before the Queue was formed (which was set to ten), and the activation necessary in a node before it is allowed to join the Queue (which was set to 0.05).

This configuration of network parameters allowed the network to produced the sequence perfectly in order from Zero to Nine. Since there was no variability in the network, the same solution was produced on all occasions.

D. Parameter Investigations

As stated earlier the weights to and within the CQ, the decay for nodes with no input, and the outside input to the SON node to begin recitation are not addressed here. The investigations begin with the parameters unique to this combination of networks, termed combination parameters (see above).

It is important to remember that the network ceased recitation when the largest element in the Queue was reached regardless of the number of elements traversed. This seemed a reasonable approximation of children's counting behaviour, which kept the network from entering infinite loops. It is equivalent to a child knowing she can count to Six and stopping when this value is reached.

1. Combination Parameters

These include the number of epochs of activation cycled in the SON before the Queue is formed, and the amount of activation needed for a node to join the Queue.

The number of epochs of cycled activation before the Queue is formed, allows the generation of activation in the network. This activation starts with activation passing to the first node, and from there it passes to all other nodes in the network. As noted in Chapter Three, the activation starts relatively even across the nodes, then forms a pattern of increasing activation from the first to the last node, and finally a near linear pattern with most nodes responding maximally. Thus, it was expected that as the number of cycles increased, the network would be initially unable to recite correctly, become able and pass out the other side to return to inability.

1.1 Design

The number of epochs before the Queue was formed was increased as the remaining parameters remained fixed at the levels described in the initial observation.

1.2 Results

The results proceeded as expected over the first few epochs (see table 20). The network was initially unable to recite the sequence correctly. Interestingly, this inability took the form of reversals of some adjacent elements, after the number of elements in the Queue reached all elements in the sequence. These reversals may be due to the direct connections taking larger values than more distant connections in the SON weight configuration used. For the first few epochs, element Zero is not produced because it is the most highly active, and Nine is produced first.

The effect of the number of epochs before the Queue was formed was as expected. If the number was too low, activation had not reached all of the nodes, or had, but the levels of activation in last nodes caused them to be output first, which also caused recitation to cease. As the number of epochs increased, the activation in node Zero decayed through lack of input and it ceased to join the Queue. This was the only node to encounter this difficulty up to one hundred epochs.

Epochs	Network Production
1	1 (only element in Queue)
2	5,7,9 (all in Queue)
3	2,1,3,5,4,7,6,9(all in Queue)
4	2,1,3,5,4,6-9 (all in Queue)
5	0,2,1,3,5,4,6-9
6	0-3,5,4,6-9
7	0-9
10	0-9
16	0-9
17	1-9
50	1-9
100	1-9

Table 20. Results of Increasing Number of Epochs of Cycled Activation in the SON before the Queue is Formed. The number in Queue indicates that recitation ceased when element Nine was reached, which was before the Queue was exhausted. Elements connected with a dash (e.g. 0-3) indicates that these and intervening elements were produced in the correct order.

The threshold of activation (or cut-off) necessary to join the Queue is a parameter designed to keep inactive nodes from joining the Queue. The way the program functions is to add all nodes to the Queue inverting its activation. Without the cut-off parameter, inactive nodes join the Queue with activation1.0. This method of operation is inappropriate. The cut-off parameter can be set arbitrarily low. A cut-off of 0.00 is fine if all nodes receive activation, but since that is too strong an assumption, the cut-off is set to 0.05. The only problem occurs if the cut-off is increased to 0.7 when active nodes are kept from joining the Queue. However, there is no valid theoretical reason for setting the cut-off so high. Therefore it is not a problem.

2. CQ Parameters

Since the weights to and within the CQ are explored elsewhere, the remaining parameter is the amount of inhibition returned to the CQ from the Competitive Filter. This parameter prevents the network from producing the same element more than once during recitation. If it is too low the network is expected to repeat elements during recitation. It is meaningless to postulate too large a value for this parameter.

2.1 Design

The amount of inhibition returned was reduced as the other parameters remained fixed at the levels outlined in the initial observations section.

2.2 Results

Inhib.	Network Production
0.1	0,0,0,0,1,0,2,3,1,0,4,2,5,3, 6,1,7,0,4,2,8,5,9
0.2	0,0,1,0,2,3,4,5,6,1,7,0,2,8,9
0.3	0,0,1,2,3,4,4,6,7,0,8,9
0.4	0,1,0,2-9
0.5	0,1-3,0,4-9
0.6	0-7,0,8,9
0.7	0-9
1.0	0-9

Table 21. Effect of Altering the Inhibition Returned from the Competitive Filter. Elements connected with a dash (e.g. 1-3) indicate that these and their intervening elements were produced in the correct order.

The results outlined in table 21, were as expected when the inhibition value was very low. Elements were repeated, especially element Zero, which had the highest level of activation in the Queue.

This investigation shows that while this parameter is necessary, it has a predictable effect on the performance on the network. Thus, the network's ability to produce the sequence is not the product of special conditions of this parameter.

3. Interaction of Epochs and Inhibition

It should be noted that the two parameters are expected to interact. That is, the higher the number of epochs, the more activation in the SON network when the Queue is made, which leads to higher Queue values for each node, and therefore more inhibition required to ensure that Queue nodes do not become re-activated after being produced.

3.1 Design

To demonstrate this interaction, three levels for each parameter were chosen (5, 10 and 15 epochs and 0.5, 1.0 and 1.5 inhibition), while the remaining parameters remained fixed at the levels stated in the initial observations section.

3.2 Results

Epochs → **Inhib. ↓**	**Low** **5**	**Med** **10**	**High** **50**
Low **0.5**	0,2,1,3,5,4, 6,7,8,9	0,1-3,0,4-9	1-9
Med **1.0**	Same as above	0-9	1-9
High **1.5**	Same as above	0-9	1-9

Table 22. Interaction of Number of Epochs Before Queue Formed and the Inhibition Returned from the Competitive Filter. Elements connected by a dash (e.g. 0-9) indicates that these and intervening elements are produced in the correct order.

The interaction results outlined in table 22 were as expected, except in the case of 10 epochs of activation before the Queue was formed and with 0.5 returned as inhibition. In this case 0 became re-activated and re-joined the Queue. In all other cases the number of epochs before the Queue was formed dictated the output of the network.

Again, the combination of these parameters has a predictable effect on the network, indicating that the setting of parameters is principled.

4. SON Parameters

The only parameter investigated here is the weights within the SON. This is the parameter with the largest effect on the network's ability to recite the sequence. The following three simulations demonstrate this effect.

4.1 Simulation One

This first simulation demonstrates the effect of learning two groups of numbers with no connection between them. The weight structure used is outlined in table 23.

To → From ↓	One	Two	Three	Four	Five	Six	Seven	Eight	Nine
Zero	0.20	0.15	0.10	0.50	0.00	0.00	0.00	0.00	0.00
One	XXXXX	0.20	0.15	0.10	0.00	0.00	0.00	0.00	0.00
Two	XXXXX	XXXXX	0.20	0.15	0.00	0.00	0.00	0.00	0.00
Three	XXXXX	XXXXX	XXXXX	0.20	0.00	0.00	0.00	0.00	0.00
Four	XXXXX	XXXXX	XXXXX	XXXXX	0.00	0.00	0.00	0.00	0.00
Five	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	0.20	0.15	0.10	0.05
Six	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	0.20	0.15	0.10
Seven	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	0.20	0.15
Eight	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	0.20

Table 23. The SON Weight Structure Used in Simulation One. This structure represents two groups of numbers learned without connection between them.

The results of this simulation were as expected. The network produced the first four elements only as the remaining elements never became active and did not join the Queue.

4.2 Simulation Two

The second simulation represented two groups of numbers learned with only a connection between the last element of the first group and the first element of the second. The weight structure used is outline in table 24.

To → From ↓	One	Two	Three	Four	Five	Six	Seven	Eight	Nine
Zero	0.20	0.15	0.10	0.50	0.00	0.00	0.00	0.00	0.00
One	XXXXX	0.20	0.15	0.10	0.00	0.00	0.00	0.00	0.00
Two	XXXXX	XXXXX	0.20	0.15	0.00	0.00	0.00	0.00	0.00
Three	XXXXX	XXXXX	XXXXX	0.20	0.00	0.00	0.00	0.00	0.00
Four	XXXXX	XXXXX	XXXXX	XXXXX	0.20	0.15	0.10	0.05	0.05
Five	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	0.20	0.15	0.10	0.05
Six	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	0.20	0.15	0.10
Seven	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	0.20	0.15
Eight	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	0.20

Table 24. The SON Weight Structure Used in Simulation Two. This structure represents two groups of numbers learned with their adjoining nodes connected.

The results of this simulation were also as expected. Since the amount of activation in each subgroup would be approximately the same after a few epochs, when the network attempts to recite the sequence the two groups become intermixed (0,5,1,6,2,7,8,3,9). Four is excluded since it had less activation than Nine.

E. Summary

In summary this chapter has described the only promising sequence model for the task at hand, the SON/CQ model. It is the only model

which appears potentially able to learn a sequence in the fully incremental fashion demanded by this task. The parameter investigations illustrate that the network architecture is responsible for its success and not the tweaking of parameters. As well as demonstrating the influence of the SON weights on sequence recitation, the results of the final two simulations also demonstrate that the network in its present form is incapable of some types of incremental learning. Although nodes can be added to the network, the interconnection of these new nodes has a large effect on the network's ability to recite the sequence. These two problems are addressed in Chapter Nine. As stated in Chapter Three, the SON weights must be constrained by being learned as otherwise they represent too many degrees of freedom. Chapter Eight discusses the learning stages which one would like to simulate in a successful model. It is followed in Chapter Nine by a further update of the SON/CQ, SON/RGN system which is then explored for its ability to learn the sequence and simulate the learning stages desired.

Chapter Eight - Development

This chapter outlines the effects that the system should display as it develops. Thus, the development of the ability to make relative order judgements is discussed as is the development of the ability to sequence elements, taking the number series as an example in both cases. Due to the failure of the sequence networks outlined in Chapter Six to fulfil the needs outlined, a new sequencing system was identified - the SON/CQ. The SON/CQ system's ability to model the two developmental processes is investigated in Chapter Nine.

A. Development of Relative Order Abilities

The first developmental area to discuss is the ability to make relative order judgements. In an attempt to understand how relative order effects develop, Sekular and Miekiewicz (1977) investigated the change in ability to make relative judgements of numeric stimuli with age. They used four groups of subjects whose mean ages were 5.89 (kindergarten), 6.88 (first grade), 10.18 (fourth grade) and 13.13 (seventh grade). They found that with increasing age, subjects' response times to relative order questions decreased significantly. The kindergarten group was significantly slower than all groups. First grade was significantly slower than all but kindergarten. Fourth grade, seventh grade and adult groups did not differ significantly.

In all age groups, the SDE was found. Error rates decreased from kindergarten to adult and tended in all groups to decrease with increase in Distance. This is the opposite effect expected in speed/accuracy trade-off situations. There was no correlation between the number of errors and mental age. Sekular and Miekiewicz found that RT differences were significant when measured in linear terms. The slopes of the SDE lines for the different age groups were significantly different, but their basic shape remained the same (monotonic decreasing). Slope increased

as age decreased. These findings indicate that the basic RT effects are present at least from the time children enter school in kindergarten, and that as the sequence is used they become more pronounced, but do not change substantially.

Pike and Olsen (1977) investigated the question of whether children's performance decrement on a more/less task is due to the original representation assigned to events and/or questions, or in a different set of procedures for comparing those representations. Their first experiment supports a multi-representational view i.e. one forward representation and one backward representation. That is, immature children code addition in terms of "A has more" and subtraction in terms of "A has not more", "B has more". In this case, "Less" questions will always be slower. More mature children code subtraction in terms of "A has less", so answering a "more" question will be slower. Pike and Olsen theorise that immature children have *one* through *n* equals "more" so that to say something is "more" is easiest and "not more" is next. "Less" questions are more difficult because they require recoding the "not more" answer, which is characterised as clearing activation and swapping the active node. Mature children have two representations one knows "more" and one knows "less". Addition activates the "more" net and subtraction activates the "less" net, with both defaulting to "more" coding.

Their analysis suggests that the comprehension process involved in this question-answering task is fundamentally the same for all children. It consists of a series of independent recoding operations that have the goal of arriving at congruence between the display and the question. What differs in the two cases primarily, is the representation assigned to the original display. For the pass operations subjects (ones who show competence at both addition and subtraction) the act of subtraction is primarily compatible with the less question i.e. the representation of subtraction involved a coding similar to that given to the less question. Hence,

for these children the less item was relatively easy. For those in the fail operations group the coding assigned to the act of subtraction was not similar to that given to the less question. For these children this item was most difficult of all.

For the fail operations group there were more addition than subtraction responses. These children, on viewing subtraction, could conceivably remember that something was done which was not addition, therefore A does not have more. This would account for the results. Pike and Olsen also found that the nature of the representations seems to be related more closely to the cognitive than the linguistic competence of the child. Children who did not understand the relation between addition and subtraction as alternative and complementary means for producing a particular effect, coded both addition and subtraction in terms of "more" even though some of them could produce both terms as descriptions of static inequalities.

Pike and Olsen find that in immature children coding is in terms of sextant i.e. a perceptual property as in length or adding, while mature children code in terms of intentional properties i.e. "more" contrasts with "less" and with "same". This appears to underlie what Piaget called operational thought.

In another type of study Dehaene (1992) looked at some numerical abilities, including comparison and addition approximation, in preverbal children and animals. He argues that these abilities do not depend on competence for language, but require access to an analogical representation of numerical quantities, and that they constitute a separate preverbal system of arithmetical reasoning.

He found that the ability to select the larger of two numerosities appears in children around 14 months of age, and the distance effect in numerical comparison is present from 6 years of age (the earliest age at which its been tested). He also found that numerical

comparison can be taught to animals (i.e. pigeons and rats), and a distance effect is again found. He further notes that it is striking that the abilities evident in animals and young children (estimating) coincide with those that remain accessible even in deeply aphasic and acalculic human adults.

Wagner and Walters (1982) divide numerical abilities into a collection of different developmental strands. They define magnitude evaluation judgements as all those behaviours that designate the plurality of a collection without counting. These might take any of the following verbal forms: "two", "three", "lots", "many", "little", etc. They define differentiation judgements as essentially algebraic in nature, with these expressed by verbal forms like "more", "less", "same", etc. They believe that the important factor in these judgements is that they are defined as occurring without overt counting. Subitization (Klahr and Wallace, 1976 - the automatic apprehension of the values 1, 2 and 3) for example, would be classified as a magnitude evaluation mechanism.

In an attempt to further their explanation of the difference between evaluation and differentiation, Wagner and Walters refer to a paper by Gréco (1962), which distinguishes between "quantity" judgements and "quotity" judgements. The former judgements are those coming in response to questions such as "how many are there?" and the latter in response to questions of the form "where is there more?". The quotity-quantity distinction is substantially the same as is made between evaluation and differentiation.

Wagner and Walters question whether there might be a corpus of "central skills" that are domain-independent that produce numeric behaviours (errors, etc.) almost as a by-product. To this aim they ask: do children evaluate magnitudes before they can count them? They attempt to answer this by determining if infants, who by definition cannot count (because they have no language), can

evaluate and discriminate between magnitudes. They find, as Dehaene (1992) did, that numerosity itself (independent of specific perceptual cues) can be perceived by infants.

If these results are confirmed by other investigators, then clearly counting is not a prerequisite for small magnitude evaluation. One caution worth noting is that the infancy results to date do not speak to the issue of what underlies the discrimination of two from three. Although an infant sees three as different from two, it does not mean that she sees it as "more than" two. Dehaene contends that the determination of "more than" is due to the architecture used to discriminate.

Wagner and Walters believe that the development of number starts propositionally in the magnitude evaluation domain. With the same classificatory skills that an infant uses to break down the world into adult-like classes (e.g. animals, humans, furniture, faces; they cite Ross, 1977) she classifies quantities at least up to three. This is, as yet, only a propositional designation because (by hypothesis) there is no relation between one and two, one and three and two and three. Even though a child might agree that two rows of three (or two) counters are the same (more precisely: "members of the same class"), this does not imply that they have equal quantities.

With this caveat in mind Wagner and Walters turn to magnitude differentiation. They believe that quantitative differentiation of collections is possible only after the child has abstracted the notion of "arbitrary member of a set". Without this ability, they do not see how the distinction between two sets defined equally in intention can be made in extension.

Wagner and Walters see their task in analysing the longitudinal data in terms of the counting principles as twofold. First, they are trying to separate the semantic use of numbers (under the name of magnitude evaluation) from the procedural use (which is clearly

counting in the earlier instances). Second, they expect to show that these two processes develop separately - that the ability to estimate small magnitudes emerges quite early and that the mature notion of counting as a procedure with which to necessarily generate a conclusion, is developed more than a year later.

From this analysis they have developed scales for the development of numerical ability. They note that the scales below should be read with caution, if not suspicion. They are aware of the fact that just because they say that step four comes after step three, this does not make the two steps developmentally related. Ordinal scales are in the first analysis simply descriptive of temporal progressions. Although they group two kinds of behaviours in the same scale, this does not mean they share the same computational mechanisms. Not only can the same behavioural event serve plural conceptual ends, but correspondingly, the same conceptual end can be plurally served (i.e., a particular counting scheme can produce different kinds of errors).

For these reasons, they consider it feckless to try and establish one ordinal scale of number development because there is not one developmental filiation to be traced. They summarise their data as saying that the toddler does not have a number concept; he has several number concepts, each at a different stage of evolution. They can only speculate as to whether there is an underlying "superscale" beneath the numerous hypothesised ordinal scales. They can predict, however, what elements any superscale could draw on. Specifically:

(1) as children leave infancy they are equipped with the cognitive capacities to discriminate four numerosities (1, 2, 3 and 4).

(2) They have the capacity to form abstract adult-like classes where memberships cannot be specified either in extension or intention (e.g., "humans", "animals", "furniture").

(3) They have a cross-modal intensity metric to which they can map the discriminated numerosities. The intensity metric is an amodal scale that mediates comparisons between sensory events in terms of their physical intensities.
(4) They have a predisposition both to list exhaustion and bijections, which means that an intensity value will have only one image in the numerosity set. [Wagner and Walters, 1982, p.157]

B. Development of Sequence Abilities

The first area reviewed is the influences on construction of novel serial orders. Potts et al. (1978) present two accounts of the interaction of linguistic form and pairs presentation order on serial order construction. They both refer to "transitive inference" (TI), which is used to designate the subject's ability to connect two elements into a sequence. The first explanation is presented by Huttenlocher (1968). He hypothesises that when two sentences are presented containing elements to sequence, the grammatical status of elements described in the second sentence influences speed and accuracy of making the TI. He generalises that the TI is easier to make when the new element is the subject of the second sentence. The second explanation, by Potts and Scholz (1975), is that the time to study the two premises and the time required to answer the sequence question are equal to the time to make the TI. They found that the time to study the sentences did not vary with grammatical status, but with whether or not the adjective used in each premise was congruent with the placement of the end-term item described in the premise. This congruity principal makes predictions about all presentation orders. It was also found to account for a greater proportion of the interaction variance in the data than the grammatical status principle of Huttenlocher. The expression of a relationship that makes explicit the polar opposition of the

underlying dimension seems to bias the constructive process in a subtle way.

Foos (1975; et al., 1976) developed a model which represents an attempt to extend the memory load hypothesis to account for the more subtle differences in the construction processes that arise when elements of a new relationship match with elements already processed, and also when the new pair does not have an element that matches with old information. The model assumes an ordered rehearsal buffer in STM. He hypothesised that when items are encountered they are entered into the memory buffer with a marker i.e. 52 - 52⊕. These can be transformed by two processes i.e. M1: 52, 23 = 52⊕23⊕ = 523⊕ or M2: 52, 35 = 52⊕35⊕ = 352⊕. The second process requires rearrangement of the terms before the transformation can be made.

His experiments confirmed that the processing of M2 sequences is both slower and less accurate than the processing of M1 sequences. Smith and Mynatt (1977) produced the same results. Information can be presented in sentences or pairs. The central point of the extension of the memory load hypothesis by Foos was that when the total demands on memory exceed capacity, both markers for pairs and sentence organisation are abandoned, leaving an ordered string of elements in memory.

Bryant and Tarabasso (1971) demonstrated that the use of both forms of the comparative (Longer and Shorter) is necessary for teaching young children (4 - years old) the relations between pairs of sticks. They also demonstrated that the number of errors on the training pairs (premises) containing end terms was significantly less than on relations in the middle of the 5-term ordering. Riley and Tarabasso suggested that such a serial-position would be obtained if children were constructing orderings from the ends inward rather than storing each "premise" relation separately in memory. They give two compelling reasons for this (1) serial-

position effects have been observed in numerous studies of serial-list learning and in situations in which subjects learned to map responses onto existing continua. (2) a major problem for young children in a task such as this is the memory load. The linear-order representation eliminates the redundancy contained in encoding a set of ordered relations with common elements, thus reducing the memory load.

Tarabasso et al., (1975) found that data from 6-year olds, 9-year olds and college students clearly supports the ends-inward model. Although there was a reliable decrease in the total number of errors (and numbers of trials to criterion) made with increasing age, the patterns of errors were similar across age groups. The data are important because they indicate that both children and adults learn a series of comparative relations in the same way, by constructing a single ordering of all the elements. No distance effects were found for test pairs that include one of the end terms of the ordering. One frequently observes that RT to any pair containing an end-term is unusually short. No end-term effects are observed in the number-comparison experiments. No agreement was reached on the reason for this.

The second area addressed here is the effects found when a serial ordering of elements is expanded upon once the essentials of the ordering have been established. A five level elaboration of the number word sequence in children was produced by Fuson (1988):

1. String Level: The words are a forward-directed, connected, undifferentiated whole.
2. Unbreakable List Level: The words are separated, but the sequence exists in a forward-direction recitation form and can only be produced by starting at the beginning.
3. Breakable Chain Level: Parts of the chain can be produced starting from arbitrary entry points rather than always starting at the beginning.

4. Numerable Chain Level: The words are abstracted still further and become units in the numerical sense; thus sets of sequence words can themselves represent a numerical situation and can be counted or matched.
5. Bidirectional Chain Level: Words can be produced easily and flexibly in either direction.

Just After and Just Before relations appear during the Unbreakable List Level. Backward skills (counting down) at the Breakable Chain Level. [Fuson, 1988, pp. 46-49].

She noted that different parts of the sequence could be at different levels of development at the same time and that not all children have a string level.

The pauses that occur when a subject learns a sequence are also very important to the type of representation which is developed in response to that sequence. Wilkes (1972) looked at pause literature and found that:

1. A positive correlation exists between pause duration during acquisition and accuracy in recall. (Belmont and Butterfield, 1969).
2. Regrouping a prose passage in violation of pause boundaries leads to impaired learning (Suci, 1967).
3. Pause-defined grouping patterns during the reading of serial lists are reproduced in immediate recall (McLean and Gregg, 1967).
4. Pause-defined groupings serve as differentiated units during sub-sequence probed recall (Wilkes and Kennedy, 1970).
5. Changing the grouping pattern of a list during learning interferes with the usual improvement in performance following repetition (Bower and Winzenz, 1969). This assumed that input grouping influences perceptual coding, which in turn determined an address in memory at which the trace for that list was stored. Changes in grouping

> resulted in the string being shunted to a new address with a resultant loss in cumulative learning. [Wilkes (1972), p. 206]

They also found that changes in grouping affected recognition memory. Holding certain subgroups constant only helped when it applied to the first group of a string. It was concluded that initial subgroups of a multi-grouped string critically determined addresses in the memory store. Thus, changing these subgroups caused changes in memory location which were the major source of interference. They found that a drop in the duration of encoding pauses as the learning criterion approached, suggested a shift from productive encoding to maintenance rehearsal, as did the additional observation that the duration of encoding pause tended to be more closely related to learning speed during early learning trials.

C. Summary

It is important that any system attempting to model sequence learning and representation has the ability to explain why different stages occur in learning and the use of the resultant representation. In the following chapter the SON/CQ and SON/RGN systems are discussed in relation to their ability to explain the phenomena described in this chapter.

Chapter Nine - Further SON Extension and Exploration

In this chapter the SON is extended by adding a second layer which is designed to account for the sequence effects which appear to stem from a hierarchical construction. The hierarchical SON is then explored for its ability to model the stages encountered in sequence learning (SON/CQ), and the development of and effects associated with relative order judgement abilities (SON/RGN). A listing of the final program is given in Appendix 3. It is found that this system has great potential to explain these stages and to generate all of the basic relative order judgement effects outlined in Chapter One.

A. Hierarchical Construction

1. Description

As outlined in Chapter Five, many of the sequence effects seem to indicate that the storage is in the form of a hierarchy. These effects revolve around the blocking of the sequence i.e. the Breakable Chain level in Fuson's developmental sequence. To account for these effects a second layer of SON nodes was added to the network. The two layers are now referred to as the Word Layer and the Sequence Layer. Each node in the Sequence Layer receives connections from a sub-portion of the Word Layer nodes. The Word Nodes constitute the members of the sub-sequence accessed via that Sequence Node (see figure 34). The Sequence Node can only activate the first element of its sub-sequence. However, there appears to be no reason why Sequence Nodes could not send connections to all of their sub-sequence elements. The cycling of activation before the Queue is implemented would ensure that similar activation patterns developed in both cases. The functioning of the CQ ensures that recitation begins at the initial element of the sub-sequence. Since the results should be the same, one connection was used for simplicity's sake.

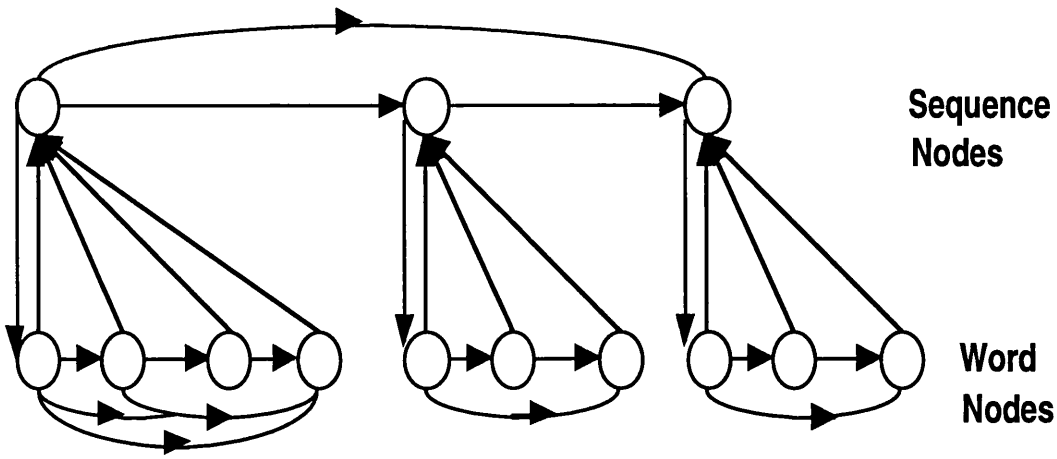


Figure 34. Hierarchical SON Structure. Each sequence node accesses the first element of its sub-sequence, while all word nodes access their sequence node.

Recitation of the sequence proceeds in the same way as for a single layer SON. That is a Competitive Queue is generated, with elements to be output taken from the Word Nodes only. A Competitive Queue is also attached to the Sequence Nodes to allow access to all of the sub-sequences, see figure 35.

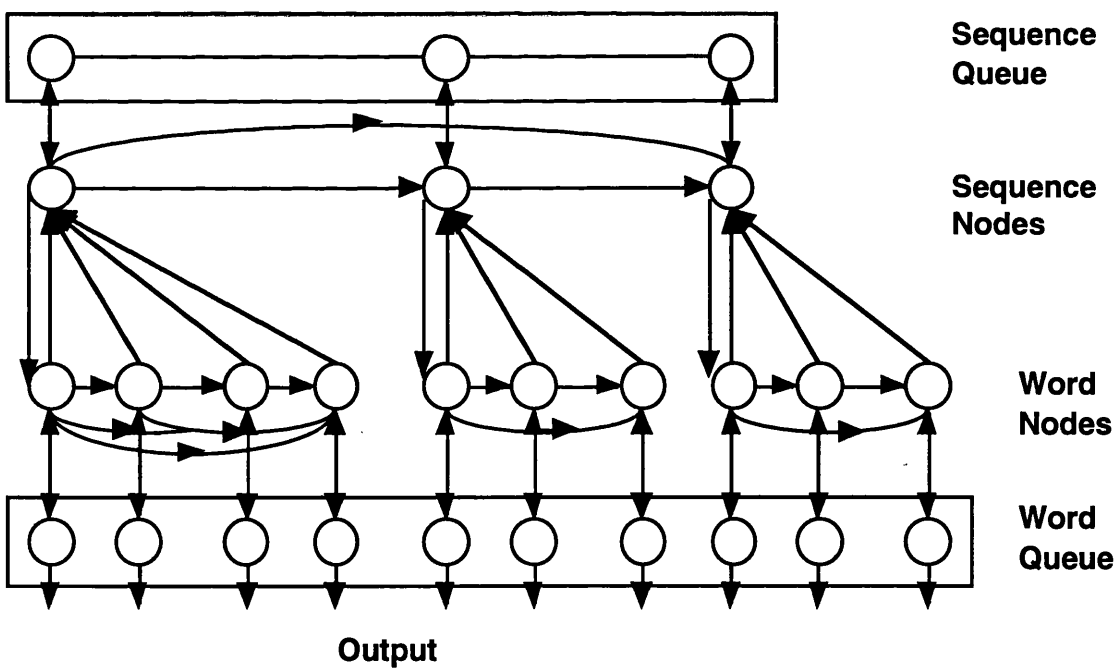


Figure 35. Structure of Recitation System with Hierarchical SON. Competitive Queues are attached to the sequence and word layers, with output generated only from the word Queue. The Competitive Filters attached to each Queue are not pictured.

Recitation of a typical blocked sequence takes place in stages. First the word node representing the desired starting point is activated. This Word Node sends activation to its Sequence Node. Activation is cycled around the Sequence Layer for a number of epochs (*eps*). For the first few epochs the Word Node continues to send activation, the exact number is determined by the (*in-eps*) parameter. During the remainder of the epochs, the Word Node that fired decays. The activations from the Sequence Layer are then inverted (subtracted from one) and used to form a Queue. Only nodes with activation above criterion (*cut-off*) are allowed to join the Queue. The Queue members are sent to the Competitive Filter and compete, as outlined above, until only one remains active. The Queue node corresponding to the winning Queue member is inhibited (*rval*) and the activation of its Sequence Node is set to 1.0, with all other Sequence Node activations set to 0.0.

Activation is then cycled in the Word Nodes, (*eps*), with (*input*) from the Sequence Nodes for the first few epochs (*in_eps*). The nodes with activation values above criterion (*cut-off*) are inverted and join the output Queue. These are sent to the Competitive Filter and compete until a winner is chosen. The winner's Queue node receives inhibition (*rval*). The competition of the Word Queue values continues until there are no Queue members with positive activation remaining.

If the desired *stop* node has not been reached, then the second stage of recitation is entered beginning with a competition among the values in the Sequence Queue (the Sequence Queue is not reformed, it remains from previous competition) and continuing until the Word Queue is exhausted. The stages continue until the *stop* node is reached.

It is important that sub-sequences are not connected directly on the Word Layer. Such direct connections would cause activation to develop in nodes belonging to other sub-sequences. These nodes would then interfere with the recitation of the sequence as demonstrated in Chapter Seven. In some situations there is no

reason for sub-sequences to be directly connected (e.g. area code - local code - phone number). In other cases such interference may occur as the sequence learning progresses from a collection of sub-sequences to one fully interconnected sequence.

2. Simulations

This section is used to determine if changing the structure of the SON/CQ affects the network's ability to recite the sequence. As was seen in Chapter Seven a single layer SON/CQ could not learn a sequence in blocks and successfully recite that sequence. Learning in blocks is simulated here by pre-setting the weights in the network. The learning of weights is addressed in section C.

The setting of the parameters in the SON/CQ network was discussed in Chapter Seven. There it was determined that the parameters had explicable, and theoretically justified effects on the output of the network. Since the Word SON/CQ portion of the network is identical to the single SON/CQ from Chapter Seven, there is no reason to repeat the investigations done there. This leaves the parameters associated with the Sequence SON/CQ portion of the network. These parameters include the number of epochs of activation cycled before the Queue is formed (*eps*), the threshold of activation required before a node can join the Queue (*cut-off*), and the amount a node's activation decays when it is not receiving input (*decay*). A final parameter arises from the connection of the Sequence and Word portions of the network. That is the number of epochs over which the Word nodes receive input from their Sequence Nodes (*in-eps*).

2.1 Initial Observation

Initially it is important to determine that the network is indeed able to recite the sequence. The parameters were set to values described in Chapter Seven as medium values for each parameter (displayed in table 25). The SON weights used were those in table 19 Chapter Seven, except that the addition of the second layer allowed the *start* node to become an ordinary word node, labelled node Zero. The new weight configuration is displayed in Table 26.

Only a single Sequence Node was used. The *in-eps* parameter was set to 3, as this was also considered to be a medium value for this parameter.

Param	Value
decay	0.2
input	1.0
rval	1.0
eps	10
cut-off	0.05

Table 25. Medium Parameter Values. The values determined in Chapter Seven to represent the medium value for each parameter.

To → From ↓	One	Two	Three	Four	Five	Six	Seven	Eight	Nine
Zero	0.25	0.15	0.15	0.20	0.10	0.15	0.10	0.15	0.10
One	XXXXX	0.25	0.15	0.15	0.20	0.10	0.15	0.10	0.15
Two	XXXXX	XXXXX	0.25	0.15	0.15	0.20	0.10	0.15	0.10
Three	XXXXX	XXXXX	XXXXX	0.25	0.15	0.15	0.20	0.10	0.15
Four	XXXXX	XXXXX	XXXXX	XXXXX	0.25	0.15	0.15	0.20	0.10
Five	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	0.25	0.15	0.15	0.20
Six	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	0.25	0.15	0.15
Seven	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	0.25	0.15
Eight	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXX	0.25

Table 26. Weight Configuration Used in Initial Observation. Weights from each node to all following nodes. This approximate weight configuration was found in Chapter Four to best approximate the RT data found by Poltrock (1990).

The resultant network was able to recite the sequence perfectly, establishing that the restructuring of the network did not inhibit its ability to produce the sequence.

2.2 Block Investigation

Secondly, it is important to demonstrate that the new network structure is able to recite a sequence that has been learned in

blocks. For this investigation the network was split into three blocks i.e. three Sequence Nodes and ten Word nodes as in figure 35. SON weights were set to identical values for each connection, which was considered to represent the simplest weight structure available. This structure is outlined in tables 27 and 28 below.

To → From ↓	Zero	One	Two
Zero	XXXXXX	0.20	0.20
One	XXXXXX	XXXXXX	0.20
Two	XXXXXX	XXXXXX	XXXXX

Table 27. Weights in the Sequence SON Portion of the Network for Block Investigation.

To → From ↓	One	Two	Three	Four	Five	Six	Seven	Eight	Nine
Zero	0.20	0.20	0.20	0.00	0.00	0.00	0.00	0.00	0.00
One	XXXX	0.20	0.20	0.00	0.00	0.00	0.00	0.00	0.00
Two	XXXX	XXXX	0.20	0.00	0.00	0.00	0.00	0.00	0.00
Three	XXXX	XXXX	XXXX	0.00	0.00	0.00	0.00	0.00	0.00
Four	XXXX	XXXX	XXXX	XXXX	0.20	0.20	0.00	0.00	0.00
Five	XXXX	XXXX	XXXX	XXXX	XXXX	0.20	0.00	0.00	0.00
Six	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	0.00	0.00	0.00
Seven	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	0.20	0.20
Eight	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	0.20

Table 28. Weights in the Word SON Portion of the Network for Block Investigation.

The only connections from the Word Layer to the Sequence Layer were from each Word Node to its Sequence Node. Thus, Words Nodes Zero, One, Two and Three connected to Sequence Node Zero, Word Nodes Four, Five and Six connected to Sequence Node One, and Word Nodes Seven, Eight and Nine connected to Sequence Node Two. These connections were set arbitrarily to 0.5.

Following the principal that Sequence Node only feeds information to the first node of its sub-sequence, Sequence Node Zero was connected to Word Node Zero, Sequence Node One was connected to Word Node Four and Sequence Node two was connected to Word Node Seven. These connections were set to 0.8, again arbitrarily.

The resulting network was able to recite the sequence successfully only after the Word Queue was adapted to interact with the Word SON. The winning node of the competition had to have its activation set to zero, otherwise when the second Sequence Node fired and spread activation to the second sub-sequence, the residual activation in the first sub-sequence caused those elements to be produced again. This interaction between layers also takes place in the Sequence portion of the network, as described above. The activation of the winning node is set to 1.0 with all others set to 0.0.

Not only was the network able to recite the sequence from the beginning, but it was able to enter the sequence at the initial block elements, having been started at any member of the block. Thus, if the desired starting point was Three, which is in the first block, the network began reciting at Zero. This corresponds to the Breakable Chain level of development as explained by Fuson (1982) and in Chapter Eight. The correspondence between sequence learning and the levels of development is addressed further in section C.

2.3 Parameter Investigation

The final set of simulations addresses the setting of parameters specific to the Sequence Layer and the combination of the Sequence and Word Layers, using the blocked sequence from the previous simulation. These parameters are the number of epochs of cycled activation in the Sequence Layer before the Queue forms (*eps*) and the number of epochs during which the Word Nodes receive input from the Sequence Nodes and vice versa (*in_eps*). Both of these parameters should interact with the *decay* parameter.

To limit the number of free parameters, the number of epochs of activation cycled before the Queue forms (*eps*), the number of epochs over which the nodes receive outside input (*in-eps*) and the *decay* parameter will be the same for both layers. Since the general effect of these parameters is known for SON/CQ systems, only their interaction will be investigated here.

It is expected that too low an *in_eps* value will result in nodes decaying before they are able to join the Queue. This effect may be offset by a low *decay* value and/or a low *eps* value. A high *in_eps* value is expected to produce large activation values, and thus large CQ values which may not be fully countered by the *rval*. This is again expected to be offset by high *eps* or *decay* values.

Eps → **Decay ↓**	**Low** **5**	**Med** **10**	**High** **50**
Low **0.05**	0-9	0-9	1,2,3 1,5,2,6 (6) 1,5,8,2,9(9)
Med **0.20**	0-9	0-9	5,6 5,8,9 (5)
High **0.80**	5,6 5,8,9 (5)	5,6 5,8,9 (5)	5,6 5,8,9 (4)

Table 29. Interaction of Number of Epochs Before Queue Formed and the Inhibition Returned from the Competitive Filter, with in_eps = 1. Elements connected by a dash (e.g. 0-9) indicates that these and intervening elements are produced in the correct order. If a number in parentheses follows a list of number this is the number of elements in the Queue. It indicates that the highest element in the Queue was reached before it was exhausted.

The results of these simulations, illustrated in table 29 are as expected. With a low *in_eps* value, high values for the *eps* and *decay* parameters cause difficulty, with these parameters interacting somewhat. The effect of a high number of epochs before the Queue is formed is somewhat offset by a low *decay* value. This indicates that the problem is arising because nodes are decaying

before they have a chance to join the Queue. With medium and high *decay* values Sequence Node Zero does not join the Sequence Queue causing the first four elements to be excluded.

There appears to be no explanation of why nodes in previous blocks become re-activated and are produced in the following block (e.g. Five is produced twice when either decay or epochs or both are high.

Eps → **Decay ↓**	**Low** **5**	**Med** **10**	**High** **50**
Low **0.05**	1,2,3 (4) 0,1,5,6 (7) 2,4,3,5,8,9 (8)	0-9	0-9
Med **0.20**	0-9	0-9	5,6 5,8,9 (5)
High **0.80**	5,6 5,8,9 (5)	5,6 5,8,9 (5)	5,6 5,8,9 (4)

Table 30. Interaction of Number of Epochs Before Queue Formed and the Inhibition Returned from the Competitive Filter, with in_eps = 3. Elements connected by a dash (e.g. 0-9) indicates that these and intervening elements are produced in the correct order. If a number in parentheses follows a list of number this is the number of elements in the Queue. It indicates that the highest element in the Queue was reached before it was exhausted.

The results of the second set of simulations are recorded in table 30. The only difference between the results of the first (low *in_eps*) and second (medium *in_eps*) simulation sets is that the combination of low *eps* and low *decay* causes problems in the second, but not the first, and the combination of high *eps* and low *decay* causes problems in the first, but not the second. The pattern of results in the second set is due to the low *eps* combining with the low *decay* to allow nodes to become re-activated in the Queue, leading to elements being produced in inappropriate subgroups,

while high *eps* and high *decay* combine with medium *in_eps* to counteract each other.

The results of the third set of simulations (see table 31) are also as expected i.e. the high *in_eps* value combined with high *eps* and low *decay* is able to produce the sequence, because the activation in the nodes is so strong that it does not decay even over a large number of epochs. However, medium *eps* means that the first node of the Sequence Layer receives input on every epoch, and since its activation does not decay, it remains more active than the other Sequence Nodes and thus does not win the Competition. Since the Competition ends when the highest node is fired the initial node does not get the chance to win. This combination of parameters also produces responses in inappropriate subgroups as explained above. When *decay* is high, all Sequence Nodes fire and only responses in inappropriate subgroups remain a problem.

Eps → **Decay ↓**	**Med** **1 0**	**High** **5 0**
Low **0.05**	5,6 (3) 4,5,8,9 (6)	0-9
Med **0.20**	5,6 (3) 4,5,8,9 (6)	5,6 5,8,9 (5)
High **0.80**	1,2,3 (4) 1,2,5,6 (6) 1,5,2,8,3,9 (9)	5,6 8,9(4)

Table 31. Interaction of Number of Epochs Before Queue Formed and the Inhibition Returned from the Competitive Filter, with in_eps = 10. Elements connected by a dash (e.g. 0-9) indicates that these and intervening elements are produced in the correct order. If a number in parentheses follows a list of number this is the number of elements in the Queue. It indicates that the highest element in the Queue was reached before it was exhausted.

When the *eps* parameter is given a high value and the *decay* is medium or high the activations in the initial Sequence Node and the

first Word Nodes of the remaining subgroups decay to where these nodes may not join the Queue. Again, there appears to be no reason why elements are produced in inappropriate subgroups during these simulations - the previous explanation does not hold.

The network functions in predictable ways with the manipulation of these parameters as well, indicating again that it is the architecture of the system that is producing the desired response, and not the tweaking of specific parameters.

B. Learning

The final and most important aspect of the SON/CQ system to address, is learning. As was first pointed out in Chapter Three, for this system to be viable the weights within the SON portions must be learnable, otherwise it may be construed that the performance of the network is dependent on the setting of weights without theoretical motivation. The system needs, not only to learn, but also to demonstrate the potential to exhibit the stages associated with the development of sequences outlined in Chapter Eight.

The dynamic allocation of nodes as the sequence elements are encountered is not addressed here. The assumption is that the sequence learner is aware of the existence of the sequence elements as individual elements, and is learning to associate them into a sequence.

1. Description

Learning in the SON system is through Hebbian update of the weights. All SON connections in the Sequence and Word Layers, as well as the connections between the layers are plastic. A simple Hebbian learning rule is used:

$$W_{ij} \mathrel{+}= act_i * act_j * lrate$$

where:

W_{ij} = weight from unit i to unit j
act_i = activation of unit i
act_j = activation of unit j
$lrate$ = learning rate

For the purposes of the present simulations it did not matter that the weights grew increasingly large. However, if a much larger training set were to be used, some control would need to be placed on the size of the weights.

Weights within the Word Layer, and from the Word Layer to the Sequence Layer were learned at the same time using the same training patterns) This would correspond to exposure to a sub-part of the sequence and knowledge that it was a sub-part. The training set was designed to mimic (in a very simple manner) Word Nodes with fixed activation intervals. That is, nodes in the sub-sequence became active one at a time from the beginning, and then were turned off (decayed) one at a time, also from the beginning. This activation span should also be more rigorously defined in future experiments.

Weights from the Sequence Layer to the Word Layer were learned separately. This would correspond to the association between thinking "subset one" then "A", for example. As described above, there appears to be no reason why Sequence Nodes could send connections to all of their sub-sequence elements. Since the results should be the same, one connection was used for simplicity's sake.

Finally, weights between the Sequence Nodes were again learned separately. This would correspond to thinking "sub-sequence one, then sub-sequence two", etc. In the case of a telephone number this may consist of "area code - 3-digit number - 4-digit number". The next section examines training files and the weights they produce.

2. Simulations

These simulations are designed to illustrate the ability of the network to generate the weight patterns outlined in the previous simulation section using realistic training files. This would demonstrate that the network is capable of modelling the learning and recitation of a sequence due to its architecture and not to the specific manipulation of parameters.

2.1 Initial Observation

The first simulation looks at the development of a fully interconnected sequence, with no sub-sequences. This is the basic SON described in Chapter Three. To generate this weight structure the nodes are considered to stay "active" for ten epochs. This may constitute a rather long period, but a shorter period could be used to generate the full interconnection of a shorter sequence. The learning rate was set to a moderate value of 0.2. The effect of learning rate on the weight structure and recitation is addressed in the next section.

The SON/CQ parameters were set to the medium values described in Chapter Seven. These values are listed in table 32.

Param	Value
decay	0.2
input	1.0
rval	1.0
eps	10
in_eps	3
cut-off	0.05

Table 32. Medium Parameter Values. The values determined in Chapters Seven and Nine to represent the medium value for each parameter.

A very simple pattern of one pass through the ten patterns, with each node decaying fully after ten epochs, was used. This training file is listed in table 33.

Patt No.	Pattern	Patt No.	Pattern
1	1 1 1 0 0 0 0 0 0 0 0	**10**	1 0 1 1 1 1 1 1 1 1 1
2	1 1 1 1 0 0 0 0 0 0 0	**11**	1 0 0 1 1 1 1 1 1 1 1
3	1 1 1 1 1 0 0 0 0 0 0	**12**	1 0 0 0 1 1 1 1 1 1 1
4	1 1 1 1 1 1 0 0 0 0 0	**13**	1 0 0 0 0 1 1 1 1 1 1
5	1 1 1 1 1 1 1 0 0 0 0	**14**	1 0 0 0 0 0 1 1 1 1 1
6	1 1 1 1 1 1 1 1 0 0 0	**15**	1 0 0 0 0 0 0 1 1 1 1
7	1 1 1 1 1 1 1 1 1 0 0	**16**	1 0 0 0 0 0 0 0 1 1 1
8	1 1 1 1 1 1 1 1 1 1 0	**17**	1 0 0 0 0 0 0 0 0 1 1
9	1 1 1 1 1 1 1 1 1 1 1	**18**	1 0 0 0 0 0 0 0 0 0 1

Table 33. Training Patterns used for Word to Sequence and Word SON weights. The Sequence to Word weights were trained with five patterns in which only Word Node Zero and Sequence Node Zero were active.

The results of this simulation were as expected. The network was able to recite the sequence from Zero to Nine. Since there was only one Sequence Node, recitation began at Zero regardless of the desired start point. This was again as expected. The weight structure generated is illustrated in table 34.

To → From ↓	One	Two	Three	Four	Five	Six	Seven	Eight	Nine
Zero	1.80	1.60	1.40	1.20	1.00	0.80	0.60	0.40	0.20
One	XXXX	1.80	1.60	1.40	1.20	1.00	0.80	0.60	0.40
Two	XXXX	XXXX	1.80	1.60	1.40	1.20	1.00	0.80	0.60
Three	XXXX	XXXX	XXXX	1.80	1.60	1.40	1.20	1.00	0.80
Four	XXXX	XXXX	XXXX	XXXX	1.80	1.60	1.40	1.20	1.00
Five	XXXX	XXXX	XXXX	XXXX	XXXX	1.80	1.60	1.40	1.20
Six	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	1.80	1.60	1.20
Seven	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	1.80	1.60
Eight	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	1.80

Table 34. The Weight Structure Resulting from the Training Patterns in Table 33. Note that the weights are (lrate) 0.20 multiplied by the number of times both members of the pair are activated at the same time.

The weight structure developed is simply the learning rate (*lrate*) multiplied by the number of times both members of the pair are active at the same time in the training file. This is a direct result of the Hebbian learning scheme used.

2.2 Effect of Learning Rate

The only parameter that affects learning is the learning rate (*lrate*). This parameter determines the size of the weights generated and interacts with the number of training patterns in the training file. For the initial observation there was 18 patterns in the training file with a learning rate of 0.2. With the pattern file held constant, as the lrate was increased or decreased the weights changed accordingly.

Decreasing the lrate to 0.02 has no effect on the network's ability to recite the sequence. However, increasing it to 1.0 caused the network to be unable to decide which element followed Three. This inability probably sprang from the large activation values that would have been generated in the nodes of the sequence. With large values, the competition would have to continue for a very long

period of time before only one node was active. This would cause the network to appear to be caught in an infinite loop, which it did. It is therefore advisable that when further tests are carried out, subsequent to this thesis, weights in the network are constrained.

2.3 Learning a Blocked Sequence

The final type of learning necessary for the network to do, is the learning of a blocked sequence. In this case only members of the same sub-sequence are simultaneously active in the training file. Such a training file would contain three parts. In the first part connections with the sub-sequences and from the sub-sequences to their Sequence Nodes would be trained. In the second part connection back from the Sequence Node to the initial element of its sub-sequence would be trained. In the final part connection between Sequence Nodes would be trained. An example training file is partially illustrated in table 35. The weight structure it generated is listed in Appendix 2.

1	1 0 1 1 0 0 0 0 0 0 0 0	**8**	0 1 0 0 0 0 0 1 1 0 0 0
2	1 0 1 1 1 0 0 0 0 0 0 0	**9**	0 1 0 0 0 0 0 1 1 1 0 0
3	1 0 1 1 1 1 0 0 0 0 0 0	**10**	0 1 0 0 0 0 0 1 1 1 1 0
4	1 0 1 1 1 1 1 0 0 0 0 0	**11**	0 1 0 0 0 0 0 1 1 1 1 1
5	1 0 0 1 1 1 1 0 0 0 0 0	**12**	0 1 0 0 0 0 0 0 1 1 1 1
6	1 0 0 0 1 1 1 0 0 0 0 0	**13**	0 1 0 0 0 0 0 0 0 1 1 1
7	1 0 0 0 0 1 1 0 0 0 0 0	**14**	0 1 0 0 0 0 0 0 0 0 1 1

Table 35. Training Patterns used for Word to Sequence and Word SON weights in Blocked Sequence. The Sequence to Word weights were trained with four patterns in which only Word Node Zero and Sequence Node Zero were active and four patterns in which only Sequence Node One and Word Node Five were active.

Param	Value
decay	0.2
input	1.0
rval	1.0
eps	10
in_eps	3
cut-off	0.05
lrate	0.2

Table 36. Medium Parameter Values. The values determined in Chapters Seven and Nine to represent the medium value for each parameter.

For the recitation of the sequence the parameters were set as described in table 36.

The results were again as expected the network had no difficulty reciting the sequence from Zero, and any entry into the sequence was restricted to block initial elements. That is, if the network was told to begin recitation at Three it began at Zero and if it was told to begin at Seven it began at Five.

2.4 Summary

These simulations demonstrate that the learning system is capable of generating the types of weight structures necessary for the SON/CQ system to recite a sequence correctly regardless of whether the sequence was learned as a whole or in blocks. This makes the SON/CQ system the only system encountered thus far with this capability.

C. Modifications to the RGN for Use with Hierarchical SON

To enable it to function with the hierarchic SON, the RGN described in Chapter 4 must be modified. The new structure is illustrated in figure 36.

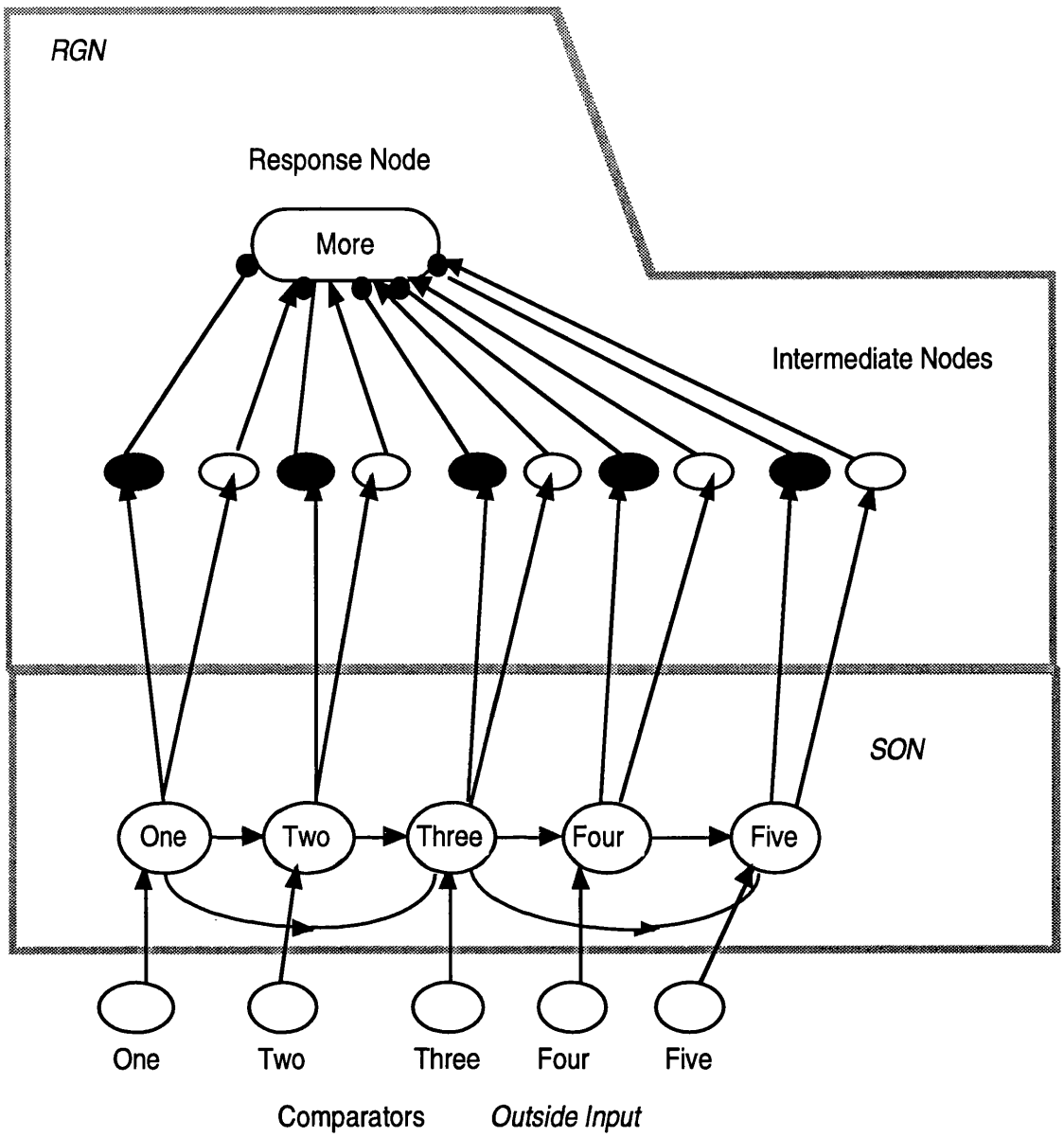


Figure 36. Modified Response Generating Network Attached to SON. Intermediate nodes with positive connections to the response node (Positive Intermediate Nodes) are denoted by clear circles, while intermediate nodes with negative connections to the response node (Negative Intermediate Nodes) are denoted by filled circles.

The response generating network (RGN) is still based on the random walk model as explained in Link (1978) and implemented in Cohen, Dunbar and McClelland (1990). The logistic function is not used to calculate activation in the Intermediate, Comparator or Response nodes. The Response node acts as an evidence accumulator. At the onset of each comparison the Response node's activation is cleared.

On each of the network cycles, a small amount of activation is added to the Response node.

As trials begin with the activation of the two nodes to be compared, and not the node representing the start of the sequence, they are now both called Comparators, instead of calling one the Standard and the other the Comparator.

A comparison trial begins with the activation of the first comparator in epoch one. This activation corresponds to the outside stimulation of the intermediate nodes with negative connections to the response node (Negative Intermediate Nodes). The combination of activation from the SON node and outside activation, lowers the threshold of the Negative Intermediate Node and, correspondingly, the active SON node to zero. These threshold values all begin at 1.0, so no activation travels to the intermediate nodes. The same process occurs on the second epoch for the second comparator, except the intermediate nodes with positive connections to the response node (Positive Intermediate Nodes) are externally stimulated. The result of the first two epochs is that in all following epochs the first comparator inhibits the response node, while the second comparator stimulates it.

The same process occurs with the Sequence Nodes. As the first comparator fires, it passes activation to its Sequence Node at which time the Negative Intermediate Nodes for Sequence are stimulated opening a negative pathway. As the second Comparator passes activation to its Sequence Node the same thing occurs with the Positive Intermediate Nodes for Sequence. Thus, if the two elements are in different subgroups, the Sequence Nodes contribute to the Random Walk as well.

As the two comparators are encountered they send activation to the Sequence Layer. Their activations then decay. On each following epoch, activation is cycled first in the Sequence Layer with a Random Step (RS_i) calculated if the comparators are in different subgroups. Activation is then cycled in the Word Layer and a (RS_i) calculated. This continues until a threshold is reached.

$$a_r(t+1) = a_r(t) + RS_i \qquad \forall \text{ intermediate node}$$

where:

$a_r(t+1)$ = Activation of Response Node at time t+1.

RS_i = Random Step Derived from Intermediate Node Activation.

As in Chapter Four the (RS_i) is randomly distributed with mean (μ_i) and a fixed standard deviation (σ_i). The mean of the distribution (μ_i) is based on the amount of activation passed from the nodes of the SON through the intermediate nodes to the response nodes. Thus, it corresponds to the activation of the first comparator, minus that of the second.

$$RS_i > \mu_i = \left(a_{c1} - a_{c2}\right)$$

where:

RS_i = Random Step from Intermediate node act.

μ_i = Mean of distribution of Random Steps.

a_{c1} = Activation of intermediate node c1.

a_{c2} = Activation of intermediate node c2.

Thus, the intermediate node is noisy, responding with (RS_i) - a more or less corrupted version of the input from the SON nodes (a_{c1} and a_{c2}). These corrupted activation values are evenly distributed around the mean (μ_i). A response is given when the activation of the Response nodes reaches that node's threshold (A). Since there is only one response node the network is restricted to **more** decisions. A second response node could be added for **not more** decisions.

Learning in the SON portion of the network proceeds in the way described in Section B. The weights on the RGN portion of the network are hard-wired, not learned. It may be possible in a future exploration of this system to develop a learning mechanism for this portion of the network.

D. Relationship Between SON System and Relative Order Judgement Effects

It is important to demonstrate that the system described is capable of modelling the effects found in Chapter One to be associated with the relative order decision processes. These effects are listed in table 37. They are limited to SDE, direction, and possibly, Compression effects. Direction effects are thought to arise from a separate network, and therefore need not be modelled here. The remaining effects described in Chapter One appear to be influenced by run-through. It is therefore less important to model them.

Numeric Effects
1. Intuitive 'feel' for size of number
2. Distance (non-linear)
3. Zero-term
4. End-term
5. Compression
Alphabetic Effects
1. Rough position tags for letters
2. SDE (non-linear).
3. Discontinuity in SDE.
4. Direction
5. Direction by SDE Interaction
6. End-term
7. Compression

Table 37. Relative Order Effects to be Modelled.

The type of results obtained with this network are expected to be the same as those in Chapter Four, because the underlying functioning of the system has not been changed by adding the second SON layer. The second set of simulations is designed to demonstrate that the modelling of these effects is not altered when the weight configurations are learned by the network.

1. SDE Simulations

The most important effect to model is the SDE. It is important to determine that the addition of the second layer has not influenced processing to the extent that this effect is no longer present, or that the effectiveness of the model is diminished. The first simulation uses a fully interconnected network, and the second a blocked sequence.

1.1 SDE in a Fully Interconnected Sequence

In this simulation, the output of the final SON/RGN network is compared to Parkman's findings of subject's RTs to numeric stimuli. Since the comparison is to numeric stimuli, a full interconnected network is used. The weight structure illustrated in table 26 was used to approximate Parkman's data in Chapter Three, and was also used in Chapters Four and Seven. It is used again here. The network is expected to perform to the same standard as in Chapter Four.

The free network parameters were set as illustrated in table 38. The fixed parameters are standard deviation of Random Walk (0.1), and the rate of cascade in the SON nodes (0.07). Each comparison was done eighty-two times, although this is slightly higher than the number of comparisons Parkman did at each separation value.

Param	Value
decay	0.2
input	1.0
eps	10
in_eps	3
Start of RW	0.0
Threshold of RW	1.0

Table 38. Medium Parameter Values. The values determined in Chapters Seven and Nine to represent the medium value for each parameter.

The results obtained in this simulation are graphed along with an approximation of Parkman's findings in figure 37. Although the graphs are not directly overlaid in the figure, the correlation between the number of epochs to decision at each separation value, and the approximate RT is very high (Pearson's r = 0.974). This indicates that the network is still modelling SDE slightly better than in Chapter Four, where r = 0.95.

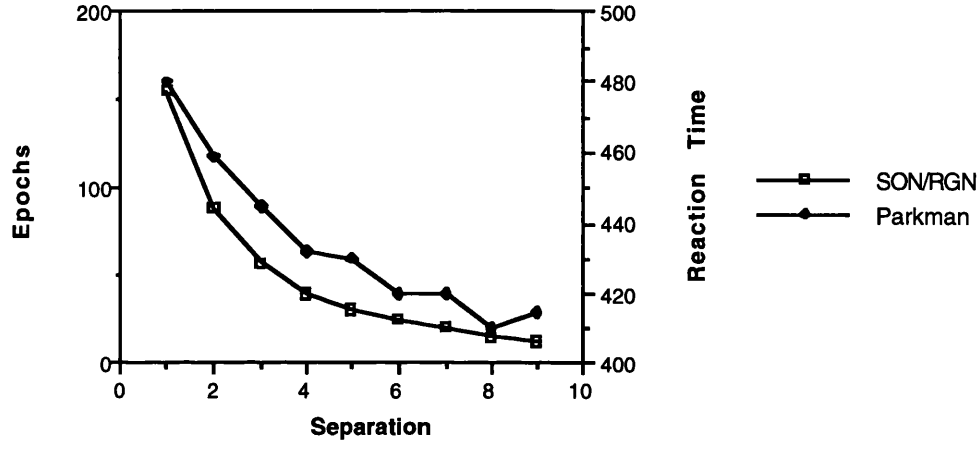


Figure 37. Comparison of Results of Final Network to an Approximation of Parkman's Findings. Although the graphs are not perfectly, overlaid the correlation between the RT and Epochs to solution is very high (Pearson's r = 0.974).

1.2 SDE in a Blocked Sequence

It was expected that the blocked sequence would produce an SDE curve roughly equivalent to the fully interconnected sequence. Some differences were expected due to the added input of the steps taken from the Sequence Layer Nodes.

The network used was the same as in simulation 1.1. The learning rate was set to 0.2. Each comparison was done fifty times.

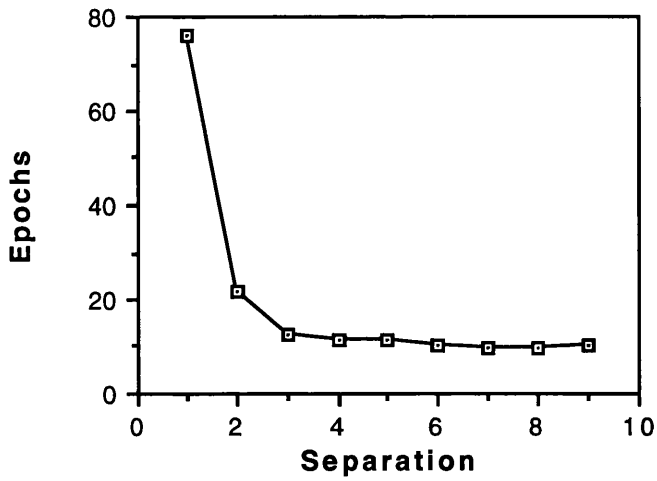


Figure 38. SDE Curve Obtained from a Network with Pre-set Weights Designed to Mimic a Blocked Sequence Training Regime. Note that this curve is steeper at the beginning and flatter at the end than the curve obtained in simulation 1.1.

The SDE produced by the network was steeper at the beginning and flatter at the end, than the curve produced in simulation 1.1. The curve is illustrated in figure 38. The cause of these differences is the additional random walk steps generated by the Sequence Nodes. These steps only occur across subgroup boundaries. For separation 1 very few comparisons are across subgroups. These comparisons generate results in a small number of epochs, but are swamped by comparisons within subgroups which require many more epochs for a decision to be made. Table 39 illustrates this point.

The SDE curve flattens out after separation 3 because the results of all comparisons after this point are due mostly to the steps generated in the Sequence Layer.

Standard	Compar.	Epochs
0	1	16.48
1	2	127.32
2	3	140.46
3	**4**	**12.14**
4	5	17.00
5	6	143.87
6	**7**	**13.58**
7	8	16.72
8	9	194.57
	Average	**75.79**

Table 39. Epochs to Solution for Each Comparison at Separation 1 in a Blocked Sequence. Comparison which cross subgroup boundaries are highlighted. These produce quicker responses than comparisons within subgroups.

1.3 Summary

This set of simulations has demonstrated that the altered structure of the network has not removed its ability to generate SDE. Nor is this ability removed by using a network whose weights are designed to mimic a sequence learned using a blocked training regime. The following simulations investigate the effect of learning the weights from a training file instead of pre-setting them.

2. Effect of Learning on the Modelling of SDE

This set of simulations is used to demonstrate that the network is capable of modelling SDE using learned weights. Since the training regimes have not been designed to specifically mimic recognised training schemes, the results are not compared to subjects' data. They are simply graphed to reflect the network's ability to generate SDEs.

The first simulation is used to establish that the fully interconnected training set from simulation B.2.1 is capable of producing SDE. The second explores the effect of learning rate on the fully interconnected sequence model. The third demonstrates the effect produced using the blocked training regime from simulation B.2.3.

2.1 SDE for Learned Fully Interconnected Sequence

The network parameters and training file used were the same as in simulation B.2.1. All comparisons were made fifty times. The random walk parameters were set as in simulation C.1.1.

It is interesting to note that the Compression effect is quite pronounced in this simulation. The results of comparisons at separation 1, illustrated in table 40, demonstrate this effect dramatically.

Standard	Compar.	Epochs
0	1	15.17
1	2	19.44
2	3	26.52
3	4	38.16
4	5	50.60
5	6	71.48
6	7	108.71
7	8	178.40
8	9	204.49
	Average	**79.22**

Table 40. Comparisons at Separation 1. Note the dramatic increase in Epochs to solution as the pairs originate nearer the high end of the sequence.

Although the Compression effect is found in relative order decisions, the extent demonstrated here was unexpected. It appears to stem from the large amount of activation received by node Zero from the Sequence Node, and then passed to its adjacent nodes. This large amount of activation leads to large differences in activation

between the nodes at the beginning of the sequence, leading to fewer epochs to solution. It also leads to a larger number of errors, as steps toward the wrong threshold are also large. This effect may also be interpreted as an effect of Minimum Stimulus as it is found across all separation values.

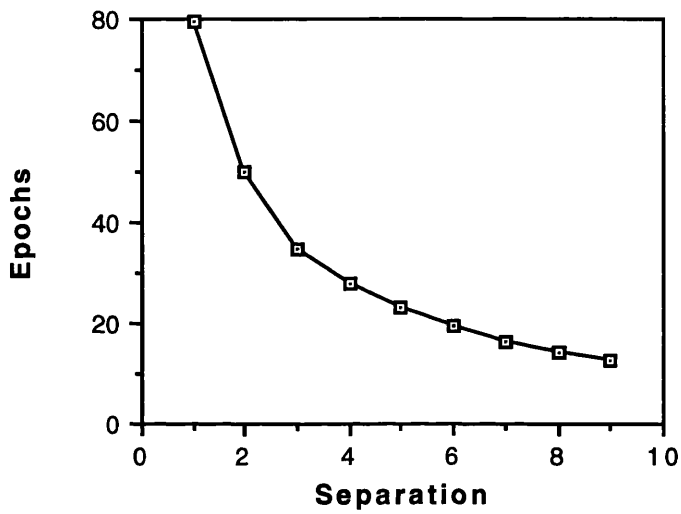


Figure 39. SDE Found with Fully Interconnected Sequence Training Regime.

Minimum	Sep 1	Sep 2
0	15.17	13.34
1	19.44	16.00
2	26.52	22.08
3	38.16	30.92
4	50.60	43.24
5	71.48	61.33
6	108.71	86.40
7	178.40	128.24
Average	**79.22**	**50.19**

Table 41. Reduction in Epochs for Each Minimum Stimulus Across Separation 1 and 2. Note that there is an effect of minimum stimulus or compress as well as an effect of separation.

The main effect of interest is the SDE, as displayed in figure 39. Despite the fact that the large Compression effect may indicate the Minimum Stimulus is having an influence on the reduction in the number of epochs at each separation value, the raw data shows that there is a reduction for each Minimum Stimulus across the separation values. The amount of reduction increases as the Minimum Stimulus increases. An indication of the interaction between Minimum Stimulus and separation is given in table 41.

2.2 Effect of Learning Rate on SDE

The only parameter that affects the results of learning is the learning rate. It is therefore essential to ascertain how changing the learning rate influences the network's ability to demonstrate SDE. As only the magnitude of the weights changes due to changes in learning rate, it is expected that the network's ability to model SDE will remain unchanged.

The network used was identical to that in simulation C.2.1, except that the learning rate was altered across trials, with each comparison carried out fifty times. The learning rates used were low (0.02), medium (0.2) and high (1.0).

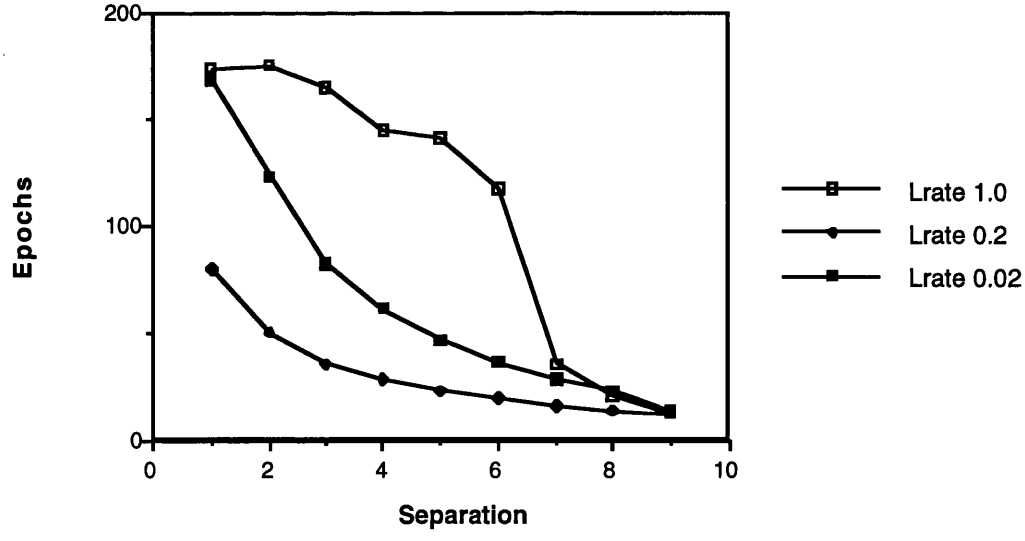


Figure 40. Change in SDE Curve Across Learning Rates. Note that the curves for learning rates of 1.0 and 0.02 are more closely related than curves for learning rates of 1.0 and 0.2.

Figure 40 illustrates how the output of the network changed across learning rates. It is interesting to note that the nature of the curve remains the same from a learning rate of 0.02 to a learning rate of 0.2, except that the slope of the curve decreases. Also, the output with a learning rate of 1.0 appears more closely related to the output with a learning rate of 0.02 than to the output with a learning rate of 0.2.

The most remarkable result of this set of simulations is that the Minimum Stimulus effect increases and the SDE decreases as the learning rate changes from 0.2 to 0.02 to 1.0. At a learning rate of 1.0 the SDE has all but disappeared, with the Minimum Stimulus effect responsible for the curve in figure 40.

This set of simulations forces a rethink of the position of the SDE-Minimum Stimulus interaction. It is not necessary to postulate one influence or the other. It may be that the two effects operate in tandem, with some of the Minimum Stimulus effect interpreted as Compression.

2.3 SDE in Learned Blocked Sequence

The learned weight structure from a blocked sequence is expected to produce SDE flatter than that of the fully interconnected sequence, due to the equivalent activation values for elements at the same positions in different subgroups. This is the result found in simulation 1.2.

In this simulation the training file contained two subgroups instead of the three in the pre-set blocked sequence condition. This is the same training file used in simulation B.2.3. This allows a less complicated training file to be used. The parameters in the network were set as described in simulation 1.1, with a learning rate of 0.2. Different learning rates were not investigated as they are expected to have the same result on individual blocks as they do on the fully interconnected sequence. Therefore, exploring them would be uninformative.

The results of this simulation are illustrated in figure 41. An interesting finding was that the network was unable to make the correct decision in comparisons between elements Two, Three and Four when compared to element Five. The small values of weights in the Sequence Layer led to small activation values on this layer. These were then incapable of generating steps large enough to offset the negative steps which occurred when comparisons were made between the final three elements of the first block and the first element of the second block (e.g. Four to Five, Three to Five, and Two to Five). Negative steps are generated because the activation of node Five decays due to lack of input from previous nodes, thus it becomes lower than the activations of nodes Two, Three and Four. This may explain why at some levels of learning, comparisons over block boundaries may to cause subjects to run-through more often.

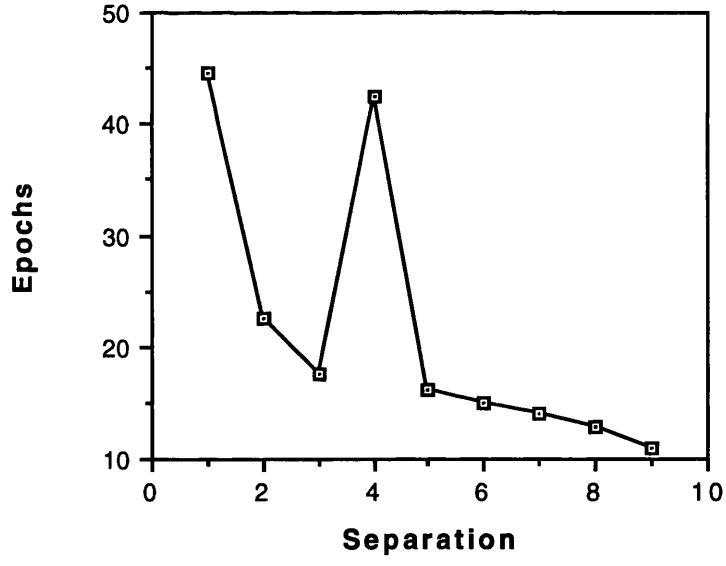


Figure 41. SDE Curve for Network Trained on a Blocked Sequence. The jump at separation 4 is due to the network's ability to begin making decisions involving element Five at this point.

In figure 41 it can be seen that because the difference in activation between node Five and node One is slightly positive, the network is able to make the correct decision but takes a large number of epochs, causing a jump in the SDE curve at separation 4. This may indicate why jumps are found in subject's SDE curves.

2.4 Summary

These simulations show that the network is capable of generating reasonable SDE curves from weights learned using very simple training regimes. They demonstrate the SDE-Minimum Stimulus interaction may not be a straight forward case of one or the other, and that these effects are also tied up in what is called the Compression Effect. It would be informative in future experiments to use more complex training regimes, noting the effect on the SDE, Minimum Stimulus and the Compression interaction which occurs in the network.

3. Remaining Effects

Among the remaining effects to be modelled, the 'feel' for the size of the number, or the rough position tags for the letters may stem from the Sequence nodes of the SON. If these nodes were labelled, and the subject were aware of their existence, they could be used for this purpose. Zero-term and end-term effects could also stem from the use of the Sequence nodes. If a specific node developed which led directly to either the first or the last element of the sequence, comparisons with these elements would benefit from sequence level steps as well as word level steps in the random walk.

The usefulness of modelling the compression effect is not apparent. This effect is complicated by the interaction between minimum stimulus and separation values. Experimenters who report these effects either do not control for SDE, or decide that the compression effect is the more interesting of the two. However, it appears from the literature that the SDE is found more often than the other effects. This, combined with the fact that in Chapter 5 compression effects were not found, indicates that modelling them is not important.

The discontinuity found for alphabet judgements and number judgements made under very strict time constraints, may be due to larger connections between elements at separation one, than for

any other separation value. With relatively little activation in the system, either from the scarcity of connections postulated for the alphabet or the time constraint, it may be possible for these weights to influence the output of the network. Thus far in the simulations, this effect has not been modelled.

The two layer structure in combination with sparse connections may explain the direction effects found with alphabetic stimuli. If the speed of decisions increases with practice, backward decisions may be expected to be slower overall. The interaction with SDE may originate from comparisons within a block. In which case the subject may have to run-through, thus necessitating the co-ordination of forward recitation with a backward comparison.

Slower alphabetic than numeric comparisons, is thought to be due to the difference in the number and strength of connections in two networks. The alphabet is believed to be still in the stage where subgroups are highly salient, causing comparisons within a subgroup to suffer from very little activation, with all subgroups (except the first) having to wait for activation to come through the Sequence nodes. This lack of activation should cause alphabetic comparisons to be slower than numeric.

E. Relationship Between SON System and Developmental Effects

Based on the extended SON defined here and in the previous chapters of this thesis, it is possible to speculate on how the various levels of sequence and random order judgement development may be modelled by the finished system. These speculations would need to be tested in further research either by devising a training scheme and testing the model against humans on a novel sequence, or by determining the training scheme used in either alphabet or number series learning, reproducing it, and looking for novel predictions.

The Wagner and Walters findings can be accounted for by the existence of the SON system and its potential to differentiate

items. This, combined with the restricted attentional abilities of very young children, may combine to restrict differentiation to 1, 2, 3 and 4. The tendency to list exhaustion may be a by-product of the necessity to exhaust the Queue before continuing with sequence recitation.

The weight changes in the system may produce Fuson's levels of sequence development depending on the training scheme used. This would necessarily begin after the String level, as it is essential that the elements are recognised as distinct for them to be activated individually.

The Unbreakable Chain level would develop if the connections within sub-sequences and between the sub-sequence and its sequence nodes were trained with either insufficient experience to connect the sequence nodes to the start of their sub-sequences, or to connect sequence nodes to each other beyond the connection of a Start node to the initial sequence or word node. The Breakable Chain level would develop as the missing connections are learned.

The Numerable Chain level would be entered as each word node developed its own sequence node. This would allow entry at any element of the sequence with continuation from there, as long as the appropriate connections between elements were also learned.

The final, Bi-directional Chain, level is another matter. Originally, in chapter four, the hypothesis was presented that two separate SON systems would be required for backward recitation of the sequence. With the extension of the SON system to include the CQ, it may now be possible to recite either forward or backward from the same network. This could be achieved by employing a second CQ network which did not subtract the activation values from one before adding them to the Queue. In this way the most active node would be the first produced. The co-ordination of this second CQ may result in the effects found when subjects are asked to produce the alphabet backwards. Specifically, that the portion in question is generally recited in the forward direction first. This may be due to the stronger connection to the forward recitation CQ, and the

need to cycle activation around the network which may only start at the beginning of the block of interest. However, this requires that the connections to the CQ network are learned, and not fixed as they appear in the model to date.

The system described is capable of incremental learning in two ways: new Sequence node or new Word nodes can be added. Neither of these additions would cause the network to overwrite previous material.

F. Conclusions

The model presented in this chapter offers the first combination of long-term sequence storage, recitation and relative order judgement effects. It remains to test the speculations above about how the model may be able to explain sequence learning and relative order effects. It would also be necessary to test the robustness of these effects in the model to changes in the free parameters. A final test of interest would be to train the model on a human training regime and examine its functioning to establish the viability of the model.

Chapter Ten - Summary

The purpose of this thesis was to develop a model of long-term sequence learning, representation and relative order judgements.

This thesis began with an overview of the effects encountered when subjects are asked to make relative order judgements with numeric, alphabetic or other ordered stimuli, demonstrating that the same types of phenomena are found with diverse stimuli. The theories and models generated in response to these effects were then outlined. The purpose of this was to develop the background knowledge necessary to generate a new model designed to incorporate more of the effects detailed.

I went on to examine the implications on the nature of the structure of a possible sequence learning and representation system based on the neuropsychological findings. This information was used to determine that, although numbers appear to be unlike other sequences in the amount of information attached to them, it would not be necessary to develop a model which could account for all of this information. The neuropsychological findings indicated that recitation and relative order judgements were carried out in systems separate from those used for calculation.

An attempt was made to define a basic component of the new model, the SON. This component was tested and appeared to be capable of generating the fundamental effect associated with relative order judgements, the SDE. A response mechanism was added to the SON and the resulting network was compared to a mathematical model of RTs associated with relative order judgements, the Poltrock (1989) Random Walk model. It was found that the SON/RGN out performed the Random Walk model.

Next, the effects associated with recitation and learning of a sequence were outlined, and the theories and models developed in response to them were described. This was necessary to aid in development of the network. The network must be capable of learning and reciting the sequence, with all of the associated

effects. A decision was made to explore new ideas which were thought to enable the classical sequence learning models to generate the effects of interest. This turned out to be too optimistic a claim. So, a recitation mechanism developed by Houghton (1990), was added to the SON and the resulting system tested for its ability to generate the effects of interest. The results of these tests indicated that the SON/CQ system was promising, but still lacked the ability to model hierarchical sequences. It also lacked a learning mechanism, with provision for true incremental learning.

The effects to be generated by a successful learning system were described, and an addition to the SON/CQ system was outlined which would allow it to learn hierarchical sequences . The resulting system was tested to determine if it still produced the effects associated with relative order judgements. On successful completion of these tests, the learning mechanism first mentioned in Chapter Three was used with the system. Tests found that this learning mechanism enabled the system to develop representations of both hierarchical (blocked) sequences and non-hierarchical (fully interconnected) sequences. The resulting representations were then shown to continue to demonstrate the basic SDE and Compression effects associated with relative order judgements. An indication was given as to how the system could be used to generate the remaining relative order judgement and developmental effects.

Future research in this area, and on this system of networks would be beneficial. Neuropsychological research into the relationships between the different types of sequences, particularly the alphabet and number series, may indicate if these sequences could share a common learning and representation mechanism. They may also indicate if a common underlying representation of sequence (i.e. a "number line") is generated, or if the natural result of the sequence learning architecture is a "number line" for each sequence. Investigation of the system of networks developed here should include an attempt to train it on a training regime designed to mimic regimes used by human subjects. This may allow the

model to generate novel predictions about the course learning might take and/or the effects to be found when the subject is asked to make judgements about that sequence. Finally, this model would benefit from the addition of a mechanism which uses run-through to solve relative order judgements. Such a mechanism would complete the model.

Appendix 1

Files used with the Rumelhart and McClelland (1986) simulator to test SRN for their ability to learn sequences incrementally (Chapter Six).

Name	Input	Recurrence	Output
aee	1 1 0 0 0 0 0 0	0.5 0.5 0.5 0.5 0.5	1 1 0 0 0 0 0 0
bee	0 1 1 0 0 0 0 0	0.5 0.5 0.5 0.5 0.5	0 1 1 0 0 0 0 0
cee	0 0 1 1 0 0 0 0	0.5 0.5 0.5 0.5 0.5	0 0 1 1 0 0 0 0
dee	0 0 0 1 1 0 0 0	0.5 0.5 0.5 0.5 0.5	0 0 0 1 1 0 0 0
eee	0 0 0 0 1 1 0 0	0.5 0.5 0.5 0.5 0.5	0 0 0 0 1 1 0 0
eef	0 0 0 0 0 1 1 0	0.5 0.5 0.5 0.5 0.5	0 0 0 0 0 1 1 0
gee	0 0 0 0 0 0 1 1	0.5 0.5 0.5 0.5 0.5	0 0 0 0 0 0 1 1
haitch	1 0 1 0 0 0 0 0	0.5 0.5 0.5 0.5 0.5	1 0 1 0 0 0 0 0
aie	0 1 0 1 0 0 0 0	0.5 0.5 0.5 0.5 0.5	0 1 0 1 0 0 0 0
jay	0 0 1 0 1 0 0 0	0.5 0.5 0.5 0.5 0.5	0 0 1 0 1 0 0 0
kay	0 0 0 1 0 1 0 0	0.5 0.5 0.5 0.5 0.5	0 0 0 1 0 1 0 0
eel	0 0 0 0 1 0 1 0	0.5 0.5 0.5 0.5 0.5	0 0 0 0 1 0 1 0
eem	0 0 0 0 0 1 0 1	0.5 0.5 0.5 0.5 0.5	0 0 0 0 0 1 0 1
een	1 0 0 1 0 0 0 0	0.5 0.5 0.5 0.5 0.5	1 0 0 1 0 0 0 0
oow	0 1 0 0 1 0 0 0	0.5 0.5 0.5 0.5 0.5	0 1 0 0 1 0 0 0
pea	0 0 1 0 0 1 0 0	0.5 0.5 0.5 0.5 0.5	0 0 1 0 0 1 0 0
que	0 0 0 1 0 0 1 0	0.5 0.5 0.5 0.5 0.5	0 0 0 1 0 0 1 0
are	0 0 0 0 1 0 0 1	0.5 0.5 0.5 0.5 0.5	0 0 0 0 1 0 0 1
ees	1 0 0 0 1 0 0 0	0.5 0.5 0.5 0.5 0.5	1 0 0 0 1 0 0 0
tea	0 1 0 0 0 1 0 0	0.5 0.5 0.5 0.5 0.5	0 1 0 0 0 1 0 0
you	0 0 1 0 0 0 1 0	0.5 0.5 0.5 0.5 0.5	0 0 1 0 0 0 1 0
vee	0 0 0 1 0 0 0 1	0.5 0.5 0.5 0.5 0.5	0 0 0 1 0 0 0 1
dbl-u	1 0 0 0 0 1 0 0	0.5 0.5 0.5 0.5 0.5	1 0 0 0 0 1 0 0
eex	0 1 0 0 0 0 1 0	0.5 0.5 0.5 0.5 0.5	0 1 0 0 0 0 1 0
why	0 0 1 0 0 0 0 1	0.5 0.5 0.5 0.5 0.5	0 0 1 0 0 0 0 1
zee	1 0 0 0 0 0 1 0	0.5 0.5 0.5 0.5 0.5	1 0 0 0 0 0 1 0

Table 1-1. Pretraining File without Recurrence (pretrain.pat). The input, recurrence, and training patterns used. A recurrence value of 0.5 indicates that no recurrence was available to the network.

Name	Input	Recurrence	Output
start	1 1 1 1 0 0 0 0	0.5 0.5 0.5 0.5 0.5	1 1 0 0 0 0 0 0
aee	1 1 0 0 0 0 0 0	-13 -14 -15 -16 -17	0 1 1 0 0 0 0 0
bee	0 1 1 0 0 0 0 0	-13 -14 -15 -16 -17	0 0 1 1 0 0 0 0
cee	0 0 1 1 0 0 0 0	-13 -14 -15 -16 -17	0 0 0 1 1 0 0 0
dee	0 0 0 1 1 0 0 0	-13 -14 -15 -16 -17	0 0 0 0 1 1 0 0
eee	0 0 0 0 1 1 0 0	-13 -14 -15 -16 -17	0 0 0 0 0 1 1 0
eef	0 0 0 0 0 1 1 0	-13 -14 -15 -16 -17	0 0 0 0 0 0 1 1
gee	0 0 0 0 0 0 1 1	-13 -14 -15 -16 -17	1 0 1 0 0 0 0 0
haitch	1 0 1 0 0 0 0 0	-13 -14 -15 -16 -17	0 1 0 1 0 0 0 0
aie	0 1 0 1 0 0 0 0	-13 -14 -15 -16 -17	0 0 1 0 1 0 0 0
jay	0 0 1 0 1 0 0 0	-13 -14 -15 -16 -17	0 0 0 1 0 1 0 0
kay	0 0 0 1 0 1 0 0	-13 -14 -15 -16 -17	0 0 0 0 1 0 1 0
eel	0 0 0 0 1 0 1 0	-13 -14 -15 -16 -17	0 0 0 0 0 1 0 1
eem	0 0 0 0 0 1 0 1	-13 -14 -15 -16 -17	1 0 0 1 0 0 0 0
een	1 0 0 1 0 0 0 0	-13 -14 -15 -16 -17	0 1 0 0 1 0 0 0
oow	0 1 0 0 1 0 0 0	-13 -14 -15 -16 -17	0 0 1 0 0 1 0 0
pea	0 0 1 0 0 1 0 0	-13 -14 -15 -16 -17	0 0 0 1 0 0 1 0
que	0 0 0 1 0 0 1 0	-13 -14 -15 -16 -17	0 0 0 0 1 0 0 1
are	0 0 0 0 1 0 0 1	-13 -14 -15 -16 -17	1 0 0 0 1 0 0 0
ees	1 0 0 0 1 0 0 0	-13 -14 -15 -16 -17	0 1 0 0 0 1 0 0
tea	0 1 0 0 0 1 0 0	-13 -14 -15 -16 -17	0 0 1 0 0 0 1 0
you	0 0 1 0 0 0 1 0	-13 -14 -15 -16 -17	0 0 0 1 0 0 0 1
vee	0 0 0 1 0 0 0 1	-13 -14 -15 -16 -17	1 0 0 0 0 1 0 0
dbl-u	1 0 0 0 0 1 0 0	-13 -14 -15 -16 -17	0 1 0 0 0 0 1 0
eex	0 1 0 0 0 0 1 0	-13 -14 -15 -16 -17	0 0 1 0 0 0 0 1
why	0 0 1 0 0 0 0 1	-13 -14 -15 -16 -17	1 0 0 0 0 0 1 0
zee	1 0 0 0 0 0 1 0	-13 -14 -15 -16 -17	0 0 0 0 0 0 0 0

Table 1-2. Fully Recurrent Auto-Associative Training File (entire.pat). The input, recurrence and output patterns used. A recurrence value of 0.5 indicates that no recurrence was available to the network. Recurrence values which are negative numbers indicate that the value of that unit on the previous epoch is used as the present input value. Notice that input and output patterns are the same.

Name	Input	Recurrence	Output
start	1 1 1 1 0 0 0 0	0.5 0.5 0.5 0.5 0.5	1 1 0 0 0 0 0 0
aee	1 1 0 0 0 0 0 0	-13 -14 -15 -16 -17	0 1 1 0 0 0 0 0
bee	0 1 1 0 0 0 0 0	-13 -14 -15 -16 -17	0 0 1 1 0 0 0 0
cee	0 0 1 1 0 0 0 0	-13 -14 -15 -16 -17	0 0 0 1 1 0 0 0
dee	0 0 0 1 1 0 0 0	-13 -14 -15 -16 -17	0 0 0 0 1 1 0 0
eee	0 0 0 0 1 1 0 0	-13 -14 -15 -16 -17	0 0 0 0 0 1 1 0
eef	0 0 0 0 0 1 1 0	-13 -14 -15 -16 -17	0 0 0 0 0 0 1 1
gee	0 0 0 0 0 0 1 1	-13 -14 -15 -16 -17	1 0 1 0 0 0 0 0
haitch	1 0 1 0 0 0 0 0	0.5 0.5 0.5 0.5 0.5	0 1 0 1 0 0 0 0
aie	0 1 0 1 0 0 0 0	-13 -14 -15 -16 -17	0 0 1 0 1 0 0 0
jay	0 0 1 0 1 0 0 0	-13 -14 -15 -16 -17	0 0 0 1 0 1 0 0
kay	0 0 0 1 0 1 0 0	-13 -14 -15 -16 -17	0 0 0 0 1 0 1 0
eel	0 0 0 0 1 0 1 0	-13 -14 -15 -16 -17	0 0 0 0 0 1 0 1
eem	0 0 0 0 0 1 0 1	-13 -14 -15 -16 -17	1 0 0 1 0 0 0 0
een	1 0 0 1 0 0 0 0	-13 -14 -15 -16 -17	0 1 0 0 1 0 0 0
oow	0 1 0 0 1 0 0 0	-13 -14 -15 -16 -17	0 0 1 0 0 1 0 0
pea	0 0 1 0 0 1 0 0	-13 -14 -15 -16 -17	0 0 0 1 0 0 1 0
que	0 0 0 1 0 0 1 0	0.5 0.5 0.5 0.5 0.5	0 0 0 0 1 0 0 1
are	0 0 0 0 1 0 0 1	-13 -14 -15 -16 -17	1 0 0 0 1 0 0 0
ees	1 0 0 0 1 0 0 0	-13 -14 -15 -16 -17	0 1 0 0 0 1 0 0
tea	0 1 0 0 0 1 0 0	-13 -14 -15 -16 -17	0 0 1 0 0 0 1 0
you	0 0 1 0 0 0 1 0	-13 -14 -15 -16 -17	0 0 0 1 0 0 0 1
vee	0 0 0 1 0 0 0 0	-13 -14 -15 -16 -17	1 0 0 0 0 1 0 0
dbl-u	1 0 0 0 0 1 0 0	0.5 0.5 0.5 0.5 0.5	0 1 0 0 0 0 1 0
eex	0 1 0 0 0 0 1 0	-13 -14 -15 -16 -17	0 0 1 0 0 0 0 1
why	0 0 1 0 0 0 0 1	-13 -14 -15 -16 -17	1 0 0 0 0 0 1 0
zee	1 0 0 0 0 0 1 0	-13 -14 -15 -16 -17	0 0 0 0 0 0 0 0

Table 1-3. Semi-recurrent Training File (entire2.pat). This file represents the general block structure of the sequence. The input, recurrence and output patterns used. A recurrence value of 0.5 indicates that no recurrence was available to the network. Recurrence values which are negative numbers indicate that the value of that unit on the previous epoch is used as the present input value.

Name	Input	Recurrence	Output
start	1 1 1 1 0 0 0 0	0.5 0.5 0.5 0.5 0.5	1 1 0 0 0 0 0 0
aee	1 1 0 0 0 0 0 0	0.5 0.5 0.5 0.5 0.5	0 1 1 0 0 0 0 0
bee	0 1 1 0 0 0 0 0	0.5 0.5 0.5 0.5 0.5	0 0 1 1 0 0 0 0
cee	0 0 1 1 0 0 0 0	0.5 0.5 0.5 0.5 0.5	0 0 0 1 1 0 0 0
dee	0 0 0 1 1 0 0 0	0.5 0.5 0.5 0.5 0.5	0 0 0 0 1 1 0 0
eee	0 0 0 0 1 1 0 0	-13 -14 -15 -16 -17	0 0 0 0 0 1 1 0
eef	0 0 0 0 0 1 1 0	-13 -14 -15 -16 -17	0 0 0 0 0 0 1 1
gee	0 0 0 0 0 0 1 1	-13 -14 -15 -16 -17	1 0 1 0 0 0 0 0
haitch	1 0 1 0 0 0 0 0	0.5 0.5 0.5 0.5 0.5	0 1 0 1 0 0 0 0
aie	0 1 0 1 0 0 0 0	-13 -14 -15 -16 -17	0 0 1 0 1 0 0 0
jay	0 0 1 0 1 0 0 0	-13 -14 -15 -16 -17	0 0 0 1 0 1 0 0
kay	0 0 0 1 0 1 0 0	-13 -14 -15 -16 -17	0 0 0 0 1 0 1 0
eel	0 0 0 0 1 0 1 0	0.5 0.5 0.5 0.5 0.5	0 0 0 0 0 1 0 1
eem	0 0 0 0 0 1 0 1	-13 -14 -15 -16 -17	1 0 0 1 0 0 0 0
een	1 0 0 1 0 0 0 0	-13 -14 -15 -16 -17	0 1 0 0 1 0 0 0
oow	0 1 0 0 1 0 0 0	-13 -14 -15 -16 -17	0 0 1 0 0 1 0 0
pea	0 0 1 0 0 1 0 0	-13 -14 -15 -16 -17	0 0 0 1 0 0 1 0
que	0 0 0 1 0 0 1 0	0.5 0.5 0.5 0.5 0.5	0 0 0 0 1 0 0 1
are	0 0 0 0 1 0 0 1	-13 -14 -15 -16 -17	1 0 0 0 1 0 0 0
ees	1 0 0 0 1 0 0 0	-13 -14 -15 -16 -17	0 1 0 0 0 1 0 0
tea	0 1 0 0 0 1 0 0	0.5 0.5 0.5 0.5 0.5	0 0 1 0 0 0 1 0
you	0 0 1 0 0 0 1 0	-13 -14 -15 -16 -17	0 0 0 1 0 0 0 1
vee	0 0 0 1 0 0 0 1	-13 -14 -15 -16 -17	1 0 0 0 0 1 0 0
dbl-u	1 0 0 0 0 1 0 0	0.5 0.5 0.5 0.5 0.5	0 1 0 0 0 0 1 0
eex	0 1 0 0 0 0 1 0	0.5 0.5 0.5 0.5 0.5	0 0 1 0 0 0 0 1
why	0 0 1 0 0 0 0 1	0.5 0.5 0.5 0.5 0.5	1 0 0 0 0 0 1 0
zee	1 0 0 0 0 0 1 0	0.5 0.5 0.5 0.5 0.5	0 0 0 0 0 0 0 0

Table 1-4. Semi-recurrent Training File (entire3.pat). This file represents the general block structure with the beginning and end of the sequence well learned. The input, recurrence and output patterns used. A recurrence value of 0.5 indicates that no recurrence was available to the network. Recurrence values which are negative numbers indicate that the value of that unit on the previous epoch is used as the present input value.

Name	Input	Recurrence	Output
start	1 1 1 1 0 0 0 0	0.5 0.5 0.5 0.5 0.5	1 1 0 0 0 0 0 0
aee	1 1 0 0 0 0 0 0	0.5 0.5 0.5 0.5 0.5	0 1 1 0 0 0 0 0
bee	0 1 1 0 0 0 0 0	0.5 0.5 0.5 0.5 0.5	0 0 1 1 0 0 0 0
cee	0 0 1 1 0 0 0 0	0.5 0.5 0.5 0.5 0.5	0 0 0 1 1 0 0 0
dee	0 0 0 1 1 0 0 0	0.5 0.5 0.5 0.5 0.5	0 0 0 0 1 1 0 0
eee	0 0 0 0 1 1 0 0	0.5 0.5 0.5 0.5 0.5	0 0 0 0 0 1 1 0
eef	0 0 0 0 0 1 1 0	0.5 0.5 0.5 0.5 0.5	0 0 0 0 0 0 1 1
gee	0 0 0 0 0 0 1 1	0.5 0.5 0.5 0.5 0.5	1 0 1 0 0 0 0 0
haitch	1 0 1 0 0 0 0 0	0.5 0.5 0.5 0.5 0.5	0 1 0 1 0 0 0 0
aie	0 1 0 1 0 0 0 0	0.5 0.5 0.5 0.5 0.5	0 0 1 0 1 0 0 0
jay	0 0 1 0 1 0 0 0	0.5 0.5 0.5 0.5 0.5	0 0 0 1 0 1 0 0
kay	0 0 0 1 0 1 0 0	0.5 0.5 0.5 0.5 0.5	0 0 0 0 1 0 1 0
eel	0 0 0 0 1 0 1 0	0.5 0.5 0.5 0.5 0.5	0 0 0 0 0 1 0 1
eem	0 0 0 0 0 1 0 1	0.5 0.5 0.5 0.5 0.5	1 0 0 1 0 0 0 0
een	1 0 0 1 0 0 0 0	0.5 0.5 0.5 0.5 0.5	0 1 0 0 1 0 0 0
oow	0 1 0 0 1 0 0 0	0.5 0.5 0.5 0.5 0.5	0 0 1 0 0 1 0 0
pea	0 0 1 0 0 1 0 0	0.5 0.5 0.5 0.5 0.5	0 0 0 1 0 0 1 0
que	0 0 0 1 0 0 1 0	0.5 0.5 0.5 0.5 0.5	0 0 0 0 1 0 0 1
are	0 0 0 0 1 0 0 1	0.5 0.5 0.5 0.5 0.5	1 0 0 0 1 0 0 0
ees	1 0 0 0 1 0 0 0	0.5 0.5 0.5 0.5 0.5	0 1 0 0 0 1 0 0
tea	0 1 0 0 0 1 0 0	0.5 0.5 0.5 0.5 0.5	0 0 1 0 0 0 1 0
you	0 0 1 0 0 0 1 0	0.5 0.5 0.5 0.5 0.5	0 0 0 1 0 0 0 1
vee	0 0 0 1 0 0 0 1	0.5 0.5 0.5 0.5 0.5	1 0 0 0 0 1 0 0
dbl-u	1 0 0 0 0 1 0 0	0.5 0.5 0.5 0.5 0.5	0 1 0 0 0 0 1 0
eex	0 1 0 0 0 0 1 0	0.5 0.5 0.5 0.5 0.5	0 0 1 0 0 0 0 1
why	0 0 1 0 0 0 0 1	0.5 0.5 0.5 0.5 0.5	1 0 0 0 0 0 1 0
zee	1 0 0 0 0 0 1 0	0.5 0.5 0.5 0.5 0.5	0 0 0 0 0 0 0 0

Table 1-5. Fully Recurrent Training File (entire4.pat). This file represents the fully learned sequence. The input, recurrence and output patterns used. A recurrence value of 0.5 indicates that no recurrence was available to the network.

Name	Input	Recurrence	Output
start	1 1 1 1 0 0 0 0	0.5 0.5 0.5 0.5 0.5	1 1 0 0 0 0 0 0
aee	1 1 0 0 0 0 0 0	-13 -14 -15 -16 -17	0 1 1 0 0 0 0 0
bee	0 1 1 0 0 0 0 0	-13 -14 -15 -16 -17	0 0 1 1 0 0 0 0
cee	0 0 1 1 0 0 0 0	-13 -14 -15 -16 -17	0 0 0 1 1 0 0 0
dee	0 0 0 1 1 0 0 0	-13 -14 -15 -16 -17	0 0 0 0 1 1 0 0
eee	0 0 0 0 1 1 0 0	-13 -14 -15 -16 -17	0 0 0 0 0 1 1 0
eef	0 0 0 0 0 1 1 0	-13 -14 -15 -16 -17	0 0 0 0 0 0 1 1
gee	0 0 0 0 0 0 1 1	-13 -14 -15 -16 -17	1 0 1 0 0 0 0 0

Table 1-6. First Subsequence Training File (part1.pat). This file represents the first block of the sequence. The input, recurrence and output patterns used. A recurrence value of 0.5 indicates that no recurrence was available to the network. Recurrence values which are negative numbers indicate that the value of that unit on the previous epoch is used as the present input value.

Name	Input	Recurrence	Output
haitch	1 0 1 0 0 0 0 0	0.5 0.5 0.5 0.5 0.5	0 1 0 1 0 0 0 0
aie	0 1 0 1 0 0 0 0	-13 -14 -15 -16 -17	0 0 1 0 1 0 0 0
jay	0 0 1 0 1 0 0 0	-13 -14 -15 -16 -17	0 0 0 1 0 1 0 0
kay	0 0 0 1 0 1 0 0	-13 -14 -15 -16 -17	0 0 0 0 1 0 1 0
eel	0 0 0 0 1 0 1 0	-13 -14 -15 -16 -17	0 0 0 0 0 1 0 1
eem	0 0 0 0 0 1 0 1	-13 -14 -15 -16 -17	1 0 0 1 0 0 0 0
een	1 0 0 1 0 0 0 0	-13 -14 -15 -16 -17	0 1 0 0 1 0 0 0
oow	0 1 0 0 1 0 0 0	-13 -14 -15 -16 -17	0 0 1 0 0 1 0 0
pea	0 0 1 0 0 1 0 0	-13 -14 -15 -16 -17	0 0 0 1 0 0 1 0

Table 1-7. Second Subsequence Training File (part2.pat). This file represents the second block of the sequence. The input, recurrence and output patterns used. A recurrence value of 0.5 indicates that no recurrence was available to the network. Recurrence values which are negative numbers indicate that the value of that unit on the previous epoch is used as the present input value.

Name	Input	Recurrence	Output
que	0 0 0 1 0 0 1 0	0.5 0.5 0.5 0.5 0.5	0 0 0 0 1 0 0 1
are	0 0 0 0 1 0 0 1	-13 -14 -15 -16 -17	1 0 0 0 1 0 0 0
ees	1 0 0 0 1 0 0 0	-13 -14 -15 -16 -17	0 1 0 0 0 1 0 0
tea	0 1 0 0 0 1 0 0	-13 -14 -15 -16 -17	0 0 1 0 0 0 1 0
you	0 0 1 0 0 0 1 0	-13 -14 -15 -16 -17	0 0 0 1 0 0 0 1
vee	0 0 0 1 0 0 0 1	-13 -14 -15 -16 -17	1 0 0 0 0 1 0 0

Table 1-8. Third Subsequence Training File (part3.pat). This file represents the third block of the sequence. The input, recurrence and output patterns used. A recurrence value of 0.5 indicates that no recurrence was available to the network. Recurrence values which are negative numbers indicate that the value of that unit on the previous epoch is used as the present input value.

Name	Input	Recurrence	Output
dbl-u	1 0 0 0 0 1 0 0	0.5 0.5 0.5 0.5 0.5	0 1 0 0 0 0 1 0
eex	0 1 0 0 0 0 1 0	-13 -14 -15 -16 -17	0 0 1 0 0 0 0 1
why	0 0 1 0 0 0 0 1	-13 -14 -15 -16 -17	1 0 0 0 0 0 1 0
zee	1 0 0 0 0 0 1 0	-13 -14 -15 -16 -17	0 0 0 0 0 0 0 0

Table 1-9. Fourth Subsequence Training File (part4.pat). This file represents the final block of the sequence. The input, recurrence and output patterns used. A recurrence value of 0.5 indicates that no recurrence was available to the network. Recurrence values which are negative numbers indicate that the value of that unit on the previous epoch is used as the present input value.

Appendix 2

Results of Simulation B.2.3 (Chapter Nine). The weight structure generated in response to the training patterns displayed and described in Table 34.

To → From ↓	0	1
0	XXXXXX	0.20
1	XXXXXX	XXXXX

Table 2-1. Weights in the Sequence SON Portion of the Network for Block Investigation.

From → To ↓	Zero	One
Zero	0.80	0.00
One	0.00	0.00
Two	0.00	0.00
Three	0.00	0.00
Four	0.00	0.00
Five	0.00	0.80
Six	0.00	0.00
Seven	0.00	0.00
Eight	0.00	0.00
Nine	0.00	0.00

Table 2-2. Weights from Sequence Layer to the Word Layer for Block Investigation.

To → From ↓	Zero	One
Zero	0.80	0.00
One	1.00	0.00
Two	1.00	0.00
Three	1.00	0.00
Four	0.80	0.00
Five	0.00	0.80
Six	0.00	1.00
Seven	0.00	1.00
Eight	0.00	1.00
Nine	0.00	0.80

Table 2-3. Weights from the Word Layer to the Sequence Layer for Block Investigation.

To → From ↓	One	Two	Three	Four	Five	Six	Seven	Eight	Nine
Zero	0.80	0.60	0.40	0.20	0.00	0.00	0.00	0.00	0.00
One	XXXX	0.80	0.60	0.40	0.00	0.00	0.00	0.00	0.00
Two	XXXX	XXXX	0.80	0.60	0.00	0.00	0.00	0.00	0.00
Three	XXXX	XXXX	XXXX	0.80	0.00	0.00	0.00	0.00	0.00
Four	XXXX	XXXX	XXXX	XXXX	0.00	0.00	0.00	0.00	0.00
Five	XXXX	XXXX	XXXX	XXXX	XXXX	0.80	0.60	0.40	0.20
Six	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	0.80	0.60	0.40
Seven	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	0.80	0.60
Eight	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	0.80

Table 2-4. Weights in the Word SON Portion of the Network for Block Investigation.

Appendix 3

```
/*****************************************************  */
/*****************************************************  */
/*                        SON_blc.c                      */
/ *                                                       */
/*  Program designed by Kristina Doing Harris, based on ideas  */
/*  of Prof. A.J. Sanford and herself.  Two of the ideas is taken  */
/*  from Cohen; Dunbar and McClelland, (1990) this is marked  */
/*   with ** in the body of the program.  It also employs the  */
/*  ideas of Houghton (1990), which are marked ##.  This  */
/*   network determines the relative placement of objects  */
/*  learned as a sequence using a Random Walk inspired  */
/*   comparison subprogram, in this particular case the number  */
/*  sequence, in english from 1 to 10 and recites the sequence  */
/*   as learned.                                          */
/ *                                                       */
/*****************************************************  */
/*****************************************************  */

#include "stdio.h"
#include "math.h"
#include "ctype.h"

#define nds 10                    /* Number of nodes in seq */
#define layers 2                  /* Number of layers in net */
#define rate 0.07                 /* Fixed Network Parameters */
#define temperature 0.15027380
#define standDev 0.1              /* Standard Dev. of RW steps */
#define input 1.0


void run_net();
void change_params();
void learn();
```

```
void  read_net();
void  rec_seq();
void   cycle_act_ws();
void  set_queue();
void  comp_queue();
void   batch_net();
void   set_up_net();
void   clear_net();
void   clear_weights();
void   cycle_activation();
float   calc_step();
void   print_act_step();
void   print_result();
void    write_wts_file();
void   pt_act();
void   open_act();


int  nodes[layers];                  /* Number of nodes in each layer0 = seq,1 = wd */
int epoch = 1;                       /* The epoch number used determine ave input*/
int eps = 10;                        /* Number of activation cycles*/
float rval = 1.0;                    /* Value returned to winning node of queue */
float cut_off = 0.05;                /* Minimum act for inclusion in queue */
float  decay = 0.2;                  /* Decay on nodes with no input */
int in_eps = 3;                      /* Epochs activation cycled from S-lay to W-lay */
int  resp[2] = {nds+1,nds+1};        /* Response of the CQ net for the two layers */
int  errors;                         /* Number of errors made by the network  */
int  err_eps;                        /* Total epochs taken on error trials */
int  step_c[4];                      /* Nodes contributing to the RW steps */
float  weight[layers][nds][nds];     /* Weights each node to following on layer */
float ws_wght[nds][nds];             /* Weights from word to sequence nodes */
float sw_wght[nds][nds];             /* Weights from sequence to word nodes */
float  node_act[layers][nds];        /* Activation value for each node I1, I2 */ float
net_inp[layers][nds];                /* Average input to each node at time t */
float  que[layers][nds];             /* activation of nodes in the S[0] & W[1] queues*/
float startWalk = 0.0;               /* Free Network Parameters: Start of RW */
```

```
float threshold = 1.00;              /* Threshold to reach for result */
float step_accum;                    /* Accumulation of steps toward threshold    */
char stop_sign = 'N';                /* Signals the user wishes to end program */
static float z_table[5][10] = {                                    /* Z scores */
{0.00, 0.03, 0.05, 0.08, 0.10, 0.13, 0.15, 0.18, 0.20, 0.23}, /* 50% - 59% */
{0.26,  0.28, 0.31, 0.33, 0.36, 0.39, 0.41, 0.44, 0.47, 0.50}, /* 60% - 69% */
{0.53,  0.56, 0.58, 0.61, 0.65, 0.68, 0.71, 0.74, 0.77, 0.81}, /* 70% - 79% */
{0.84,  0.88, 0.92, 0.96, 1.00, 1.04, 1.08, 1.13, 1.18, 1.23}, /* 80% - 89% */
{1.29,  1.34, 1.41, 1.48, 1.56, 1.65, 1.76, 1.89, 2.06, 2.33}  /* 90% - 99% */
};
FILE *actlog;                        /* File for activations during RW */
FILE *actfile;                       /* File for activations during seq recitation*/


main()
{
  run_net();
  while (stop_sign == 'N')
    run_net();
}


void run_net()
/*  Coordinates the running of the network.  Allowing user to    * /
/*  choose between five options learn weights, read weights,     * /
/*  run net, write weights and Stop */
{
  int ans;                           /* Number of option desired */
  static int tmthru = 1;             /* to set up net */

  if (tmthru == 1)
  {
    clear_net();
    clear_weights();
    tmthru++;
  }
```

```
  puts("\n\nWould you like to:");
  puts("  0 - Change Free Parameters");
  puts("  1 - Learn Weights for net");
          puts("  2 - Read Network Weights from a File");
  puts("  3 - Recite the sequence");
           puts("  4 - Do a Batch Run on a Number Pair");
  puts("  5 - Write weights to a file");
  puts("  6 - Stop");
  scanf("%d",&ans);
  switch  (ans)
  {
    case  0 : change_params();
                 break;
    case  1 : learn();
                 break;
    case  2 : read_net();
                 break;
    case  3 : rec_seq();
                 break;
    case  4 : batch_net();
                 break;
    case   5  :  write_wts_file();
                 break;
    case  6 : puts("\n End of Session");
                 stop_sign = 'y';
                 break;
    default : puts("\nThe answer you have given is invalid. Please
                 re-enter.");  run_net();
  }
}
```

```
void change_params()
/*  Allows the free parameters in the network to be changed.  */
{
  int ans;

  puts("\nChange ");
  printf("\n    1 - starting point (default %1.4f)",startWalk);
  printf("\n    2 - threshold (default %1.4f)",threshold);
  printf("\n    3 - epochs of activation (default %d)",eps);
  printf("\n    4 - inhibition returned to queue (default
                                                    %1.4f)",rval);
  printf("\n    5 - threshold for queue entry (default
                                                     %1.4f)",cut_off);
  printf("\n    6 - epochs of outside activation (default
                                                    %d)",in_eps);
  printf("\n    7 - decay (default %1.2f)",decay);
  printf("\n    8 - done\n");
  scanf("%d",&ans);
  switch (ans)
  {
    case 1  : puts("\nNew starting point : ");
              scanf("%f",&startWalk);
              change_params();
              break;
    case 2  : puts("\nNew threshold value : ");
              scanf("%f",&threshold);
              change_params();
              break;
    case 3  : puts("\nNew epoch value : ");
              scanf("%d",&eps);
              change_params();
              break;
```

```
    case 4  : puts("\nNew return value : ");
              scanf("%f",&rval);
              change_params();
              break;
    case 5  : puts("\nNew cut_off value : ");
              scanf("%f",&cut_off);
              change_params();
              break;
    case 6  : puts("\nNew in_eps value : ");
              scanf("%d",&in_eps);
              change_params();
              break;
    case 7  : puts("\nNew decay value : ");
              scanf("%f",&decay);
              change_params();
              break;
    case 8  : break;
    default : puts("\n All parameters remain unchanged.");
  }
}

void learn()
/*  Allows the network to learn by updating the connections     */
/*  between nodes using a Hebbian Learning Method.  Thus        */
/*  nodes active at the same time develope strong connections.  */
 {
  int  i,j,k,l;                  /*  Counters  */
  int  patts,patts2;             /*  Number of SON patts to learn  */
  int  st;                       /*  Start node for pattern reading  */
  int  learn_act[layers][nds];   /*  Activation values of nodes during
                                     training  */
  float  lrate;                  /*  Learn rate for weight changes*/
  char  learnfile_name[15];      /*  Name of file cont. act values  */
  FILE  *learnfile;              /*  Ptr to act values for learning  */
```

```
puts("What is the name of the file containing the data to learn
                                        ?"); scanf("%s",learnfile_name);
puts("What learning rate is to be used ?");
scanf("%f",&lrate);
learnfile = fopen(learnfile_name,"r");
if (learnfile == (FILE*) 0)
{
    puts("\n Cannot open file containing activation values.");
  exit(1);
}
fscanf(learnfile,"%d %d %d %d", &nodes[0], &nodes[1], &patts,
                                                    &patts2);
for (i = 0;i < patts;i++)
{
  for (j = 0;j < layers;j++)
    for (k = 0;k < nodes[j];k++)
      fscanf(learnfile,"%d",&learn_act[j][k]);
  for (j = 0;j < nodes[1];j++)
    for (k = j + 1;k < nodes[1];k++)
      weight[1][j][k] += lrate * learn_act[1][j] * learn_act[1][k];
  for (l = 0;l < nodes[1];l++)
    for (j = 0;j < nodes[0];j++)
      ws_wght[l][j] += lrate * learn_act[1][l] * learn_act[0][j]; }
for (l = 0;l < patts2;l++)
{
  for (i = 0;i < layers;i++)
    for (j = 0;j < nodes[i];j++)
      fscanf(learnfile,"%d",&learn_act[i][j]);
  for (i = 0;i < nodes[0];i++)
    for (j = 0;j < nodes[1];j++)
      sw_wght[i][j] += lrate * learn_act[0][i] * learn_act[1][j]; }
while (fscanf(learnfile,"%d",&learn_act[0][0]) == 1)
{
  for (i = 1;i < nodes[0];i++)
    fscanf(learnfile,"%d",&learn_act[0][i]);
```

```
    for (i = 0;i < nodes[0];i++)
      for (j = i + 1;j < nodes[0];j++)
        weight[0][i][j] = lrate * learn_act[0][i] * learn_act[0][j]; }
  fclose(learnfile);
}

void read_net()
/*  Defines the network by allocating weight matrix and        */
/*  entering values. Also enters values for threshold and decay */
/*  Sets the activation and input values for all nodes to zero.  */
 {
  int i,j,k;                    /* Counters */
  char infile_name[15];         /* Name of file containing weights */
  FILE *infile;                 /* Ptr to file containing weights */

  puts ("What is the name of the file containing the network
    specifications ?");
  scanf("%s",infile_name);
  infile = fopen(infile_name,"r");
  if (infile == (FILE*) 0)
  {
    puts("\n Cannot open weights file.");
    exit(1);
  }
  fscanf(infile,"%d %d",&nodes[0],&nodes[1]);
  for (i = 0;i < layers;i++)
    for (j = 0;j < nodes[i];j++)
      for (k = j + 1;k < nodes[i];k++)
        fscanf(infile, "%f",&weight[i][j][k]);
  for (i = 0;i < nodes[0];i++)
    for(j = 0;j < nodes[1];j++)
      fscanf(infile,"%f",&sw_wght[i][j]);
```

```
  for (i = 0;i < nodes[1];i++)
    for (j = 0;j < nodes[0];j++)
      fscanf(infile,"%f",&ws_wght[i][j]);
}


void rec_seq()
/*  Oversees the reciting of the sequence.  Clearing any        */
/*   remaining activation. Setting the variable parameters, and */
/*  Cycling values around the network, to produce output of     */
/*  sequence elements. This subroutine is based on the ideas of */
/* competitive queuing from Houghton (1990).                    */
 {
  int con1 = 1;                 /* continue queue its 1 = layer 0 */
  int con2 = 1;                 /* 2 = layer 1  */
  int begin;                    /* begin recitation from here */
  int st_que[2],ed_que[2];      /* start, end elements of the queue*/
  int i,j;                      /*  counters  */

  for (i = 0;i < layers;i++)
    resp[i] = nodes[i]+1;
  clear_net();
  open_act();
  puts("\nWhere would you like to begin reciting (0-9)");
  scanf("%d",&begin);
  for (i = 0;i < eps;i++)
  {
    if (i < in_eps)
    {
      if (begin == 0)
        node_act[0][0] = input;
      else
      {
        node_act[1][begin] = input;
        cycle_act_ws();
      }
```

```
    }
    cycle_activation(0,i);
  }
  set_queue(0,&st_que[0],&ed_que[0]); node_act[1][begin] = 0;
  while (con1 == 1)
  {
    puts("\nLayer 0"); comp_queue(0,st_que[0],ed_que[0],&con1);
    for (j = 0;j < nodes[0];j++)
      if (j == resp[0])
        node_act[0][j] = 1;
      else
        node_act[0][j] = 0;
    con2 = 1;
    for (i = 0;i < eps;i++) cycle_activation(1,i);
                set_queue(1,&st_que[1],&ed_que[1]);
    while (con2 == 1)
    { comp_queue(1,st_que[1],ed_que[1],&con2);
      node_act[1][resp[1]] = 0.0;
    }
  }
fclose(actfile);
}

void  cycle_act_ws()
/* Cycles activation from the word layer to the sequence      * /
/*  layer to begin recitation of the sequence.                * /
{
  int  i,j;

  for (i = 0;i < nodes[1];i++)
    for (j = 0;j < nodes[0];j++)
      node_act[0][j] += ws_wght[i][j] * node_act[1][i];
}
```

```
void  set_queue(lar,st_q,ed_q)
int  lar;
int  *st_q,  *ed_q;
/*  Coordinates the running of the CQ for each layer                 * /
/*  determining the number of elements in the queue by way           * /
/*  of the start and end elements.    * /
{
   int  i;                          /*  counter  */
   int  temps,tempe;                /* start and end of queue temps */

   temps = nds+1;
   tempe = 0;
   for (i = 0;i < nodes[lar];i++)
      if  (node_act[lar][i]  >  cut_off)
      {
         if (i < temps)
            temps = i;
         if (i > tempe)
            tempe = i;
         que[lar][i] = 1 - node_act[lar][i];
      }
       printf("\nThere are %d in the queue for layer %d.",tempe -
                                    temps+1,lar);
   *st_q = temps;
   *ed_q = tempe;
}


void comp_queue(lyr,que_s,que_e,go)
int  lyr,que_s,que_e;
int  *go;
/*  Completes one cycle of the competitive queue from                * /
/*  Houghton (1990). That that is cycling activation and             * /
/*  producing a winner for the competition.                          * /
{
```

```
static int timethru = 1;     /* for originating the subseq */
int  i,j;                    /* loop counters */
int done;                    /* denotes finished competition */
int  tempg;                  /* placeholder for go pointer */
int  max;                    /* maximium filter value position */
int supp_e;                  /* number of eps for competition */
float  fil[nds];             /* activation of nodes in the filter */
float  neg[nds];             /* negative input to node i in comp */
float  temp[nds];            /* temporary value holder */

if (timethru == 1)
  timethru++;
else
  que[lyr][resp[lyr]] -= rval;
for(i = 0;i < nds;i++)
  fil[i] = 0;
for (i = que_s;i <= que_e;i++)
  fil[i] = que[lyr][i];
tempg = 0;
done = 0;
for (i = que_s;i <= que_e;i++)
  for (j = que_s;j <= que_e;j++)
 if ((fil[i] > 0 && fil[j] > 0 && fil[i] - fil[j] > 0.001) || i == que_e)
  tempg = 1;
supp_e = 1;
while (done == 0 && tempg == 1)
{
  for (i = que_s;i <= que_e;i++)                neg[i] = 0;
                  for (i = que_s;i <= que_e;i++)
  {
    for (j = que_s;j <= que_e;j++) if ((j != i) && (fil[j] > 0))
      neg[i] += 0.09 * fil[j];                          /*##*/
    temp[i] = (0.5 * fil[i]) - neg[i];                  /*##*/
  }
  max = 0;
```

```
    for (j = que_s;j <= que_e;j++)
    {
      fil[j] += temp[j];                                    /*##*/
      if (fil[j] > 1)                                       /*##*/
      {
        if (fil[j] > fil[max])
          max = j;
        done = 1;
        resp[lyr] = max;
      }
    }
    supp_e++;
  }
  if (resp[lyr] == que_e)
    tempg = 0;
  if (done == 1)
  {
    printf("\nThe next element is %d in %d
                                    epochs.",resp[lyr],supp_e);
    fprintf(actfile,"\nThe next element is %d in %d
                                    epochs.",resp[lyr],supp_e);
  }
 *go = tempg;
}


void batch_net()
/*  Oversees the running of the network.  Clearing any          * /
/*   remaining activation, setting the variable parameters, and  * /
/*  Cycling values around the network, for each comparison in    */
/*  a variable number of comparisons of the same two numbers  * /
{
```

```
int  access_1, access_2;  /* The values to access the array  */
                          /* for the numbers. 1 = first, 2 =  * /
                          /*  second. */
int num_comps;            /* Number of digit comparisons  */
int  i,j,k,l;             /*  Counter */
int ep_sum = 0;           /* Sum of the number of epochs  */
int  its;                 /* Iterations for each layer */
int  include;             /* Layers contributing to the walk */
float  step_size;         /* The size of the step for the RW */
float  temp1;
float  temp2;


set_up_net(&access_1,&access_2);
puts("\nHow many times would you like to compare these
numbers ?");
scanf("%d",&num_comps);
step_accum = startWalk;
errors = 0;
err_eps = 0;
for (i = 0;i < num_comps;i++)
{
  clear_net();
  node_act[1][access_1] = input;
  node_act[1][access_2] = input;
  step_c[2] = access_1;
  step_c[3] = access_2;
  cycle_act_ws();
  l = 0;
  for(k = 0;k < nodes[0];k++)
    if (node_act[0][k] > cut_off && l < 2)
    {
      step_c[l] = k;
      fprintf(actlog,"\nsteps are provided by %d in layer
                                                  0\n",step_c[l]);
      l++;
```

```
    }
  if (l == 2)
    include = 0;
  else
    include = 1;
  fprintf(actlog,"\nl = %d and include = %d.\n",l,include);
  node_act[1][access_1] = 0;
  node_act[1][access_2] = 0;
  its = 1;
  step_accum = startWalk;
  while (step_accum >= (-1.0 * threshold) && step_accum <=
                                                    threshold)
  {
    for (j = 0;j < layers;j++)
    {
      cycle_activation(j,epoch);
      if (j >= include)
      {
        step_size = calc_step(j,step_c);
        print_act_step(step_size);
        step_accum += step_size;
      }
    }
    its++;
  }
  print_result(access_1,access_2);
  ep_sum += epoch;
}
printf("\n\nThe sum of the number of epochs to decide is
                                                    %d.",ep_sum);
printf("\n  And the number of errors made is %d, epochs =
                                                    %d",errors,err_eps);
temp1 = ep_sum - err_eps;
temp2 = num_comps - errors;
```

```
    printf("\nThe average number of epochs for non-errors is
                                                %4.2f.",temp1/temp2);
    fprintf(actlog,"\n\nThe sum of the number of epochs to decide is
                                                    %d.",ep_sum);
    fprintf(actlog,"\n  And the number of errors made is %d, epochs =
                                                  %d",errors,err_eps);
    fclose(actlog);
}


void  set_up_net(ptacc_1,ptacc_2)
int  *ptacc_1,*ptacc_2;
/*  Sets the variable parameters of the network.  The rate of    * /
/*  activation build in the nodes, the file to output to, and the  * /
/*  numbers to be compared.  On the first call of this procedure * /
/* the random number seed is changed allowing variability in    * /
/*  response times.                                 */
{
    static int timethru = 1;     /* Identifies the first call of proc. */
    int seed;                    /* Seed for the random num gener. */
    char act_log[15];            /* File Name to output act values */
    int tempi1,tempi2;           /* Two temporary integer vaiables. */

    if (timethru == 1)
    {
        puts("Enter a value to seed the random number generator.");
        scanf("%d",&seed);
        srand(seed);
        timethru++;
    }
    puts("\nEnter the name of the file to output activation and step
                                                        values to.");
    scanf("%s",act_log);
    actlog = fopen(act_log,"w");
```

```
  if (actlog == (FILE*) 0)
  {
    puts("\n Cannot open file to output activation and step
    values."); exit(1);
  }
  puts("\nEnter the two numbers to be compared. ");
  scanf("%d  %d",&tempi1,&tempi2);
  *ptacc_1 =  tempi1;
  *ptacc_2 = tempi2;
}

float  calc_step(ler,c_step)
int  ler;
int  c_step[];
/*  Calculates the reaction time, in iterations, to decide if     */
/*  the first of two numbers is greater than the second.  By      */
/*  using the difference in activation between the node of        */
/*  interest and the least active node in the network as the      */
/*  mean of a gaussian distribution for the step size for a       */
/*  random walk, with standard deviation of 0.1.  The positive    */
/*  threshold of 1.0 which represents the decision that the       */
/*  first number as the larger. There is no facility for          */
/*  determining that the second of the two numbers is the         */
/*  smaller.  The ideas behind the implementation of the          */
/*  random walk are taken from Cohen, Dunbar and McClelland       */
/*   (1990)                                                       */
{
  float mean;          /*  Mean of the Gaussian Distribution. */
  int x;               /*  Random percentage to access dist. */
  float z_score;       /*  Z score of random percent calc'd   */
  float temp;          /*  Holds the value to be returned     */
  int sub1,sub2;       /*  Subscripts to access Z -table      */
```

```
  if (ler == 0)
    mean = node_act[ler][c_step[1]] - node_act[ler][c_step[0]];
                                                              /**/
  else
    mean = node_act[ler][c_step[3]] - node_act[ler][c_step[2]];
  x = rand();
  sub2 = x % 10;
  sub1 = (x % 100) - sub2;
  sub1 /= 10;
  if (sub1 >= 5)
    z_score = z_table[sub1 - 5][sub2];
  else
    z_score = (-1) * z_table[sub1][sub2]; temp = (z_score *
  standDev) + mean; return temp;
}

void  print_act_step(s_size)
float  s_size;
/*  This procedure prints the activation values and step-size   * /
/*  at each step in the random walk to the screen and to the    * /
/*   activation file.                                           * /
{
  int i,j;                              /*  Counter.  */

  fprintf(actlog,"\n\n        Epoch Number %d \n",epoch);
  for (i = 0;i < layers;i++)
  {
    fprintf(actlog,"\nLayer  %d\n",i);
    for (j = 0;j < nodes[i];j++)
    {
fprintf(actlog,"\nThe activation of node %d is %f",j,node_act[i][j]);
    }
  }
  fprintf(actlog,"\nThe step size is %2.4f",s_size);
}
```

```
void  print_result(acc_1,acc_2)
int  acc_1,acc_2;
/*  Prints the decision the network has reached and the number * /
/*  of epochs taken to reach that decision.    */
{
  fprintf(actlog,"\n\nThe threshold has been reached in %d
                                                    epochs.",epoch);
  fprintf(actlog,"\nThe  comparator  %d  is",acc_2+1);
  if (step_accum >= 1.0)
    fprintf(actlog," MORE ");
  else
  {
    fprintf(actlog," NOT MORE ");
    errors++;
    err_eps += epoch;
  }
  fprintf(actlog,"than  the  comparator  %d.",acc_1+1);
}

void  clear_net()
/*  Clears the activation from the network to allow the          * /
/*  generation of a response to a new combination of numbers.  * /
 {
  int i,j;                                  /*  Counter.  */

  for (i = 0;i < layers;i++)
    for (j = 0;j < nodes[i];j++)
    {
      node_act[i][j] = 0;
      net_inp[i][j] = 0;
    }
  for (j = 0;j < layers;j++)
    for (i = 0;i < nds;i++)
      que[j][i] = 0;
  epoch = 1;
}
```

```
void  clear_weights()
/* Clears the weight values of the network to allow new values */
/*  to be read in or learned.                                  * /
{
  int  i,j,k;

  for (i = 0;i < layers;i++)
    for (j = 0;j < nodes[i];j++)
      for (k = j + 1;k < nodes[i];k++)
        weight[i][j][k] = 0.0;
  for (i = 0;i < nodes[0];i++)
    for (j = 0;j < nodes[1];j++)
      sw_wght[i][j] = 0.0;
}

void  cycle_activation(lay,ech)
int  lay,ech;
/*  Calculates the activation value for each unit and cycles this */
/*  activation around the network.  Outputting the activation    * /
/*  value for each node to the screen and the activation file.   * /
/*  The building of activation by determining the average input  */
/*  to the node over time is taken from Cohen, Dunbar and        * /
/*  McClelland (1990).                                           * /
 {
  int j,k,l;                          /*  Counters  */
  float net_inp_new[nds];              /* Input to a Node at time t */
  float net_inp_old[nds];             /* Net input to node at (t - 1) */
  static int time[2] = {1,1};         /* First time thru for layer  */

  if (time[lay] == 1)
  {
    for (j = 0;j < nodes[lay];j++)
    {
      net_inp_old[j] = 0; net_inp_new[j] = 0;
    }
    time[lay]++;
```

```
  }
  for (j = 0;j < nodes[lay];j++)
  {
    net_inp_old[j] = net_inp[lay][j]; net_inp_new[j] = 0;
  }
  for (j = nodes[lay] - 1;j >= 0;j--)
  {
    for (k = 0;k < j;k++)
      net_inp_new[j] += node_act[lay][k] * weight[lay][k][j];
    if (lay == 1 && ech < in_eps)
      for (l = 0;l < nodes[0];l++)
        net_inp_new[j] += node_act[0][l] * sw_wght[l][j];
    if (net_inp_old[j] != 1 && net_inp_new[j] < 0.0001)
      net_inp[lay][j] = 0;
    else
      net_inp[lay][j] = (rate * net_inp_new[j]) + ((1 - rate) *
                                          (net_inp_old[j] / epoch)); /**/
    net_inp[lay][j] = net_inp[lay][j] / temperature;
                                                  /* Div by Temp */
    if (net_inp[lay][j] != 0)
      node_act[lay][j] = (float)(1 / (1 + exp((-1) *
                                        (double)net_inp[lay][j])));
    else
      node_act[lay][j] -= decay * node_act[lay][j];
  }
  epoch++;
}

void open_act()
/* Opens activation file      */
{
  char actfile_name[15];          /* File Name to output weights to */

  puts("\nWhat would you like to name the activation file ?");
  scanf("%s",actfile_name);
  actfile = fopen(actfile_name,"w");
```

```
  if (actfile == (FILE*) 0)
  {
    puts("\n Cannot open file to output weights.");
    exit(1);
  }
}

void pt_act()
/* prints activations */
{
  int i,j;                    /* Counters */

  fprintf(actfile,"\nEpoch %d",epoch);
  for (i = 0;i < layers;i++)
  {
    fprintf(actfile,"\nlayer %d",i);
    for (j = 0;j < nodes[i];j++)
      fprintf(actfile,"\nact [%d][%d] = %f",i,j,node_act[i][j]);
  }
}

void write_wts_file()
/* Writes the weights of the network to a file.                    */
{
  int i,j,k;                  /* Counters */
  char outfile_name[15];      /* File name to output weights to */
  FILE *outfile;              /* Pointer to the output file */

  puts("\nWhat would you like to name the weight file ?");
  scanf("%s",outfile_name);
  outfile = fopen(outfile_name,"w");
  if (outfile == (FILE*) 0)
  {
    puts("\n Cannot open file to output weights.");
    exit(1);
  }
```

```
  for (i = 0;i < layers;i++)
  {
    fprintf(outfile,"\n  Layer  %d\n",i);
    for (j = 0;j < nodes[i];j++)
    {
      for (k = j + 1;k < nodes[i];k++)
        fprintf(outfile,"weight [%d][%d][%d] = %1.3f", i, j, k,
                                                    weight[i][j][k]);
      fprintf(outfile,"\n");
    }
  }
  for (i = 0;i < nodes[0];i++)
  {
    for (j = 0;j < nodes[1];j++)
  fprintf(outfile,"sw_weight [%d][%d] = %1.3f     ",i,j,sw_wght[i][j]);
    fprintf(outfile,"\n");
  }
  fprintf(outfile,"\n");
  for (i = 0;i < nodes[1];i++)
  {
    for (j = 0;j < nodes[0];j++)
  fprintf(outfile,"ws_weight [%d][%d] = %1.3f",i,j,ws_wght[i][j]);
    fprintf(outfile,"\n");
  }
  fclose(outfile);
}
```

REFERENCES

Albert, M.; Yamadori, A.; Gardner, H. and Howes, D. (1973). Comprehension and alexia. *Brain,* **96**, 317-328.

Allen, R. (1988). Sequential connectionist networks for answering simple questions about a microworld. *Proceedings of the Cognitive Science Society.* pp.489-495.

Allen (1990) Connectionist language users. *Connection Science*, **2(4)**, 279-311.

Ash, T. (1989). Dynamic node creation in backpropagation networks. *Technical Report ICS-8901*, Institute of Cognitive Science, University of California, San Diego.

Ashby, F. (1982). Deriving exact predictions from the cascade model. *Psychological Review*, **89(5)**, 599-607.

Ashby, F. and Perrin, N. (1988). Toward a unified theory of similarity and recognition. *Psychological Review*, **95(1)**, 124-150.

Ashcraft, M. (1992). Cognitive arithmetic: A review of data and Theory. *Cognition*, **44**, 75-106.

Ashcraft, M. (1987). Children's knowledge of simple arithmetic: A developmental model and simulation. In J. Bisanz, C. Brainerd and R. Kail (Eds.). *Formal Methods in Developmental Psychology.* Springer-Verlag.

Ashcraft, M. and Battaglia, T. (1978). Cognitive arithmetic: Evidence for retrieval and decision processes in mental addition. *Journal of Experimental Psychology: Human Learning and Memory*, **4(5)**, 527-538.

Ashcraft, M. and Christy, K. (1991). Tabulation of problem frequency in elementary texts: Grades 3-6. Unpublished Manuscript.

Ashcraft, M. and Stazyk, E. (1981). Mental addition: A test of three verification models. *Memory and Cognition*, **9**, 185-196.

Ashcraft, M.; Yamashita, T. and Aram, D. (1992). Mathematics performance in left and right brain-lesioned children. *Brain and Cognition*, **19**, 208-252.

Assal, G. and Jacot-Descombes, C. (1984). Intuition arithmétique chez un acalculique. *Review Neurologique*, **140(5)**, 374-375.

Assal, G.; Buttet, J. and Jolivet, R. (1981). Dissociations in aphasia: A case report. *Brain and Language*, **13**, 223-240.

Audley, R. and Wallis, C. (1964). Response instructions and the speed of relative judgements. *British Journal of Psychology*, **55**, 59-93.

Baddeley, A. (1976). *The Psychology of Memory*. New York: Basic.

Baddeley, A. and Lewis, V. (1981). Inner active processes in reading: The inner voice, the inner ear, and the inner eye. In A. Lesgold and C. Perfetti (Eds.). *Interactive Processes in Reading*. Hillsdale, NJ: Lawrence Erlbaum. pp. 107-129.

Balde, P. and Venkatesh, S. (1988). On properties of networks with neurone-like elements. *Proceedings of IEEE Conference on NIPS*, Denver, American Institute of Physics.

Banks, W. and Flora, J. (1977). Semantic and perceptual processes in symbolic comparisons. *Journal of Experimental Psychology: Human Perception and Performance*, **3**, 278-290.

Banks, W.; Fuji, M. and Kayra-Stuart, F. (1976). Semantic congruity effects in comparative judgements of magnitudes of digits. *Journal of Experimental Psychology: Human Perception and Performance*, **2**, 435-447.

Barbizet, J.; Bindefeld, N.; Moaty, F. and Le Goff, P. (1967). Persistance de possibilité de calcul élémentaire au cours des aphasies massives. *Review Neurologique*, **116**, 170-178.

Barclay, J. (1973). The role of comprehension in remembering sentences. *Cognitive Psychology*, **4**, 229-254.

Bartell, B. and Cottrell, G. (1991). A model of symbol grounding in a temporal environment. *Proceedings of the International Joint Conference on Neural Nets*, Settle.

Barto, A. (1985). Learning by statistical co-operation. *Human Neurobiology*, **4**, 229-256.

Bates, E. and Elman, J. (1992). Connectionism and the study of change. to appear in M. Johnson (Ed.). *Brain Development and Cognition: A reader.* Oxford: Blackwell.

Battig, W.; Brown, S. and Schild, M. (1964). Serial position and sequential associations in serial learning. *Journal of Experimental Psychology*, **67**, 449-457.

Becker, J. (1993). Young children's use of number words: Counting in many-to-one situations. *Developmental Psychology*, **29(3)**, 458-465.

Belmont, J. and Butterfield, E. (1969). The relations of short term memory to the development of intelligence. In L. Lipsitt and H. Reese (Eds.). *Advances in Child Development and Behaviour. Vol. 4.* New York: Academic Press.

Benson, D. and Denckla, M. (1969). Verbal paraphasia as a source of calculation disturbance. *Archives of Neurology*, **21**, pp. 96-102.

Benson, A. and Weir, W. (1972). Acalculia: Acquired anarithmetia. *Cortex*, **8**, 465-472.

Benton, A. (1981). Focal brain damage and the concept of localisation of function. In C. Loeb (Ed.). *Studies in Cerebrovascular Disease*. Milano: Masson Italia Editori. pp. 47-56.

Bertin, J. (1980). The basic test of the graph: A matrix theory of graph construction and cartography. In P.A. Kolers; M.E. Wrolstad and H. Bouma (Eds.). *Processing of Visible Language 2*. New York: Plenum.

Besner, D. (1987). Phonology, lexical access in reading, and articulatory suppression: A critical review. *Quarterly Journal of Experimental Psychology*, **39A**, 467-478.

Besner, D. and Coltheart, M. (1979). Ideographic and alphabetic processing in skilled reading of English. *Neuropsychologia*, **17**, 467-472.

Besner, D. and Davelaar, E. (1983). Basic processes in reading: Two phonological codes. *Canadian Journal of Psychology*, **36**, 701-711.

Bewley, W. (1972). The functional stimulus in serial learning. In R.F. Thompson and J.F. Voss (Eds.). *Topics in Learning and Performance*. New York: Academic Press.

Bewley, W. (1969). The effect of locus and degree of associative disruption on the acquisition of a serial list. *Unpublished Doctoral Dissertation*, University of Wisconsin.

Bewley, W. (1966). The effect of varying probability of association and position of items on acquisition and transfer of serial learning. *Unpublished M.A. Thesis*, University of Wisconsin.

Bower, G. (1971). Adaptation-level coding of stimuli and serial position effects. In M.H. Appley (Ed.). *Adaptation-Level Theory*. New York: Academic Press.

Bower, G.; Lesgold, A. and Tieman, D. (1969). Grouping operations in free recall. *Journal of Verbal Learning and Verbal Behaviour*, **8**, 481-493.

Bower, G. and Winzenz, D. (1969). Group structure, coding and memory for digit series. *Journal of Experimental Psychology*, **80** (monograph supplement), 1-17.

Breckinridge, K. and Dixon, T. (1970). Problem of the stimulus in serial learning. *Journal of Experimental Psychology*, **83**, 126-130.

Broadbent, D. (1971). *Decision and Stress*. London: Academic Press.

Broadbent, D. (1958). *Perception and Communication*. New York: Pergamon.

Bryant, P. and Tarabasso, T. (1971). Transitive inferences and memory in young children. *Nature*, **232**, 456-458.

Bub, D.; Cancelliere, A. and Kertesz, A. (1985). Whole-word and analytic translation of spelling to sound in a non-semantic reader. In K.E. Patterson; J.C. Marshall and M. Coltheart (Eds.). *Surface Dyslexia.* London: Erlbaum. pp. 15-34.

Buckley, P. and Gillman, C. (1974). Comparisons of digits and dot patterns. *Journal of Experimental Psychology*, **103**, 1131-1136.

Bugelski, B. (1942). Interference with recall of original responses after learning new responses to old stimuli. *Journal of Experimental Psychology*, **30**, 368-379.

Burgess, N. and Hitch, G. (1992). Toward a network model of the articulatory loop. *Journal of Memory and Language*, **31**, 429-460.

Campbell, J. and Clark, J. (1988). An encoding complex view of cognitive number processing: Comment on McCloskey, Sokol, and Goodman (1986). *Journal of Experimental Psychology: General*, **117**, 204-214.

Campbell, J. and Graham, D. (1985). Mental multiplication skill: Structure, process and acquisition. *Canadian Journal of Psychology*, **39**, 338-366.

Caramazza, A. and McCloskey, M. (1987). Dissociations of calculation processes. In G. Deloche and X. Seron (Eds.). *Mathematical Disabilities: A Cognitive Neuropsychological Perspective.* Hillsdale, NJ: Lawrence Erlbaum.

Carley, L. (1988). Pre-synaptic neural information processing. *Proceedings of IEEE Conference on NIPS*, Denver, American Institute of Physics.

Clark, J. and Campbell, J. (1991). Integrated versus modular theories of number skills and acalculia. *Brain and Cognition*, **17(2)**, 204-239.

Cleermans, A. and McClelland, J. (1991). Learning the structure of event sequences. *Journal of Experimental Psychology: General*, **120(30)**, 235-253.

Cleermans, A.; Servan-Schreiber, D. and McClelland, J. (1991). Graded state machines: The representation of temporal contingencies in simple recurrent nets. *Machine Learning*, **7**, 2-3.

Cohen, A.; Ivry, R. and Keele, S. (1990). Attention and structure in sequence learning. *Journal of Experimental Psychology: Learning, Memory and Cognition*, **16(1)**, 17-30.

Cohen, L. and Dehaene, S. (1991). Neglect dyslexia for numbers? A case report. *Cognitive Neuropsychology*, **8**, 39-58.

Cohen, J.; Dunbar, K. and McClelland, J. (1990). On the control of automatic processes: A parallel distributed processing account of the Stroop effect. *Psychological Review*, **97**, 332-361.

Cohen, J.; Servan-Schreiber, D. and McClelland, J. (1992). A parallel distributed processing approach to automaticity. *American Journal of Psychology*, **105(2)**, 239-269.

Collard, R. and Povel, D-J. (1982). Theory of serial pattern production tree traversals. *Psychological Review*, **89(6)**, 693-707.

Collignon, R.; Leclerq, C. and Mahy, J. (1977). Etude de la sémiologie des troubles du calcul observés au cours de lésions corticales. *Acta Neurologica Belgica*, **77**, 257-275.

Coltheart, M. (1980). Deep Dyslexia: A right hemisphere hypothesis. In M. Coltheart; K. Patterson and J.C. Marshall (Eds.). *Deep Dyslexia*. London: Routledge and Kegan Paul. pp. 326-380.

Conrad, R. (1965). Order error in immediate recall of sequences. *Journal of Verbal Learning and Verbal Behaviour*, **4**, 161-169.

Cottrell, G. and Tsung, F-S. (1993). Learning simple arithmetic procedures. *Connection Science*, **5(1)**.

Curran, T. and Keele, S. (1993). Attentional and non-attentional forms of sequence learning. *Journal of Experimental Psychology: Learning, Memory and Cognition*, **19(1)**, 189-202.

Deese, J. (1958). *The Psychology of Learning*. London: McGraw-Hill.

Dehaene, S. (1992). Variations of numerical abilities. *Cognition*, **44**, 1-42.

Deloche, G. and Seron, X. (1987). Numerical transcoding: A general production model. In G. Deloche and X. Seron (Eds.). *Mathematical Disabilities: A Cognitive Neuropsychological Perspective*. Hillsdale, NJ: Lawrence Erlbaum. pp. 137-169.

Dennis, S. and Wiles, J. (1993). Integrating learning into models of human memory: The Hebbian recurrent network. *Technical Report*, Department of Computer Science, University of Queensland.

Derthick, M. and Tebelskis, J. (1988). Ensemble boltzman units have collective computational properties like those of Hopfield and Tank neurones. *Proceedings of the IEEE Conference on NIPS*, Denver, American Institute of Physics.

Duncan, J. and Humphreys, G. (1989). Visual search and stimulus similarity. *Psychological Review*, **96(3)**, 433-458.

Ebbinghaus, H. (1913). *Memory: A Contribution to Experimental Psychology*. New York: Columbia University, Teachers College.

Ebenholtz, S. (1963). Serial learning: Position learning and sequential associations. *Journal of Experimental Psychology*, **66**, 353-362.

Ellis, A. (1982). Spelling and writing (and reading and speaking). In A.W. Ellis (Ed.). *Normality and Pathology in Cognitive Functions.* London: Academic Press. pp. 113-146.

Elman, J. (1993). Learning and development in neural networks: The importance of starting small. *Cognition*, **48**, 71-99.

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

Ersu, E. and Tolle, H. (1988). Hierarchical learning control: An approach with neurone-like associative memories. *Proceedings of the IEEE Conference on NIPS*, Denver, American Institute of Physics.

Fairbank, B (1969). Experiments on the temporal aspects of number perception. *Dissertation Abstracts International*, **30**, 403.

Farah, M. and McClelland, J. (1991). A computational model of semantic memory impairment: Modality specific and emergent category specific. *Journal of Experimental Psychology: General*, **120(4)**, 339-357.

Feldman, J. and Ballard, D. (1982). Connectionist models and their properties. *Cognitive Science*, **6**, 205-254.

Ferro, J. and Botelho, M. (1980). Alexia for arithmetical signs: A cause of disturbed calculation. *Cortex*, **16**, 175-180.

Fetz, E. (1988). Correlational strength and the computational algebra of synaptic connections between neurones. *Proceedings of the IEEE Conference on NIPS*, Denver, American Institute of Physics.

Fitts, P. and Switzer, G. (1962). Cognitive aspects of information processing I: The familiarity of S-R sets and subsets. *Journal of Experimental Psychology*, **63**, 321-329.

Foltz, G. (1983). The representing and processing of number. (Doctoral Dissertation, University of Denver, 1982), *Dissertation Abstracts International*, **43**, 2371B.

Foltz, G.; Poltrock, S. and Potts, G. (1984). Mental comparison of size and magnitude: Size congruity effects. *Journal of Experimental Psychology: Learning Memory and Cognition*, **10**, 442-453.

Foos, P. (1975). Constructive memory: Inversion effects for pairs and sentences. Unpublished Doctoral Dissertation, Bowling Green State University.

Foos, G.; Smith, K.; Sabol, M. and Mynatt, B. (1976). Constructive processes in simple linear order problems. *Journal of Experimental Psychology: Human Learning and Memory*, **2**, 759-766.

French, R. (1991). Using semi-distributed representations to overcome catastrophic forgetting. *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum, pp. 173-178.

Fuson, K. (1988). *Children's Counting and Concept of Number*. New York: Springer-Verlag.

Fuson, K.; Richards, J. and Briars, D. (1982). The acquisition and elaboration of the number-word sequence. In C. Brainerd (Ed.). *Progress in Cognitive Developmental Research*. New York: Springer-Verlag.

Gardner, H. (1974). The naming of objects and symbols by children and aphasic patients. *Journal of Psycholinguistic Research*, **3(2)**, 133-149.

Gardner, H.; Strub, R. and Albert, M. (1975). A unimodal deficit in operational thinking. *Brain and Language*, **2**, 333-344.

Gathercole, S. and Baddeley, A. (1993). *Working Memory and Language*. Hove: Lawrence Erlbaum Associates.

Gaudiano, P. (1988). Temporal patterns of activity in neural nets. *Proceedings of the IEEE Conference on NIPS*, Denver, American Institute of Physics.

Geary, D. (1993). Mathematical disabilities: Cognitive, neuropsychological and genetic components. *Psychological Bulletin*, **114(2)**, 345-362.

Gelman, R. and Gallistel, C. (1978). *The Child's Understanding of Number*. Cambridge, MA: Harvard University Press.

Gonzalez, E. and Kolers, P. (1987). Notional constraints on mental operations. In G. Deloche and X. Seron (Eds.). *Mathematical Disabilities: A Cognitive Neuropsychological Perspective*. Hillsdale, NJ: Lawrence Erlbaum. pp. 27-41.

Gonzalez, E. and Kolers, P. (1982). Mental manipulation of arithmetic symbols. *Journal of Experimental Psychology: Learning, Memory and Cognition*, **8**, 308-319.

Goodman, R. and Caramazza, A. (1986). Aspects of the spelling process: Evidence from a case of acquired dysgraphia. *Language and Cognitive Processes*, **1**, 263-296.

Grafman, J.; Passafiume, D.; Faglioni, P. and Boller F. (1982). Calculation disturbances in adults with focal hemispheric damage. *Cortex*, **18**, 37-50.

Gréco, P. (1962). Quantité et quotité. In P. Gréco and A. Morf (Eds.). *Structures Numériques Élémentaries. In Etudes d'Épistémologie Génétique (Vol.13).* Paris, P.U.F.

Grewel, F. (1969). The Acalculias. In P. Vinken and G. Bruyn (Eds.). *Handbook of Clinical Neurology (vol. 3).* Amsterdam: North Holland.

Guttman, E. (1936). Congenital arithmetic disability and acalculia. *British Journal of Medical Psychology*, **16**, 16-35.

Hamann, M. and Ashcraft, M. (1986). Textbook presentations of the basic addition facts. *Cognition and Instruction*, **3**, 173-192.

Hamilton, J. (1979). Storage and retrieval of verbal sequences. Unpublished PhD Thesis, University of Glasgow, Scotland.

Hamilton, J. and Sanford, A. (1978). The symbolic distance effect for alphabetic order judgements: A subjective report and reaction time analysis. *Quarterly Journal of Experimental Psychology*, **30**, 33-43.

Harcum, E. (1975). *Serial Learning and Paralearning.* New York: Wiley.

Harris, C. (1991). Parallel Distributed Processing Models and Metaphors for Language and Development. PhD Dissertation, University of California, San Diego.

Henderson, L.; Coltheart, M. and Woodhouse, D. (1973). Failure to find a syllabic effect in number naming. *Memory and Cognition*, **1(3)**, 304-306.

Henschen, S. (1920). *Klinische und Anatomische Beitrag zür Pathologie des Gehirns.* Stockholm: Nordiske Bofhandelen.

Henschen, S. (1926). On the function of the right hemisphere of the brain in relation to the left in speech, music and calculation. *Brain*, **49**, 111-123.

Hinrichs, L.; Yurko, D. and Hu, M. (1981). Two-digit number comparison: Use of place information. *Journal of Experimental Psychology: Human Perception and Performance*, **7**, 890-901.

Hinton, G. and Plaut, D. (1987). Using fast weights to deblur old memories. *Proceedings of the Ninth Annual Conference of the Cognitive Science Society*, Hillsdale, NJ.: Erlbaum. pp. 177-186.

Holyoak, K. (1977). The form of analog size information in memory. *Cognitive Science*, **9**, 31-51.

Holyoak, K. and Walker, J. (1976). Subjective magnitude information in semantic orderings. *JULUB*, **15**, 287-299.

Horowitz, L. and Izawa, C. (1963). Comparison of serial and paired-associate learning. *Journal of Experimental Psychology*, **65**, 352-361.

Horton, D. and Turnage, T. (1970). Serial to paired-associate learning: Utilisation of serial information. *Journal of Experimental Psychology*, **84**, 88-95.

Houghton, G. (1990). The problem of serial order: A neural network model of sequence learning and recall. In R. Dale; C. Mellish and M. Zock (Eds.). *Current Research in Natural Language Generation*. London: Academic Press.

Hovancik, J. (1975). Reaction times for naming the first next and second next letters of the alphabet. *American Journal of Psychology*, **88(4)**, 643-647.

Hull, C. (1935). The conflicting psychologies of learning - a way out. *Psychological Review*, **42**, 491-516.

Hurford, J. (1987). *Language and Number*. Oxford: Basil Blackwell.

Huttenlocher, J. (1968). Constructing spatial images: A strategy in reasoning. *Psychological Review*, **75**, 550-560.

Itsukushima, Y.; Tozawa, J. and Itagaki, F. (1990). Representation and retrieval of two-digit numbers in mental comparison. *Japanese Psychological Research*, **32(3)**, 117-127.

Jacobs, R.; Jordan, M. and Barto, A. (1991). Task decomposition through competition in a modular connectionist architecture. *Cognitive Science*, **15**, 219-250.

Jaffe-Katz, A.; Budescu, D. and Wallstein, T. (1989). Timed magnitude comparisons of numerical and non-numerical expressions of uncertainty. *Memory and Cognition*, **17(3)**, 249-264.

Jamieson, D. and Petrusic, W. (1975). Relational judgements with remembered stimuli. *Perception and Psychophysics*, **18**, 373-378.

Jennings, P. and Keele, S. (1990). A computational model of attentional requirements in sequence learning. *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*. Hillsdale NJ : Erlbaum.

Jensen, A. (1962). Temporal and spatial effects of serial position. *American Journal of Psychology*, **75**, 390-400.

Jones, J.; Miller, B. and Scarborough, D. (1990). Discovering grouping structure in music. *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum.

Jordan, M. (1986). Serial order : A parallel distributed processing approach. *Technical Report No. ICS - 8604*, La Jolle CA : University of California at San Diego.

Kahana, M. and Greene, R. (1993). Effects of spacing on memory for homogenous lists. *Journal of Experimental Psychology: Learning, Memory and Cognition*, **19(1)**, 159-162.

Kahn, H. and Witaker, H. (1991). Acalculia: An historical review of localisation. *Brain and Cognition*, **17**, 102-115.

Keppel, G. (1964). Retroactive inhibition of serial lists as a function of presence or absence of positional cues. *Journal of Verbal Learning and Verbal Behaviour*, **3**, 511-517.

Klahr, D.; Chase, W. and Lovelace, E. (1983). Structure and process in alphabetic retrieval. *Journal of Experimental Psychology : Learning, Memory and Cognition*, **9(3)**, 462-477.

Klahr, D. and Chase, W. (1978). Developmental changes in latency patterns for access to the alphabet. *Annual Meeting of the Psychonomic Society*, San Antonio.

Klahr, D. and Wallace, J. (1976). *Cognitive Development: An information processing view.* Hillsdale, NJ: Lawrence Erlbaum.

Klapp, S. (1974). Syllable-dependent pronunciation latencies in number naming: A replication. *Journal of Experimental Psychology*, **102(6)**, 1138-1140.

Kolers, P. (1983). Perception and representation. *Annual Review of Psychology*, **34**, 129-166.

Kolers, P. (1968). Bilingualism and information processing. *Scientific American*, **218**, 18-86.

Kolers, P. (1964). Specificity of a cognitive operation. *Journal of Verbal Learning and Verbal Behaviour*, **3**, 244-248.

Kolers, P. and Katzman, M. (1966). Naming sequences. *Language and Speech*, **9**, 84-95.

Kolers, P. and Smythe, W. (1984). Symbol manipulation: alternatives to the computational view of mind. *Journal of Verbal Learning and Verbal Behaviour*, **23**, 289-314.

Kosslyn, S. and Pomerantz, J. (1977). Imagery, proposition, and the form of internal representation. *Cognitive Psychology*, **9**, 52-76.

Kruschke, J. and Movellen, J. (1991). Benefits of gain: Speeded learning and minimal hidden layers in back-propagation networks. *IEEE Transactions on Systems, Man and Cybernetics*, **21**, 273-280.

Lacouture, Y. and Marley, A. (1991) A connectionist model of choice and reaction time in absolute identification. *Connection Science*, **3(4)**.

Lakoff, G. (1993). Neurobiology: Convergence zones and conceptual structure. *Proceedings of the National Academy of Sciences Neurobiology Section*, Commentary on Damasio's Theory.

Laming, D. (1973). *Mathematical Psychology*. London: Academic Press.

Lashley, K. (1951). The problem of serial order in behaviour. In L.A. Jeffress (Ed.). *Cerebral Mechanisms in Behaviour*. New York: Wiley. pp. 112-136.

Lashley, K. (1929). *Brain Mechanisms and Intelligence*. Chicago: University of Chicago Press.

Laughery, K. and Spector, A. (1972). The roles of recoding and rhythm in memory organisation. *Journal of Experimental Psychology*, **94(1)**, 41-48.

Lawler, R. (1981). The progressive construction of mind. *Cognitive Science*, **5**, 1-30.

Leonhard, K. (1979). Ideokinetic aphasia and related disorders. In R. Hoops and Y. Lebrun (Eds.). *Neurolinguistics 9: Problems of Aphasia* . Lisse: Swets and Zeitlinger. pp. 11-77.

Leplat, J. and Chernais, M. (1977). A stochastic model for the analysis of discrimination tasks. *Acta Psychologica*, **41**, 261-272.

Lesper, H-O.; Melin, L.; Sjoden, P-O. and Fagerstrom, K-O. (1977). Temporal uncertainty of auditory signals in a monitoring task. *Acta Psychologica*, **41**, 183-190.

Lewandowsky, S. and Murdock, B. (1989). Memory for serial order. *Psychological Review*, **96(1)**, 25-57.

Link, S. (1990). Modelling imageless thought: Theory of numerical comparisons. *Journal of Mathematical Psychology*, **34**, 2-41.

Link, S. (1978). The relative judgement theory of analysis of response time deadline experiments. In N.J. Castellan and F. Restle (Eds.). *Cognitive Theory III*. Hillsdale, NJ: Erlbaum.

Link, S. (1975). The relative judgement theory of two choice response time. *Journal of Mathematical Psychology*, **12**, 114-135.

Lovelace, E. and Snodgrass, R. (1971). Decision times for alphabetical order of letter pairs. *Journal of Experimental Psychology*, **88**, 258-264.

Lovelace, E.; Powell, C. and Brooks, R. (1973). Alphabetic position effects in covert and overt alphabetic reaction times. *Journal of Experimental Psychology*, **99(3)**, 405-408.

Lovelace, E. and Spence, W. (1972). Reaction times for naming successive letters of the alphabet. *Journal of Experimental Psychology*, **94**, 231-233.

Luce, R. (1986). *Reaction Times: Their Role in Inferring Elementary Mental Organisation.* New York: Oxford University Press.

Maki, R. (1981). Categorisation and distance effects with spatial linear orders. *Journal of Experimental Psychology: Human Learning and Memory*, **7**, 15-32.

Mandler, G. (1967). Organisation and memory. In K. Spence and J.T. Spence (Eds.). *The Psychology of Learning and Motivation, Vol. 1.* New York: Academic Press.

Mareschal, D. and Shultz, T. (1993). A connectionist model of the development of seriation. *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society.* Hillsdale, NJ: Erlbaum.

Markman, A. and Gentner, D. (1990). Analogical mapping during similarity judgements. *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society.* Hillsdale, NJ. Erlbaum.

Marks, D. (1972). Relative judgement a phenomenon and a theory. *Perception and Psychophysics*, **11**, 156-160.

Marsh, L. and Maki, R. (1976). Efficiency of arithmetic operations in bilinguals as a function of language. *Memory and Cognition*, **4**, 459-464.

McAuley, J. and Stampfli, J. (1993). Analysis of the effects of noise on a model for the neural mechanism of short-term active memory. submitted to *Neural Computation.*

McClain, L. and Huang, J. (1982). Speed of simple arithmetic in bilinguals. *Memory and Cognition*, **10**, 591-596.

McClelland, J. (1992). Toward a theory of information processing in graded, random, interactive networks. In D. Meyer and S. Kornblum (Eds.). *Attention and Performance XIV: Synergies in experimental psychology, artificial intelligence and cognitive neuroscience - A Silver Jubilee Volume.* Cambridge, MA: MIT Press.

McClelland, J. (1979). On the time relations of mental processes. *Psychological Review*, **86**, 287-330.

McClelland, J. and Hinton, G. (1988). Learning representations by recirculation. *Proceedings of the IEEE Conference on NIPS*, Denver, American Institute of Physics.

McCloskey, M. (1992). Cognitive mechanisms in numerical processing: Evidence from acquired dyscalculia. *Cognition*, **44**, 107-157.

McCloskey, M.; Aliminosa, D. and Sokol, S. (1991). Facts, rules, and procedures in normal calculation: Evidence from multiple single-patient studies of impaired arithmetic fact retrieval. *Brain and Cognition*, **17**, 154-203.

McCloskey, M. and Caramazza, A. (1987). Cognitive mechanisms in normal and impaired number processing. In G. Deloche and X. Seron (Eds.). *Mathematical Disabilities: A Cognitive Neuropsychological Perspective.* Hillsdale, NJ: Lawrence Erlbaum. pp. 201-219.

McCloskey, M.; Caramazza, A. and Basili, A. (1985). Cognitive mechanisms in number processing and calculation: Evidence from dyscalculia. *Brain and Cognition*, **4**, 171-196.

McCloskey, M. and Cohen, N. (1989). Catastrophic interference in connectionist nets : The sequential learning problem. In G.H. Bower (Ed.). *The Psychology of Learning and Motivation, vol. 24*. New York: Academic Press.

McCloskey, M.; Macaruso, P. and Whetstone, T. (1992). The functional architecture of numerical processing mechanisms: Defending the modular model. In Campbell, J. (Ed.). *The Nature and Origins of Mathematical Skills: Advances in Psychology, 91*. North Holland: Amsterdam. pp. 493-537.

McCloskey, M.; Sokol, S. and Goodman, R. (1986). Cognitive processes in verbal-number production: Inferences from the performance of brain-damaged subjects. *Journal of Experimental Psychology: General*, **115**, 307-330.

McCloskey, M.; Sokol, S.; Goodman-Schulman, R. and Caramazza, A. (1990). Cognitive representations and processes in number production: Evidence from cases of acquired dyscalculia. In A. Caramazza (Ed.). *Advances in Cognitive Neuropsychology and Neurolinguistics*. Hillsdale, NJ: Erlbaum, pp. 1-32.

McLean, R. and Gregg, L. (1967). Effects of induced chunking on temporal aspects of serial recitation. *Journal of Experimental Psychology*, **74**, 455-459.

McRae, K. and Hetherington, P. (1993). Catastrophic forgetting is eliminated in pre-trained networks. *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society*, pp. 723-728.

Medin, D.; Goldstone, R. and Gentner, D. (1993). Respect for similarity. *Psychological Review*, **100(2)**, 254-278.

Mesulam, M. (1981). A cortical network for directed attention and unilateral neglect. *Annals of Neurology*, **10**, 309-325.

Miller, G. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, **63**, 81-97.

Miller, K.; Perlmutter, M. and Keating, D. (1984). Cognitive arithmetic: Comparison of operations. *Journal of Experimental Psychology: Learning, Memory and Cognition*, **10**, 46-60.

Morin, R.; DeRosa, D. and Stultz, V. (1967). Recognition memory and reaction time. *Acta Psychologica*, **27**, 298-305.

Morrison, C. (1990). Towards a model of alphabet learning on a connectionist machine. Unpublished Masters Thesis, University of Glasgow, Scotland.

Moyer, R. (1973). Comparing objects in memory. *Perception and Psychophysics*, **13**, 80-184.

Moyer, R. and Dumais, S. (1978). Mental comparison. In G.H. Bower (Ed.). *The Psychology of Learning and Motivation, vol. 12.* New York: Academic Press. pp. 117-155.

Moyer, R. and Landauer, T. (1973). Determinants of reaction time for digit inequality judgements. *Bulletin of the Psychonomic Society*, **1**, 167-168.

Moyer, R. and Landauer, T. (1967). Time required for judgements of numerical inequality. *Nature*, **215**, 1519-1520.

Mozer, M. (1993). Neural net architectures for temporal sequence processing. to appear In D. Weigend and N. Gershenfeld (Eds.). *Predicting the Future and Understanding the Past*. Redwood City, CA: Addison-Wesley.

Mueller, J. and Overcast, T. (1975). Grouped presentation in free recall. *American Journal of Psychology*, **88(4)**, 649-700.

Murdock, B. (1993). TODAM2: A model for storage and retrieval of item, associative and serial-order information. *Psychological Review*, **100(2)**, 183-203.

Murdock, B. (1983). A distributed memory model for serial-order information. *Psychological Review*, **90**, 316-338.

Murdock, B. (1982). A theory for the storage and retrieval of item and associative information. *Psychological Review*, **89**, 609-626.

Murdock, B. and vom Saal, W. (1967). Transpositions in short-term memory. *Journal of Experimental Psychology*, **74**, 137-143.

Myklebust, H. (1983). *Progress in Learning Disabilities*. London: Grune and Stratton.

Narendra, K. and Thathachar, M. (1974). Learning automata: A survey. *IEEE Transactions on Systems, Man and Cybernetics*, **4(4)**, 323-334.

Neisser, U. (1967). *Cognitive Psychology*. New York: Appelton-Century-Crofts.

Nissen, M.; Knopman, D. and Schacter, D. (1987). Neurochemical dissociation of memory systems. *Neurology*, **37**, 789-794.

Omlin, C. and Giles, C. (1993). Pruning recurrent neural nets for improved generalisation performance. *Revised Technical Report No. 93-6*, Computer Science Department, Rensselear Polytechnic Institute, Troy, N.Y.

Parkman, J. (1971). Temporal aspects of digit and letter inequality judgements. *Journal of Experimental Psychology*, **91**, 191-205.

Parkman, J. and Groen, G. (1971). Temporal aspects of simple addition and comparison. *Journal of Experimental Psychology*, **89**, 335-342.

Patterson, K. and Morton, J. (1985). From orthography to phonology: An attempt at an old interpretation. In K.E. Patterson; J.C. Marshall and M. Coltheart (Eds.). *Surface Dyslexia.* London: Erlbaum. pp. 335-359.

Peereman, R. and Holender, D. (1985). Visual field differences for number-nonnumber classification of alphabetic and ideographic stimuli. *Quarterly Journal of Experimental Psychology*, **37(2)**, 197-216.

Piaget, J. (1952). *The Child's Concept of Number.* New York: Norton.

Pike, R. and Olsen, D. (1977). A question of more or less. *Child Development*, **48**, 579-586.

Plate, T. (1992). Holographic recurrent networks. In C.L. Giles, S.J. Hanson and J.D. Cowan (Eds.). *Advances in Neural Information Processing Systems 5.* San Mateo, CA: Morgan Kaufmann.

Poltrock, S. (1989). A Random walk model of digit comparisons. *Journal of Mathematical Psychology*, **33(2)**, 131-162.

Polich, J. and Potts G. (1977). Retrieval strategies for linearly ordered information. *Journal of Experimental Psychology: Human Learning and Memory*, **3**, 10-17.

Potts, G. (1972). Information processing strategies used in the encoding of linear orderings. *Journal of Verbal Learning and Verbal Behaviour*, **11**, 727-740.

Potts, G.; Banks, W.; Kosslyn, S.; Moyer, R.; Riley, C. and Smith, K. (1978). Encoding and retrieval of comparative judgements. In N.J. Castellan and F. Restle (Eds.). *Cognitive Theory III*, Hillsdale NJ : Erlbaum.

Potts, G. and Scholz, K. (1975). The internal representation of a three-term series problem. *Journal of Verbal Learning and Verbal Behaviour*, **14**, 439-452.

Pavio, A. (1975). Perceptual comparisons through the mind's eye. *Memory and Cognition*, **3**, 635-647.

Raab, D. (1962). Statistical facilitation of simple reaction times. *Transactions of The New York Academy of Sciences*, **24**, 547-590.

Rabinowitz, F.; Grant, M. and Dingley, H. (1987). Computer simulation, cognition and development: An introduction. In J. Bisanz, C.J. Brainerd and R. Kail (Eds.). *Formal Methods in Developmental Psychology*. New York: Springer-Verlag.

Ratcliff, R. (1990). Connectionist models of recognition and memory. *Psychological Review*, **97**, 285-308.

Ratcliff, R.; Clarke, S. and Shiffrin, R. (1990). List strength effect I: Data and discussion. *Journal of Experimental Psychology: Learning, Memory and Cognition*, **16**, 163-178.

Restle, F. (1973). Serial pattern learning: Higher order transitions. *Journal of Experimental Psychology*, **99(1)**, 61-69.

Restle, F. (1970). Speed of adding and comparing numbers. *Journal of Experimental Psychology*, **83**, 274-278.

Richman, L. (1983). Language-learning disability: Issues, research, and future directions. In M. Wolraich and D. Routh (Eds.). *Advances in Developmental and Behavioural Pediatrics, Vol. 4*. Greenwich, CT: JAI Press. pp. 87-107.

Riley, M. and Smolensky, P. (1984). A parallel model of (sequential) problem solving. *Proceedings of the Sixth Annual Conference of the Cognitive Science Society*.

Robertson, S. and Weber, K. (1990). Parallel processes during question answering?. *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*. Cambridge, Mass: Lawrence Erlbaum. pp. 542-549.

Rourke, B. and Strang, J. (1978). Neuropsychological significance of the variations in patterns of academic performance: Motor, psychomotor, and tactile-perceptual abilities. *Journal of Pediatric Psychology*, **3**, 62-66.

Ross, G. (1977). Concept Categorisation in one-to two-year-olds. Unpublished Doctoral Dissertation. Harvard University.

Ruffieux, C.; Jacot-Descombes, C.; Python-Thuillard, F. and Assal, G. (1981). Why acalculic have not all gone bankrupt. *International Neuropsychological Society Meeting*. Aachen, Germany. June.

Rumelhart, D. and McClelland, J. (1986). On learning the past tenses of English verbs. In D. Rumelhart; J. McClelland and the PDP Research Group (Eds.). *Parallel Distributed Processing: Explorations in the microstructure of cognition, (Vol. 2)*. Cambridge, MA: MIT Press.

Rumelhart, D.; Hinton, G. and Williams, R. (1986). A general framework for parallel distributed processing. In D. Rumelhart; J. McClelland and the PDP Research Group (Eds.). *Parallel Distributed Processing: Explorations in the microstructure of cognition, (Vol. 1)*. Cambridge, MA: MIT Press.

Rumelhart, D. and McClelland, J. (1982). An interactive activation model of context effects on letter perception : Part 2. *Psychological Review*, **89**, 60-94.

Salthouse, T. (1991). Mediation of adult age differences in cognitions by reductions in working memory and speed of processing. *Psychological Science*, **2(3)**, 179-183.

Saufly, W. (1967). An analysis of cues in serial learning. *Journal of Experimental Psychology*, **74**, 414-419.

Schmidt, J. and Scheirer, C. (1977). Empirical approaches to information processing: Speed-accuracy tradeoff functions or reaction time. *Acta Psychologica*, **41**, 321-325.

Sekular, R. and Miekiewicz, D. (1977). Children's judgements of numerical inequality. *Child Development*, **48**, 630-633.

Sekular, R.; Rubin, E. and Armstrong, R. (1971). Processing numerical information: A choice time analysis. *Journal of Experimental Psychology*, **89**, 75-80.

Sergent, J. and Takane, Y. (1987). Structures in two-choice reaction time data. *Journal of Experimental Psychology: Human Perception and Performance*, **13(2)**, 300-315.

Seron, X.; Pesenti, M.; Noel, M-P; Deloche, G. and Cornet, J-A. (1992). Images of numbers or when 98 is upper left and 6 sky blue. *Cognition*, **44**, 159-196.

Servan-Schreiber, D.; Cleermans, A. and McClelland, J. (1988). Encoding sequential structure in simple recurrent networks. *Technical Report CMU-CS-88-183*, Carnegie-Mellon University.

Sharkey, N. and Sharkey, A. (1993). Interference and discrimination in neural net memory. *The Second Annual Neural Computation in Psychology Workshop*. Edinburgh.

Shepard, R.; Kilpatrick, D. and Cunningham, J. (1975). The internal representation of numbers. *Cognitive Psychology*, **7**, 82-138.

Shiffrin, R. and Cook, J. (1978). Short-term forgetting of item and order information. *Journal of Verbal Learning and Verbal Behaviour*, **17**,189-218.

Shiffrin, R.; Ratcliff, R. and Clarke, S. (1990). List strength effects II. *Journal of Experimental Psychology : Learning, Memory and Cognition*, **16**, 179-195.

Shiffrin, R. and Schneider, W. (1984). Automatic and controlled processing revisited. *Psychological Review*, **91(2)**, 269-276.

Siegler, R. (1987). The perils of averaging data over strategies: An example from children's addition. *Journal of Experimental Psychology: General*, **116**, 250-264.

Siegler, R. and Shrager, J. (1984). A model of strategy choice. In C. Sophian (Ed.). *Origins of Cognitive Skills*. Hillsdale, NJ: Erlbaum. pp. 229-293.

Siemja, F. and Mühlenbein, H. (1992). Reflective modular neural network systems. *German National Centre for Computer Science*.

Singer, H. and Low, A. (1933). Acalculia (Henschen): A clinical study. *Archives of Neurology and Psychiatry*, **29**, 476-498.

Smith, P. (1990). A note on the distribution of reaction times for a random walk with gaussian Increments. *Journal of Mathematical Psychology*, **34(4)**, 445-459.

Smith, G.; Poon, L.; Hale, S. and Myerson, J. (1988). A regular relationship between young and old adults' latencies. *Australian Journal of Psychology*, **20(2)**, 195-210.

Smith, K. and Foos, P. (1975). Effect of presentation order on the construction of linear orders. *Memory and Cognition*, **3**, 614-618.

Smith, K. and Mynatt, B. (1977). On the time required to construct a simple linear order. *Bulletin of the Psychonomic Society*, **9**, 435-438.

Smolensky, P. (1986). Neural and conceptual interpretation of PDP Models. In D. Rumelhart; J. McClelland and the PDP Research Group (Eds.). *Parallel Distributed Processing: Explorations in the microstructure of cognition, (Vol. 2)*. Cambridge, MA: MIT Press.

Sokol, S. and McCloskey, M. (1988). Levels of representation in verbal number production. *Applied Psycholinguistics*, **9**, 267-281.

Sokol, S.; McCloskey, M.; Cohen, N. and Aliminosa, D. (1991). Cognitive representations and processes in arithmetic: Inferences from the performance of brain-damaged patients. *Journal of Experimental Psychology: Learning, Memory and Cognition*, **17**, 355-376.

Spalding, J. and Zangwill, O. (1950). Disturbances of number form in a case of brain injury. *Journal of Neurology, Neurosurgery, and Psychiatry*, **13**, 24-29.

Spiers, P. (1987). Acalculia revisited: Current issues. In G. Deloche and X. Seron (Eds.). *Mathematical Disabilities: A Cognitive Neuropsychological Perspective*. Hillsdale, NJ: Lawrence Erlbaum.

St. John, M. and McClelland, J. (1992). Parallel constraint satisfaction as a comprehension mechanism. In R. Reilly and N. Sharkey (Eds.). *Connectionist Approaches to Natural Language Processing*. Lawrence Erlbaum.

Stark, K. (1968). Transfer from serial to paired-associate learning: A reappraisal. *Journal of Verbal Learning and Verbal Behaviour*, **7**, 20-30.

Stazyk, E.; Ashcraft, M. and Hamann, M. (1982). A network approach to simple multiplication. *Journal of Experimental Psychology: Learning Memory and Cognition*, **8**, 320-335.

Stornetta, W.; Hogg, T. and Huberman, B. (1987). A dynamical approach to temporal pattern processing. *Proceedings of IEEE Conference on NIPS*, Denver, American Institute of Physics.

Suci, G. (1967). The validity of pause as an index of units in language. *Journal of Verbal Learning and Verbal Behaviour*, **6**, 26-32.

Summers, J. (1977). Adjustments to redundancy in reaction time: A comparison of three learning methods. *Acta Psychologica*, **41**, 205-223.

Sun, G.; Lee, Y. and Chen, H. (1988). A novel net that learns sequential decision process. *Proceedings of the IEEE Conference on NIPS*, Denver, American Institute of Physics.

Takahashi, A. and Green, D. (1983). Numerical judgements with Kanji and Kana. *Neuropsychologia*, **21**, 259-263.

Tank, D. and Hopfield, J. (1987). Neural computation by concentrating information in time. *Proceedings of the IEEE Conference on Neural Nets*, San Diego, CA.

Tank, D. and Hopfield, J. (1986). Concentrating information in time: Analog neural networks with applications to speech recognition. *Proceeding of the National Academy Science, USA*, **84**, 1896-1900.

Tarabasso, T. (1975). Representation, memory and reasoning: How do we make transitive inferences?. In A. Pick (Ed.). *Minnesota Symposia of Child Psychology, (Vol. 9)*. Minneapolis: University of Minnesota Press.

Tarabasso, T. and Riley, C. (1975). On the construction and use of representations involving linear order. In R.L. Solso (Ed.). *Information Processing and Cognition: The Loyola Symposium*. Hillsdale, NJ.: Lawrence Erlbaum Associates.

Thurstone, L. (1927). Psychophysical analysis. *American Journal of Psychology*, **38**, 368-389.

Tzeng, O. and Wang, W. (1983). The first two R's. *American Scientist*, **71**, 238-243.

Vaid, J. (1985). Numerical size comparisons in a phonologically transparent script. *Perception and Psychophysics*, **37**, 592-595.

Vaid, J. and Corina, D. (1989). Visual field asymmetries in numerical size comparisons of digits, words, and signs. *Brain and Language*, **36**, 117-126.

Vickers, D. (1988). Discrimination reaction times. In A. Welford (Ed.). *Reaction Times*. London: Academic Press. pp. 25-73.

Wagner, S. and Walters, J. (1982). A longitudinal analysis of early number concepts. In G. Forman (Ed.). *Action and Thought*. New York: Academic Press. pp 137-161.

Warrington, E. (1982). The fractionation of arithmetical skills: A single case study. *Quarterly Journal of Experimental Psychology*, **34A**, 31-51.

Waugh, N. and Norman, D. (1965). Primary memory. *Psychological Review*, **72**, 89-104.

Welford, A. (Ed.). (1980). *Reaction Times*. London: Academic Press.

Wickelgren, W. (1977). Speed accuracy tradeoff in information processing dynamics. *Acta Psychologica*, **41**, 67-85.

Wilkes, A. (1972). Reading pauses during serial list learning with fixed or randomly changing groups. *Journal of Experimental Psychology*. **94(2)**, 206-209.

Wilkes, A. and Kennedy, R. (1970). The relative accessibility of list items within different pause defined groups. *Journal of Verbal Learning and Verbal Behaviour*, **9**,197-202.

Winkelman, J. and Schmidt, J. (1974). Associative confusions in mental arithmetic. *Journal of Experimental Psychology*, **102**, 734-736.

Winzenz, D. (1970). Group structure and coding in serial learning. Unpublished Doctoral Dissertation, Stanford University.

Wohlwill, J. and Lowe, R. (1962). Experimental analysis of the development of conservation of number. *Child Development*, **33**, 153-167.

Young, R. (1962). Tests of three hypotheses about the effective stimulus in serial learning. *Journal of Experimental Psychology*, **63**, 307-313.

Young, R. (1961). The stimulus in serial verbal learning. *American Journal of Psychology*, **74**, 517-528.

Young, R.; Patterson, J. and Benson, W. (1963). Backward serial learning. *Journal of Verbal Learning and Verbal Behaviour*, **1**, 335-338.

Yuille, A. and Grzywacs, N. (1989). A winner-take-all mechanism based on presynaptic inhibition feedback. *Neural Computation*, **1**, 334-347.

Zbrodoff, N. and Logan, G. (1986). On the anatomy of mental processes: A case study of arithmetic. *Journal of Experimental Psychology: General*, **115**, 118-130.

Zemel, R.; Williams, C. and Mozer, M. (1992). Adaptive networks of directional units. *Technical Report CRG-TR-92-2*, University of Toronto.

GLASGOW UNIVERSITY LIBRARY