Optimal Learning and the Spacing Effect:
Theory, Application and Experiments based on
the Memory Chain Model

Artificial Intelligence Master’s Thesis
for Computational Psychology
Viktor de Boer

August, 2003

Faculty of Science
Supervised by J.M.J. Murre and A.G. Chessa,
Faculty of Social and Behavioral Sciences

University of Amsterdam
Abstract

In the growing field of Computer Assisted Language Learning, the optimal distribution of practice time can lead to more efficient learning. The spacing effect ensures us that an optimal distribution of presentations exists. In this thesis, we first describe the spacing effect and a number of qualitative theories on it. We then use a quantitative memory model, the Memory Chain Model, to account for the spacing effect. We then report on the analytical research that we have done on the workings of this model. We also propose a framework for a CALL-application based on this model, which can be used to optimize second language acquisition and we describe an implementation of a part of this framework, the OptiLearn tool. This tool is then used to carry out two online experiments to test this application and gather data on the learning and forgetting speeds of Turkish-Dutch word pairs. The results from these experiments give us insight in the workings and shortcomings of the model in describing the learning of this type of data and we offer suggestions for expansions of the model.
Acknowledgements

First and foremost I would like to thank my two great supervisors, Tony Chessa and Jaap Murre. They provided me with a truly interesting subject for this thesis, which, although maybe its proportions got a bit out of hand, never lost any of its original appeal to me. Tony, with pleasure I look back at our many hours of fitting, simulating and staring at equations together.

Of course, next to thank are the rest of the so-called ‘MurreWurres’ with whom I have had the great honor and pleasure of working with, especially those who were locked with me in the windowless Room 602: Steve, Wouter and Leendert. But also Martijn, Erik, Pedro, Robert, Janneke, Romke and Lucia.

I thank the many participants and piloteers of my experiments, although secretly I have often cursed them for producing the data they did. But that seems to be the faith of the experimental researcher.

Merlijn, thank you for being a great proofreader, most faithfull lunchingpartner and above all, a great friend. Thanks to the rest of the Golden Boys, who have made the first three years of this education so easy. I hope we will keep LANning, eating bitterballen and playing poker. I also want to thank all of my friends, just because they are my friends, and have been for a long time.

Finally I will take this opportunity to thank my family for their unconditional support, both financial and emotional. Thanks Mam, Pap, Hannah and thank you, Oma. And last but not least, thank you Marije.
'Nothing that happens is ever forgotten, even if you don't remember it.'
# Contents

Abstract iii

Acknowledgements iv

Contents vii

Chapter 1. Introduction 1

Chapter 2. Description of past studies and theories 3
   2.1 Spacing effect experiments ............................................. 3
   2.2 Applications .............................................................. 7
   2.3 Categorization of existing theories for the spacing effect ............. 8

Chapter 3. The Memory Chain Model 12
   3.1 Motivation ................................................................. 12
   3.2 General description and basic assumptions ................................ 12
   3.3 Single-store model ...................................................... 14
   3.4 Multi-store memory chain models ..................................... 16
   3.5 Fits on retention data .................................................. 19

Chapter 4. The Memory Chain Model for Multiple Presentations 20
   4.1 The single-store model .................................................. 20
   4.2 The two-store model .................................................... 25
   4.3 Fits on multiple presentation data sets ................................ 30

Chapter 5. Optimizing second language acquisition 32
   5.1 A framework for optimizing second language acquisition .......... 32
   5.2 Implementing part of the tool ......................................... 38

Chapter 6. Experiments 47
   6.1 Testing the implementation: Experiment 1 ............................ 47
   6.2: Results from Experiment 1 .............................................. 48
   6.3 Description Experiment 2 ................................................ 50
   6.4 Results and discussion Experiment 2 .................................. 51

Chapter 7. Conclusions, discussion and suggestions for further research 55
   7.1 Summary and results ...................................................... 55
   7.2 Expanding the model .................................................... 56

Literature 58

Appendix A 60

Appendix B 65
Chapter 1

Introduction

How do people learn things? How much practice or learning time does a person need to obtain the level of knowledge or skill that is needed? What strategy do we use for the distribution of practice time to obtain a certain amount of knowledge? There has always been a lot of interest in the fields of education and cognitive psychology to answer these and similar questions. Classroom teachers make assumptions about the best distribution of practice time to learn and remember the material. Second language courses, whether via tapes or in classrooms use certain assumptions and strategies to distribute the practice time to reach certain levels of lexical knowledge. These assumptions and strategies are often based on past experiences, theories learned in teacher’s education or just gut feelings. But can a solid theory of memory, learning and forgetting be constructed? One that we can use to predict the effects of certain learning strategies and more useful, enable us to directly calculate optimal learning strategies? The level of knowledge or skill a person can obtain in a field is controlled by many factors, like the capabilities of the person, the nature and difficulty of the things to be learned and the available practice or learning time. These factors can often not be modified to obtain better learning. But how can we use the given practice time in a more efficient way to ensure maximum learning effects?

In recent years, with the rise of the multimedia computer and the use of the Internet, the use of computers in education is growing. One of the fields of education where we can see this is (second) language learning. A large number of Computer Assisted Language Learning (or CALL) applications have become available both on commercial CD-ROMs and on the Internet over the last few years, many of them surpassing the ‘novelty’ status and proving to be very valuable. Presentation software and exercise generators can aid language teachers in constructing their lessons, CD-ROMs may be used to replace the traditional tape-and-book methods and online dictionaries form valuable resources. These are examples of existing CALL applications. However, CALL applications for learning second language vocabularies as efficiently as possible are still very limited while their usefulness is obvious: who wants to spend more time learning vocabulary than necessary? Can we construct CALL tools that present us with the optimal presentation schedule so that a in a minimum amount of time, we can learn a maximum amount of second language words?

A few attempts to construct CALL tools that are able to do this have been made and a handful of finished products have become commercially available. SuperMemo and Captain Mnemo are examples of such applications. These will be described in more detail in Chapter 5. The goal of tools for optimally efficient vocabulary acquisition is to make sure that a user acquires a maximum amount of knowledge of a vocabulary within a given time span. For this goal, the computational power of the computer is used to calculate optimal presentation schedules for items in a vocabulary. But first the question rises: what exactly has to be optimized and how can optimal presentation schedules be calculated? In what way does the scheduling of presentations lead to better retention?

In 1885, Ebbinghaus published his well-known works on human memory, in which he recognized the positive effects of distributed (or ‘spaced’) practice rather than ‘massed’ practice. During the next one hundred years the effect, now named the ‘spacing effect’, kept being reported. This positive effect of spaced repetitions has been found in declarative learning as well as in gaining procedural knowledge, like learning algebra, grammars or motor skills.
Although there has been a great deal of research on the spacing effect, most of the studies are descriptions of experiments trying to find the effect in different situations. There are very few studies on why and how it occurs. And the studies that try to give cognitive explanations for the effect stick to vague qualitative descriptions and fail to give solid quantitative explanations, based on consistent cognitive models. Therefore from these loosely defined theories, no optimal learning schemes for specific learning situations can be calculated. In this thesis, we will describe a research that does try to remedy this: We will examine the possibility of calculating optimal learning schemes for specific second language vocabulary learning situations. This is done by calculating optimal spacing of presentations, based on a consistent mathematical model of learning and forgetting. The model we will use is a modified version of the Memory Chain Model of learning and forgetting as described by J.M.J. Murre and A.G. Chessa (Chessa & Murre, submitted; Murre & Chessa, submitted). We will construct a first version of a CALL application (which we have called OptiLearn) that calculates optimal presentation schedules for second language vocabulary learning. This application will be tested using human data obtained from two experiments. We now give a short overview of the following chapters.

In Chapter 2, we will first give an account of previous research, both experimental and theoretical, in the field of human memory, learning and forgetting, focusing on the spacing effect. Because there has been a great amount of research in the field we will describe only a few of the many studies.

In Chapter 3, we will describe the Memory Chain Model. This model serves as the basis for our consistent quantitative model describing the spacing effect that is used to calculate optimal presentation schedules. We will look at the original model, its variations and assumptions for a single presentation and we will describe in some detail the encoding and decline of memory traces and the form of different retention functions in this model.

In Chapter 4, we will then describe the model with the expansions that allow it to describe learning and forgetting for more than one presentation, so that we can use it to describe the spacing effect and make calculations about it.

In Chapter 5, we describe a framework for a CALL-program, which uses the expanded model as described in Chapter 4 to calculate optimal presentation schedules for different language learning situations. In the same chapter we will also describe our implementation of a part of that framework.

In Chapter 6, we give an account of two experiments that were conducted by us to test this implementation. We will give a full description of the experiments itself, the subjects tested, the way of testing and the materials used, and we will give the results of the experiments, which will be discussed in the same chapter. Conclusions drawn from these experiments on the validity and applicability of the expanded model and the implementation also will be included in Chapter 6.

Finally in Chapter 7 we will summarize the results and conclusions and we will give suggestions for further research.
Chapter 2

Description of past studies and theories

In this chapter we will give a summary of research, both experimental and theoretical, that has been done in the field of human learning and forgetting, focusing on the beneficial effect of spaced presentation on learning, the spacing effect. In Section 1, we will describe experiments that have been influential in the field of research on the spacing effect, and experiments that are used to create our own experiments, which are described in Chapter 5. In Section 2 we show some possible applications for the spacing effect and Section 3 is used to describe and categorize the different existing theories on the spacing effect.

2.1 Spacing effect experiments

The positive effect of distributed practice on learning was first noted by the German psychologist Hermann Ebbinghaus. Ebbinghaus is famous for his influential body of work (‘Über das Gedächtnis’) published in 1885 and he is generally credited as being the first to systematically conduct scientific studies of human memory and learning. In those days, psychologists were not used to test theories by doing experiments with subjects, so Ebbinghaus performed most of his experiments on himself. In several experiments he taught himself lists of hundreds of nonsense syllables of a vowel-consonant-vowel form (such as “dax”, “zeb” etc.) under various conditions. He then, using various delays, measured the time he needed to relearn the lists in order to reach his standard criterion of two perfect recitations and derived forgetting percentages from the savings scores he measured. The delays Ebbinghaus used ranged from 20 minutes to 31 days. These experiments produced retention curves like the one in Figure 2.1.

![Retention curve](image-url)  
Figure 2.1: Shape of the retention curve, as found by Ebbinghaus (1885) and others (e.g. Wixted & Ebbesen, 1991).
Ebbinghaus, and later researchers, found that the overall shape of the retention functions was the same on different time scales, whether the shorter or longer delays were used (see also Wixted & Ebbesen, 1991).

In the experiments that he carried out in this way, he also found that if he distributed his relearning sessions over a larger period of time he could save almost half the total learning time. In these experiments, Ebbinghaus gave the first real scientific description of the positive effect of distributed practice, the spacing effect: “Even if one makes very great concessions to the uncertainty of numbers based on so few researches, the difference is large enough to be significant. It makes the assumption probable that with any considerable number of repetitions a suitable distribution of them over a space of time is decidedly more advantageous than the massing of them at a single time.” (Ebbinghaus, 1885).

The Glenberg experiment

In the period after Ebbinghaus’ experiments the spacing effect was reported by a number of scientists, though very few experimental studies examining the spacing effect that were conducted were truly useful, that is, containing data that could be used for testing theories of the spacing effect. Most of these experiments spoke of the effect in vague, qualitative terms. Only in the 1960s and ’70s the first experiments to produce quantitative data were conducted. A highly influential set of experiments, of which the data is still used, was done by Glenberg (1976). Glenberg conducted a number of experiments to describe the effect of distributed learning. In one of these experiments (Experiment I) a so-called continuous paired-associate design was used. The items to be learned consisted of pairs of four letter nouns. Glenberg visually presented his subjects the stimulus part of the pair and subjects had to respond by orally giving the response part. Regardless of the correctness of the response, the subjects were given the actual correct answer. In this form each trial was both a test and a learning of the item. Each item was presented three times, the first two presentations were used for the learning of the items and the final one was used for testing the final knowledge of the item. The two learning presentations were presented with various intervals (0, 1, 4, 8, 20 and 40 intervening items) and these pairs were then tested using test lags of 2, 8, 32 or 64 intervening items. This method has many advantages; one of the advantages is that the usual interfering effects of list structure learning are eliminated. The results of Glenberg’s first experiment are shown in Figure 2.2.

![Figure 2.2: Results of Glenberg’s Experiment I (Glenberg, 1976). The figure shows the average recall scores for different presentation lags. This is done for four different retention or test lags.](image-url)
The results show that the effects of the test lag and presentation lag are significant. This effect is best seen when we look at the curve for the items that were tested after a test lag of eight intervening items. If the two learning trials are presented with a presentation lag of one intervening item, only 42 percent of the items are recalled correctly. If the presentation lag is four items, 50 percent is correctly recalled and with a presentation lag of eight a score of 52 percent is reached. If even larger presentation lags are chosen, the percentage drops again, indicating that, for a certain test lag, there is a presentation lag for which an optimal recall score can be reached. For the other test lags similar effects can be seen. We can also see that for increasing test lags the presentation lags for the optimal retention scores seem to increase.

Other studies
After the existence of the spacing effect had been established, researchers began to explore its domains. Various experiments were set up to try to find out in what situations the spacing effect occurred and tested the applicability of the effect. A number of these studies and their findings will be described below.

First we will give a short description of a set of experiments conducted by Rumelhart (1967). These experiments are also used in the design of our own experiments. In Rumelhart’s unpublished paper, he describes two experiments to measure the effect of interpresentation intervals on performance in a continuous paired-associate task. These were used to test various learning-state theories that we will not describe here. In both experiments a paired associate design was used, the stimulus part consisting of three-letter syllables and the response part of a ‘3’, ‘5’ or ‘7’. In the first experiment (Experiment 1), the subjects were given six presentations of various items. These six presentations were presented in a given schedule with longer and shorter lags (from one to ten intervening items). The last presentation was analyzed as a recall test. His findings suggested a spacing effect, but to show this more explicitly he designed a second experiment. This experiment (Experiment 2) had the same setup, but Rumelhart designed nine different schedules to be used. Of these nine schedules, six consisted of six presentations, whereas the remaining three had four presentations of an item. The six schedules with six presentations were presented in a massed (one intervening item) or a spaced fashion (ten intervening items). There was also variation in the length of the final test lag; this could be 32, 16 or 2 intervening items. Rumelhart found a strong spacing effect in the schedules where the final test lag was 32. The effect was still noticeable in the test lag of 16 items. In the schedule using the smallest test lag of 2 intervening items, the massed schedule performed better than the spaced one. In the last three schedules consisting of four presentations, the effects were inconsistent to this interpretation but Rumelhart showed chance deviations could account for these discrepancies. One thing this experiment shows, as did Glenberg’s experiment, is that spacing effects are stronger when the test lag is larger.

Cuddy and Jacoby (1982) reported some experiments in which they varied factors such as the similarity of the repetitions, the type of intervening material and cue effectiveness and revealed the interactions between those factors and the spacing of the repetitions on recall probability. They found for instance that similar material intervening between item presentations results in higher recall at a later test. Cuddy and Jacoby also found that when the form of the second presentation of an item was slightly varied from the first presentation, the effect of the spacing of the repetitions was also increased.

Another researcher, Robert L. Greene, showed that spacing effects are eliminated in recognition tasks when items are learned incidentally and not in the list-like forms of Glenberg’s experiments (Greene, 1989). He also conducted a number of experiments demonstrating the spacing effect on different memory tasks: The spelling of homophonic words, word-fragment completion and
perceptual identification. In one experiment (Experiment 1) Greene demonstrated a clear spacing
effect on an implicit memory test using homophonic stimuli, showing that the spacing effect also
held for implicit memory, the unconscious learning of items. Greene’s other experiments
elaborated these results and found that in various implicit memory tasks the spacing effect
occurred (Greene, 1990).

Bjork and Allen (1970) published a paper describing an experiment using trigrams of four-letter
words varying not only the spacing of the items that had to be learned (short or long presentation
lags), but also the difficulty of the activity interpolated between the two presentations of an item.
They also varied between a long and a short test lag. They too found a significant spacing effect.
The items presented in a massed schedule had an average of 60 percent more errors in recall. But
they also found that a short interval filled with a relatively difficult task simulated the same
effects on recall of a longer interval filled with an easy task, thus demonstrating a trade-off
between difficulty of interpolating tasks and interpolating task durations.

**Long-term experiments**

Although most studies of the spacing effect examine its workings over relatively short periods of
time (shorter than one week), the few exceptions to this rule are interesting and useful for real-life
situations outside the laboratory.

Of these long-term experiments, the most famous one is the experiment done by Bahrick,
Bahrick, Bahrick and Bahrick (1993). They conducted an influential experiment in which they
themselves learned 50 Spanish-English word pairs with presentation lags of 14, 28 or 56 days.
Their test lags were even more astonishing: they tested their knowledge after 1, 2, 3 or 5 years. In
the results of this experiment the spacing effect is very evident. For instance, after five years, the
retention score for the items learned with the presentation lag of fourteen days was 33 percent,
whereas the score for the longest learning interval was 55 percent.

Another long-term research comes from Baddeley and Longman (1978). They tried to find out
what the fastest method was for employees of the British postal services to learn to type British
area codes. In this case the researchers varied the way of distributing the fixed total learning time.
They found that with distributed practice (one hour of learning per day, as opposed to the massed
learning schedule of two hours per day) the test subjects had much better results when tested after
three, six and nine months.

These examples show us that the spacing effect is a universal one that applies not only to short-
term learning but long-term learning as well.

To show that the spacing effect appears in many fields, we give yet another long-term example:
the Zielske experiment conducted in the 1950s (Simon, 1979). Zielske was the first to study the
difference in effectiveness of spaced and massed advertising. Subjects were divided into two
groups. The subjects in one group were sent printed advertisements once a week (the ‘massed
group’), while the subjects in the other group received the same material once every four weeks
(the ‘spaced group’). Both groups received the same material up to thirteen times. Subjects were
interviewed about their knowledge of the advertisements on various moments, not only after the
last week of advertising, but also during the “learning” period. In this way, the researchers could
acquire data on both learning and forgetting of the advertisements. This data is shown in Figure
2.3. As we can see, the massed group reached a higher peak in recall, but after 17 weeks, the
spaced groups have a better recall than the massed group, and the recall stays at that high level. It
is fairly easy to see that the total recall percentage over 53 weeks is higher for the spaced group
than it is for the massed group.
Optimizing second language acquisition

Another experiment that we will describe because of its interest for our own experiments is that of Atkinson. He tried to optimize the learning of a large German-English vocabulary (Atkinson, 1972). He proposed and tested four different learning strategies and evaluated them experimentally. The first strategy was to present the items randomly, this could then serve as a benchmark for the other three strategies. In the second strategy, the subject could determine which item was presented on a certain trial, thus creating a ‘learner-control’ strategy’. The third and fourth strategies were calculated by a computer program based on Atkinson’s decision-theoretic analysis of the instructional task (Atkinson, 1972). One of these strategies assumed that items were of equal difficulty; the other one used variation in difficulty level among items. The subjects were tested after one week. The effects that were found are significant: The subjects using the self-selection strategy or the strategy in which the items were assumed to be of equal difficulty performed approximately 50 percent better than the ones using the random selection strategy. The subjects learning according to the computer calculated strategy using the varied item difficulty performed with a relative gain of 108 percent. This proves, among other things, that the optimization of second language acquisition is indeed possible using memory models.

2.2 Applications

After the existence of the spacing effect had been established, people began to explore its uses. Some of these possible uses will be described below.

As we noted in Chapter 1, an obvious use for the effect lies in the optimization of second language vocabulary learning. This is where most of the suggested applications focus on. A lot of language courses use some sort of principle of spaced repetitions in their programs. More recently, electronic vocabulary teachers, online or offline, have become popular. Many of these programs take the spacing effect into account, such as Wozniak’s SuperMemo-program and the Captain Mnemo project of J.M.J. Murre.

In 1988, Dempster, a spacing effect researcher focusing on studying its possibilities for practical application, mainly in classroom education, published a paper on the spacing effect. In this paper he tried to find explanations for the fact that although the spacing effect, among other research
findings, are often not utilized by teachers and curriculum makers. He considered nine possible impediments to the use of research findings in general in the classroom. Of these nine, he found that five could apply to the spacing effect: The fact that most research is of relatively recent date, the failure to find the effect in school-like activities, a scarcity of impressive classroom demonstrations of the phenomenon, limited knowledge of classroom practice and an incomplete understanding of the psychological bases of the spacing effect. Dempster shows that the first four possible explanations for the lack of use of the spacing effect in classroom education do not apply. Dempster concludes that lack of knowledge of what causes the effect accounts for most of the resistance against using the spacing effect in classrooms, and more importantly in teachers’ education (Dempster, 1988).

But there are other fields in which the spacing effect can be used. One of these is in the scheduling of advertisements, whether they are written or broadcast on television, radio and of course on the Internet. In the preceding section we described the experiments of Zielske in the 1950’s studying written ad scheduling. A more complete knowledge of the spacing effect could lead to more effective advertisement campaigns, a reason for commercial companies to take an active interest in research on the subject (Janiszewski, Noel & Sawyer, 2002).

2.3 Categorization of existing theories for the spacing effect

There are different theories that try to explain the spacing effect, but it is difficult to recognize the differences and similarities between the individual theories or to identify clear classes of theories, mainly because of the lack of use of common terminology and the often vague definitions of the theories. Here we will try to give an account of the most popular categorizations of theories explaining the spacing effect.

Two classes

Bjork and Allen (1970) observe that the theoretical explanations of the spacing effect fall into two classes: those that attribute the advantage of two spaced presentations over two massed presentations to a better consolidation of the first presentation, and those that attribute the advantage to better encoding of the second presentation. The first set of theories assumes some sort of consolidation effect that occurs during the interval between presentations. The longer the interval between presentations, the greater the consolidation of the long-term memory storage of the item during the interval will be (Bjork & Allen, 1970). This theory focuses on the storage of the first presentation. If a second presentation of an item is offered ‘too closely’ following the first presentation, the storage of the first presentation may be disrupted and item recall will eventually be more difficult than when the presentation lag is greater. These consolidation theories are often very vaguely defined on how they exactly apply to the spacing effect.

The second class of explanations attributes the spacing effect to an assumed variation in the encoding of the second presentation of a stimulus as a function of the time that has past since the first presentation of the stimulus. This is the class of encoding variability theories, also known as differential encoding theories. This class of theories assumes that the longer the presentation lag is, the greater the probability that the encoding of the second presentation will be a new one, one that will be different from the encoding of the first presentation. There are in one form or another many different varieties of these theories and some are better and more elaborately described. As Greene (1989) shows us, most of these encoding variability theories can be placed in one of two subclasses. They are either a form of the component-levels theory, or of the deficient-
processing theory. The component-levels theory explains the spacing effect by hypothesizing that when information is presented in a more distributed manner, the subject associates the information with more contextual, structural and descriptive components. The deficient-processing theory states that, when the information is learned in a more spaced manner, the subject will make a new encoding of the information, thus leading to a better retention. If the information is presented in a massed schedule, on the second presentation it will be treated as an item that is already known and the item will not receive much attention on that presentation. We will further describe these two classes of theories below.

The component levels theory
This theory explaining the beneficial effects of presenting information in a more spaced manner focuses on the period between two presentations of an item, the so called presentation lag. Glenberg (1979) originally proposed this theory and our description mainly follows his paper. He hypothesizes that, when the period between these presentations is larger, the conditions in which the presentations of the items occur are more varied. When this is the case, the information can be linked to more different components, being contextual components, structural components and descriptive components.

The easiest to understand this is when we look at contextual components. The contextual components represent the context in which the presentation of the information is encoded and include things like the physical surroundings in which the item is presented, as well as the time and the cognitive and emotional state of the learner. This contextual component is registered automatically and if information is presented multiple times, contextual components will be stored each time. The more time there will be between two separate presentations of an item, the more different contextual components will be associated with the item and the greater the chance of recall of that item will be.

The second class of components is the class of structural components. When new information is presented to a subject, the subject tends to try to categorize information and seek structure in it as it is being learned. If the subject imposes a structure on learned elements of information, the elements will be chunked together into a cluster. It is easier to retain information in one cluster than when the elements have to be stored individually. Under influence of the context the specific structural components will be included in the memory trace. At the repetition, these structural components will only be formed under certain learning circumstances. The learner must be engaged in some sort of structural analysis if new structural components are to be added to the trace of the repetition (Glenberg, 1979). If this is the case, then the longer the presentation lags, the more different structural components will be stored in the trace of the repetitions, thus generating a beneficial spacing effect.

The third type of components is that of descriptive components. Descriptive components include information about the articulation, meaning, connotation and so on. While the contextual and structural components will differ on each presentation of an item because they are formed anew at each presentation of the stimulus, this information is retrieved from the semantic memory and it will, if a verbal item is presented, contain information about its pronunciation, spelling, possible meanings and so on. The specific components copied into the memory trace of a stimulus depend on the presentation lags and the local context. And again it is the case that the longer the interval is between two presentations, the more descriptive components will be formed. This can be realized because of changes in the local context in which the encoding takes place, in which case different descriptive components may be formed, or because the control processes between presentations make a subject encode different descriptive components.
The trace of a stimulus repeated after a substantial lag will include more contextual, structural and descriptive components than the trace of an item repeated after a short lag (Glenberg, 1979). The importance of the different components on recall also depends on the type of recall subjects are tested with. One way of testing memory is cued-recall tests. Here subjects first receive a ‘hint’ or the stimulus part of a paired-associate like Glenberg’s experiments mentioned earlier in this chapter. They then have to reply by giving the correct answer. In this form of testing, the descriptive components are the most important components because they are the most item-specific components and will facilitate better recall of the cued item. The other way of reproducing memories in tests is by using free recall. Subjects have to reproduce items without being given hints or cues. In this form, contextual components play the biggest role in retrieval, because they are less item-specific than the descriptive components. The contextual components will be associated with a larger number of items. Because the contextual components are the most important and significant components, this class of theories is often also referred to as the class of contextual-variability theories (e.g. Greene, 1989).

The deficient processing theory
The theories of the component-levels class assume that the further two learning sessions are separated in time, the more different components will be formed in the memory trace, because the environment in which the presentations occur will differ. The deficient processing theories claim that the second occurrence of an item repeated in a massed fashion is not processed as thoroughly as a second presentation in a more spaced schedule. There are two separate mechanisms that can lead to this effect. The first is a conscious choice the subject makes, when confronted with an item the subject has already seen a relatively short time before this second presentation. The subject will assume it already has seen and processed this item, that he already has correctly memorized it. He will not give the second presentation in a massed schedule as much attention as a second presentation in a spaced schedule. The other mechanism is an involuntary one. In many learning occasions, the second occurrence of a stimulus causes retrieval of the first presentation. If this retrieval of the item is more difficult, then the probability of the item being remembered during a later test is assumed to increase (Braun & Rubin, 1998; Greene, 1989). The main difference with the component-levels theories is that theories do not assume that with spaced learning sessions more different components are added to the trace, but that one repeats the same mental activity that was done on the first presentation, so instead of attributing the beneficial effect of spacing on more variation, the deficient processing theories focus on repetition (van Kreij, 2000).

Another, and probably clearer description of a deficient-processing theory is the one in Cuddy and Jacobi (1982). They pose that when two presentations of the same item are presented with a considerable presentation lag, the memory of information learned in the first presentation slowly fades away during that interval. When the item is presented a second time, the first presentation is not readily accessible and the information is more or less ‘forgotten’. The subject has to actively process the item again and must make a totally new encoding of the item. This will eventually lead to a better retention. As mentioned before, in their article, Cuddy and Jacobi argue that the manipulation of the presentation lag, the memory of information learned in the first presentation slowly fades away during that interval. When the item is presented a second time, the first presentation is not readily accessible and the information is more or less ‘forgotten’. The subject has to actively process the item again and must make a totally new encoding of the item. This will eventually lead to a better retention. As mentioned before, in their article, Cuddy and Jacobi argue that the manipulation of the presentation lag, which corresponds to manipulating the retention lag for the first presentation, is only one way to vary the accessibility of memory for a prior presentation. The positive effect of spacing will be much stronger if the intervening material presented during the presentation interval is roughly similar to the ‘target information’ because this information is far more confusing to the subject than other intervening information. For instance, when testing the retention of repeatedly presented word-pairs, in a spaced manner, the cued recall performance will be much higher when the intervening material also consists of similar word-pairs. Another way of making the information learned in the first presentation less accessible is by varying the
form of a presentation during the presentation lag. If this is done, the repetition of the item will lead to greater processing and thus better retention (Cuddy & Jacoby, 1982).

None of the abovementioned theories originate from a sound model of memory. To make quantitative claims about optimal spacing of learning trials, which we want to do later on in this paper, we cannot use any of these vaguely defined models. What these models can provide us with is clues about the theoretical assumptions we have to make in constructing a memory model for multiple presentations that we can use for this exact purpose. We will however, have to come up with a sound mathematical model of memory.
Chapter 3

The Memory Chain Model

In this chapter we will give a description of the Memory Chain Model, as developed by A.G. Chessa and J.M.J. Murre. Most of the information from this chapter is extracted from their papers on that model (Murre & Chessa, submitted; Chessa & Murre, submitted; Chessa & Murre, 2001), and in this chapter no further explicit references to their papers will be given. We will first look at the motivations that led to the development of this model. In Section 2 we give a general description of the model. In Sections 3 and 4 we will describe the model in more detail and in Section 5 we look how the model fits to retention data sets.

3.1 Motivation

Since Ebbinghaus (1885) there has been extensive study on the mathematical form of the retention function. In 1996, Rubin and Wenzel published a paper describing the results of fitting more than one 100 different retention functions to 210 datasets from the retention literature. None of these functions alone could account for all the datasets, but a set of four functions was found that fitted most of the datasets, with the exception of autobiographical data (Rubin & Wenzel, 1996). The observation that a limited set of functions could describe most retention datasets motivated the development of one unified model-based theory describing retention in the human mind. This memory model, the Memory Chain Model, was developed to have a mathematical base describing encoding, storage, retrieval and recall. A mathematical memory model can be used to make calculations about future retention levels and optimal rehearsal patterns under different circumstances. These findings can be used to support decisions about the distribution of practice, for instance, in an educational setting.

3.2 General description and basic assumptions

The Memory Chain Model can be viewed as the result from a mapping from a neural network model of learning and forgetting. The model assumes that an encoded memory of a given item consists of a countable number of representations (or copies, critical features or components as described in Glenberg, 1979) of that memory. Each of these representations captures one or more characteristics of the item and retrieval of one or more of these representations is assumed to be sufficient to retrieve the entire memory. The more of these representations are present at a given time, the stronger the encoding of the memory is. The second main assumption of the model is that these memory representations are stored in one or more ‘memory stores’ that denote abstractions of neurobiological processes that form short-term and long-term representations of a memory. These memory stores are linked together in a feedforward manner. Two features characterize each store: a decline process, which determines the speed of memory loss in that store and an induction process, which handles the copying of memory representations from one store to the next one. It is assumed that each store has a slower rate of decay than the one preceding it. So the stores are linked according to decreasing decline rates. In Figure 3.1, a schematic view of the model is given.
Since memories are represented as a countable number of features, a suitable mathematical model for describing memory is offered by the class of models known as point processes. Roughly speaking, point processes are stochastic processes that describe how points are distributed in a space of a certain dimension; the only dimension in the Memory Chain Model is time. More precisely, a point process can be regarded as a collection of random variables, which in the Memory Chain Model count the number of memory representations (‘points’) in time-intervals. A ‘point’ then represents the time at which a memory representation is available for retrieval. In a neurobiological setting, this could refer to an ensemble of firing neurons at a certain time. The probability distribution of a point process is completely determined by the joint distributions of any finite collection of random variables (Daley & Vere-Jones, 1988). The point process that we will consider throughout this thesis (the Poisson point process) is completely specified by the expectation measure $E(M)$ of a point process $M$. In this specific context it is given by Equation 3.1:

$$E(M(A)) = \int_A r(t)dt$$

for time-intervals $A$ and a density function or intensity function, $r$, which depends on the number of memory, stores. This point process has the property that the random variables are distributed according to a Poisson distribution.

The stochasticity of the process represents the variability in the availability of memory representations due to factors such as neural noise or context changes, which may alter the outcome of the same task.

Point processes are used in a broad range of fields. For instance the locations of many of the objects studied in ecology are represented as point process. Examples include trees, shrubs, weeds, fungi, and nests belonging to animals and birds. Point processes are also used to describe incidence of disease, the occurrences of fires, earthquakes, lightning strikes, tsunamis, or volcanic eruptions. As stated before, in neurobiology, a point process can describe the neural activity in the brain (Abeles, 1991). In the Memory Chain Model, point processes will be used to describe the encoding and storage of memory representations in the memory stores.

To attain a better understanding of the model, we now first describe a single-store model, in which there is only one memory store and all memory representations are located within this store. After we have established the workings and implications of this single store model for memory and retention, we will describe models with more than one store. In this chapter we will describe only the base model. This base model describes what happens if an item is presented to a subject only once. In Chapter 4 we will describe an expanded model for more than one presentation of the same item.
3.3 Single-store model

For our model a point in the point process denotes the time at which a memory representation of an item is available. This means we describe the occurrence of memory representations on the time dimension. When learning an item, for instance when a person is presented with a word or picture, the number of memory representations in a store increases to a mean value of $\mu$ per time unit. This value $\mu$ is reached at the end of an item presentation, which is at the end of the learning phase. It represents the intensity of the initial encoding of the memory$^1$.

We fix the end of the learning trial at time $t = 0$. If we assume that no forgetting or consolidation of a memory occurs, we can use a homogenous or a stationary Poisson point process $L$ with intensity $\mu$ to describe what happens to the memory representations in a store in time. The encoding will not change on average after $t = 0$ (A stationary Poisson process implies that a given number of $n$ points is distributed uniformly and independently). This situation is shown in Figure 3.2.

![Figure 3.2: Intensity function for a single-store model without forgetting before, during and after the item presentation. Also depicted is a schematic view of the gain of memory representations (or 'points') in the store during the item presentation.](image)

A stationary Poisson point process has two well-known properties: numbers of representations in time-intervals have a Poisson distribution with mean $\mu$ for interval of unit length and numbers of representations in disjoint time-intervals are independent. We can derive all point processes of our extended models from this stationary memory model.

---

$^1$ $\mu$ is therefore determined by two factors: the duration of the presentation $l$, and the learning speed $\nu$. 

14
Decline

Our base model can now only describe learning. We now elaborate the model to include memory decline (forgetting). This results in a straightforward refinement of L. The average number of representations will no longer be invariant in time. The model is expanded by introducing a decline function $\tilde{r}(t)$. This denotes, for every $t \geq 0$, the probability that an initially encoded memory representation is still available at that time, which decreases as $t$ increases. The longer ago the memory representation was formed in the store, the greater the chance that it has already disappeared. With the assumption that representations are available in time independently of each other, the new point process $M$ is a Poisson process with an intensity function $r$ that has the form:

$$r(t) = \mu\tilde{r}(t)$$

For the decline function an exponential function with decline parameter $a$ is assumed. The decline function is denoted as:

$$\tilde{r}(t) = e^{-at}, \text{ where } a \geq 0.$$  \hfill (3.3)

This is the entire model for describing the encoding and storage of memory in a single store. Figure 3.3 shows how the encoding of a memory declines as time progresses.

---

1 An exponential decline function is used to keep multi store processes computationally tractable. Exponential decline functions are often used in other memory models.
Recall Probability

To obtain recall probabilities from this model, we make an additional assumption about the conditions under which successful recall takes place. Throughout this dissertation, we assume that the individual results at recall tests are binary values (1 for ‘correct answer’, 0 for incorrect). To derive recall probability we must introduce a recall threshold $b$, which is the number of memory representations needed for successful recall. From the Poisson distribution follows that the probability that at least $b$ representations are retrieved is:

$$p(t) = 1 - \sum_{n=0}^{b-1} \frac{r(t)^n}{n!} e^{-r(t)}.$$  \hspace{1cm} (3.4)

This is our recall probability function. Note that $p$ is a monotonous transformation of $r$. The default value for $b$ is 1. For this value, the recall probability $p$ simplifies to:

$$p(t) = 1 - e^{-r(t)}$$ \hspace{1cm} (3.5)

Some function characteristics for $p$

These function characteristics all assume a default value of 1 for $b$, the recall threshold. The initial recall level, which is the recall probability at the end of the learning phase at time $t = 0$, is equal to $p(0) = 1 - \exp(-\mu)$. The choice for the decline function has no effect on this, as decline has not occurred yet. If $\mu$ tends to infinity, $p(0)$ tends to one, which describes situations of (extreme) overlearning.

We can also derive what happens when $t$ tends toward infinity. If $a > 0$, then $p$ tends to zero. If $a = 0$, the recall probability will remain constant at its initial recall level $p(0)$.

Because $r$ decreases in time, $p$ also decreases. If $\mu$ tends to infinity (again the case of extreme overlearning), the derivative of $p$ tends to zero for every $t$ and the recall probability tends to 1 for every retention interval. Thus memory loss is reduced after learning and complete memory loss is delayed with intensified learning.

3.4 Multi-store memory chain models

In the psychology of memory the distinction between different types of memories or memory stores is often made. This can be the distinction between iconic and episodic memory, but also short- and long-term memory and working memory. In the Memory Chain Model, these are called ‘stores’. Memories that first only exist in one store can be strengthened and copied to other stores, for example from short-term memory to long-term memory. One way to describe these copies is by induction through activation. These induction processes are viewed here as a feedforward chain of memory stores, between which induction processes occur that take place at different time scales. As we have noted before, it is assumed that each store receives its memory representations only from its preceding store in the chain and that memory representations have a smaller decline rate in the new store.

There are examples in the human brain where rapidly decaying neural groups activate neurons in a less rapidly decaying part of the brain. The Memory Chain Model is an abstraction of these kinds of neurobiological observations.
Decline and induction in two-store models

To describe the model with more than one memory store, we will first describe a model with only two stores. First a slight modification of the notation is necessary. In order to distinguish between parameters from different stores, subscripts will be used to label the different parameters. Thus we speak of the point process $M_1$ with intensity function $r_1$, decline parameter $a_1$ and mean initial encoding $\mu_1$. The point process $M_1$ describes the encoding, storage and retrieval of memory representations in the first memory store.

It is assumed that new memory representations in a second store are formed by an induction process. This is described as a point process where the ‘points’ represent the times at which memory representations will be induced to the second store. These induction times are distributed according to a Poisson process with intensity function $\mu_2 r_1(t)$. This Poisson process is assumed to be independent from $M_1$. We can see that the induction times are determined by the intensity of memory representations in the first store and a time-independent parameter $\mu_2$ ($\mu_2 \geq 0$), which denotes the induction rate from the first to the second store. If $\mu_2 > 1$, a strengthened version of the memory representation in the first store will be copied to the second store. If $\mu_2 < 1$ a partial version of the representation will be copied and if $\mu_2 = 0$ no induction at all takes place. This induction rate depends on the possibility of rehearsal between item presentation and test.

In this second store, a decline process also occurs, similar to the one in the first store. It is assumed that the second store has a slower rate of decline than the first store, so $a_2 < a_1$. This two-store system describes the simultaneous decline of memory representations in Store 1 and the copying of representations to Store 2, as well as their decline in Store 2.

These assumptions lead to a Poisson process $M_2$ for the memory representations in the second store. This Poisson process is independent of the one in Store 1. The intensity function $r_2$ of $M_2$ is formed by the encoding and decline of memory representations in the first store, the induction process from Store 1 to Store 2 and the decline in Store 2. We can derive the intensity function $r_2$ as the convolution of $\mu_2 r_1$ and the exponential decline function of Store 2, $\tilde{r}_2$. The function, $r_2$ can be written as:

$$ r_2(t) = \int_0^t \mu_2 r_1(z) \tilde{r}_2(t-z) dz, \quad (3.6) $$

which translates to:

$$ r_2(t) = \frac{\mu_1 \mu_2}{a_1 - a_2} (e^{-a_2 t} - e^{-a_1 t}). \quad (3.7) $$

Stores represent disjoint subsets of a space that is associated with the brain. Since point processes are measures, these are additive. The linearity of the expectation measure implies that the intensity function $r$ over all stores (in our case, two) is equal to the sum of the intensity functions $r_1, \ldots, r_S$ for the individual processes of the stores $1, \ldots, S$.

In this case for only two stores we can therefore denote the total intensity function as: $r(t) = r_1(t) + r_2(t)$.

We can see that from Equation 3.6 follows that a general characteristic of this induction process is that Store 2 will be empty at $t = 0$. If $a_2 = 0$, then $r_2$ will increase to an asymptote. If $a_2 > 0$, then

---

1 See also Chessa and Murre (unpublished).
a maximum intensity will be reached after which the intensity will approach zero as \( t \) increases. Also note that all parameters are assumed to only take on positive values. Figure 3.4 gives a schematic view of the two-store memory process.

**Multi-store models**

Now that we have shown how a two-store model works, we describe a model with an arbitrary number of stores. Every store has an induction process similar to the one in the two-store model. For \( s \) stores we now have independent Poisson processes \( M_1, ..., M_s \) with intensity functions \( r_1, ..., r_s \). Recall is determined by all the memory representations in all stores taken together and so the process \( M \) that describes all this is \( M = M_1 + ... + M_s \). \( M \) is also a Poisson point process (\( M_1, ..., M_s \) are independent) and its intensity function follows from the linearity property of expected values, so that \( r(t) = r_1(t) + ... + r_s(t) \).

If we assume the recall threshold is set to one, still only one memory representation has to be retrieved from the memory stores, so \( M \geq 1 \). The recall probability function for multi-store models therefore has the same form as for a single-store model. So, with \( b = 1 \) this is: \( p(t) = 1 - \exp(-r(t)) \), where, \( r(t) = r_1(t), ..., r_s(t) \). Again, \( p \) is a monotonous transformation of \( r \).

**Some function characteristics for \( p \)**

Again, for all function characteristics we assume \( b = 1 \). The initial recall level is the same as in the single-store model: \( p(0) = 1 - \exp(-\mu) \). We can see this has to hold because only the first store contains memory representations at \( t = 0 \) and all other stores are still empty, and no induction processes have occurred yet.

We can also look at what happens if \( t \) tends to infinity. If all decline parameters are non-zero, then the recall probability will tend to zero. If one or more of the decline parameters is zero, then the
recall probability will tend to a non-zero asymptote. For example, in a two-store process with $a_2 = 0$, the asymptote will be $1 - e^{-\mu_1 \mu_2 / a_1}$.

If we look at the recall probability in the single-store model, we can see that it is decreasing in time. This does not have to be the case for the multi-store models. If we focus on two-store models, optima and extreme values can be calculated. In a two-store model, if and only if $\mu_2$ exceeds $a_1$ (so more representations are copied from the first to the second store than are lost in the first), $p$ will reach a maximum at retention intervals $t > 0$. In this case, the mean overall number of representations in both stores will increase for a while until it reaches its maximum, after which overall decline will make the recall probability decrease.

### 3.5 Fits on retention data

Chessa and Murre (unpublished) tested the validity of the Memory Chain Model, which was fitted to 153 different retention data sets. These data sets were chosen mainly from the data sets in Rubin and Wenzel (1996), though most of the autobiographic and amnesia data sets were omitted. On all datasets three different versions of the model were fitted: a single store, a two-store and a three-store model. Together with these functions, three of the four retention functions Rubin and Wenzel found as best fitting to their data sets were fitted in the same way as a point of reference: the power-law, the exponential-power function and the logarithmic function. These three functions provided the best fits to the data sets according to Rubin and Wenzel.

The exact results of this fit can be found in Chessa and Murre (unpublished). Over all data sets, the percentage of non-rejections of fits (with $\alpha \geq 0.01$) of the two and three store models were respectively 83.0 and 87.7 percent, using $X^2$-test for the goodness-of-fit. This exceeded the number of non-rejections of the power-law, the exponential-power function and the logarithmic function (respectively 48.9, 56.4 and 53.2 percent) by far. The single-store model had 35.1 percent fits that were not rejected. This is a good indication that the (multi-store) Memory Chain Model can describe retention data at least better than the functions that are commonly used to account for them.

Another indication to this is that the three ‘traditional’ retention functions, even when they produce a good fit, do not always follow the data correctly. For instance, none of these functions is able to produce a ‘flex point’, a point where the second derivative of the function is equal to zero. In our case, this indicates a point in time where the decline speed in recall probability has a maximum. In many data sets, such a flex point can be seen in data points (e.g. Waugh & Norman, 1965). The Memory Chain Model with even only one store is capable of producing these flex points.

A third point to illustrate the superiority of the Memory Chain Model on fitting these data sets is that the same model is used to describe memory processes that act on totally different time scales (from a few seconds to years). The more traditional functions fail to describe both long- and short-term data at the same time.

---

1 All data sets were fitted with $b = 1$. 

19
Chapter 4

The Memory Chain Model for Multiple Presentations

In this chapter we present an elaborated version of the Memory Chain Model, with which we can describe the workings of the memory if an item is presented to a subject more than once. With this elaborated model, we will be able to describe spacing effects. In the Section 1 the basic elaboration and assumptions will be described, a model using one memory store will be examined for two and more presentations and we make analytical claims. In Section 2, this will be done for models with more than one store, focusing on two-store models. In Section 3 we examine the workings of the model on some known multi-presentation retention data.

4.1 The single-store model

To be able to examine the spacing effect using the Memory Chain Model, we must make some additional assumptions so that the model captures what happens when the same item is presented to a subject multiple times.

The simplest solution is to handle the new presentation in the same way as the first one, adding points to the first store in the same manner. Assuming that the learning conditions remain the same, this second presentation has the same value for the initial encoding in the first store, $\mu_1$, because the item and subject are the same. The only thing that will change is that we will have even more points in the first store than after only one presentation. The rest of the mechanics of the model remain unchanged, the points still decay according to our decline function $r(t)$.

To show this more extensively, we first have to introduce some additional notations. We use $\tau$ to denote the presentation interval (or presentation lag), which is the time between the first and the second presentation. We also use $t_1$ and $t_2$ to denote the moment of the first and the second presentation (more general: $t_i$ denotes the moment of the $i^{th}$ presentation). To denote the initial encoding of the first and second presentation we use $\mu_1^{(1)}$ and $\mu_1^{(2)}$. In Figure 4.1, the intensity function has been plotted for a case in which a subject receives two presentations of an item.

![Figure 4.1: Intensity function for a single-store model for two presentations. The parameter values are: $\mu_1 = 4.0$, $a_1 = 0.03$. The presentation moments: $t_1 = 5$ and $t_2 = 20.$](image-url)
In Figure 4.1, we see that \( \mu_1^{(1)} = \mu_1^{(2)} = \mu_1 \). In the interval \([t_1, t_2]\) the intensity is equal to \( \mu_1 \tilde{r}(t - t_1) \). After the second presentation, so for all \( t \geq t_2 \), the intensity is equal to the intensity at \( t_2 \) multiplied by the normal decline function. The total level of intensity at \( t_2 \) is the initial encoding of the second presentation added to what is left of the first one:

\[
r(t) = (\mu_1 \tilde{r}(\tau) + \mu_1)\tilde{r}(t - t_2), \text{ for } t \geq t_2.
\] (4.1)

The total intensity after moment \( t_2 \) can also be written as the intensity of the first presentation plus the intensity of the second: we can rewrite the equation to

\[
r(t) = \mu_1 \tilde{r}(t - t_1) + \mu_1 \tilde{r}(t - t_2), \text{ for } t \geq t_2.
\] (4.2)

This is what happens when we do not set an upper bound to the number of representations a store can hold and the store has an infinite capacity.

However, the main assumption to describe memory for multiple presentations is based on the insight that from a neurobiological point of view it is necessary to set just such an upper limit on the intensity a memory store can accumulate within a given time span. (Murre & Chessa, submitted). Therefore it is assumed that the intensity of the point process in a store is bounded by a maximum value, which may be related to a form of saturation during the learning process. We can explain this form of saturation by imagining that a fixed number of “sites” within a store are filled with memory representations during the learning process. These sites can represent synapses, synaptic channels, firing neurons or some other enumerable neurobiological entity or event. The model described in Chapter 3 tells us that after an item has been presented (and learned) once, at any time \( t \), the mean number of sites that are filled in a store is \( r(t) \). If the same item is now learned a second time, a portion of the points that are added by this new presentation fall into already occupied sites. This portion is \( r(t)/r_{\text{max}} \), where \( r_{\text{max}} \) is the saturation parameter and denotes the maximum number of points in a store. Note that the situation of Figure 4.1 described above is a special case where this \( r_{\text{max}} \) tends to infinity. We assume that adding points to sites that have already been filled has no effect (cf. trying to fire an already firing neural group). Now only the remaining points, i.e. the points that will be added to empty sites, contribute to an increase in the total intensity in the store. This remaining fraction is equal to \( 1-r(t)/r_{\text{max}} \). This enables us to write the effective initial encoding of a memory as \( \mu_1(1-r(t)/r_{\text{max}}) \). We can see that for the first presentation (when no sites have been filled yet and \( r(t) = 0 \)) the effective initial intensity is equal to \( \mu_1 \). For subsequent presentations, when the store is already partially filled, it will be less than \( \mu_1 \). Also note that \( r_{\text{max}} \geq \mu_1 \), because number of points the initial encoding added is a minimum for the store capacity. For simplicity, it is assumed that this saturation effect is only present in the first store.

Two presentations
We now take a closer look at the mechanics of a single-store model when an item is presented twice. We will look at the intensity function and the recall probability function and how the length of the (inter)presentation interval influences them. This model has three parameters: the initial encoding \( \mu_1 \), the decline-parameter \( a_1 \) and the saturation parameter \( r_{\text{max}} \).

The only thing we have altered with respect to the situation of Figure 4.1 is the initial encoding of the second presentation. Because of this, when using the same values for \( \mu_1 \), \( a_1 \) and \( r_{\text{max}} \), we get the same values for \( r(t) \) in the interval \([t_1, t_2]\) as in the situation without saturation. Here, \( r(t) = \)
µ_1 \tilde{r}(t-t_1). To describe the intensity after moment t_2, we rewrite Equation 4.1. The initial encoding of the second presentation is now determined by the intensity of the first presentation and r_{max}:

\[ r(t) = \mu_1 \tilde{r}(t-t_1) + \mu_1 (1 - \frac{\mu_1 \tilde{r}(\tau)}{r_{max}})\tilde{r}(t-t_2), \text{ for } t \geq t_2. \]  

(4.3)

Now we can plot the intensity curve for two presentations using a positive, finite value for r_{max}. We use the same values for the parameters \mu_1 and a_1 and for \tau as in Figure 4.1. The result is shown in Figure 4.2.

Figure 4.2: Intensity function for a single-store model for two presentations with saturation. Again, we use \mu_1 = 4.0, a_1 = 0.03; for the saturation parameter r_{max} = 6.0; t_1 = 5 and t_2 = 20.

If we compare this figure to Figure 4.1, we see that the only thing that has changed is the initial encoding of the second presentation.

We can also take a look at the effect of the length of the presentation interval on \( r(t) \). Figure 4.3 shows us the value of the intensity, \( r(t) \), in relation to \( \tau \). This has been done for a few retention intervals, \( t_2-t \), of differing length. Again we use the same parameter values as in Figures 4.1 and 4.2.

One observation is that the longer the retention interval, the lower the intensity (and the recall probability) will be. We also see that for all retention intervals, the intensity drops when longer presentation intervals are used. If this result holds for all possible retention and presentation intervals and for all possible values of the parameters \mu_1, a_1 and r_{max}, then according to this single store saturation model, it would therefore always be better (for higher recall, that is) to present the two presentations in a massed manner regardless of the retention interval. We show that this claim indeed holds. Equation 4.3 can be rewritten as:

\[ r(t) = \mu_1 (1 - \frac{\mu_1}{r_{max}})\tilde{r}(t-t_1) + \mu_1 \tilde{r}(t-t_2), \text{ for } t \geq t_2. \]  

(4.4)

It follows immediately that a maximum for the intensity can be obtained by maximizing \( \tilde{r}(t-t_1) \) and \( \tilde{r}(t-t_2) \) separately. We know that \( \tilde{r}(t) = \exp(-a_1 t) \), so respectively t-t_1 and t-t_2 have to be
minimized to maximize \( r(t) \). This means that not only the smallest retention interval \((t-t_2)\) generates the highest recall probability, but also that \(t-t_1\) will be minimized. Because \(t_1 \leq t_2\), this means that \(t_2-t_1\), the presentation interval, has to be as small as possible to achieve the highest level of recall (with an optimum at \(t_1 = t_2\)). This holds for all values of the three parameters\(^1\).

With this single-store saturation model, we are unable to describe the spacing effects that for instance Glenberg (1976), as described in Chapter 2, found. For that we will need to have a multi-store model, which we will describe in Section 4.2, but first we take a look at the workings of the single store model for more than two presentations.

![Figure 4.3: The effect of the length of the presentation interval, \(r\), on the value of the intensity function for three different retention intervals (10, 50 and 100). The parameter values used are: \(\mu_1 = 4.0\), \(\alpha_2 = 0.03\) and \(r_{\text{max}} = 6.0\).](image)

**Multiple presentations**

We do not alter any of the basic assumptions made above, and still assume that each new presentation adds points to a store, which is bounded by \(r_{\text{max}}\). The effective initial encoding of a new presentation is determined by the total intensity at the moment of this presentation, so for a presentation at \(t_i\) the initial encoding is equal to:

\[
\mu_i^{(1)} = \mu_i \left( 1 - \frac{r(t_i - t_1)}{r_{\text{max}}} \right). \tag{4.5}
\]

The equations for the intensity functions are constructed in the same way as for two presentations. The intensity at a time \(t\) is equal to the sum of the intensity levels from all presentations at time \(t\). To calculate these separate intensity levels we also need the initial encoding levels for each of the presentations. The intensity function at time \(t > t_n\) after \(n\) presentations at times \(t_1 \leq t_2 \leq \ldots \leq t_n\) is equal to:

\[^1\text{If the second presentation completely overwrites the first presentation, when } \mu_1 = r_{\text{max}},\text{ the first part of the equation is equal to zero and the placing of the first presentation has no effect on the recall probability for } t \leq t_2.\]
\[ r(t) = \sum_{i=1}^{n} \mu_i (t-t_i). \quad (4.6) \]

Because of the use of the exponential decline function for \( \tilde{r} \), we can combine and rewrite
Equations 4.5 and 4.6 to a much more graspable equation. For \( n \) presentations, it can be written as\(^1\):

\[ r(t) = \mu \sum_{i=1}^{n} \left( 1 - \frac{\mu_1}{r_{\text{max}}} \right)^n \tilde{r}(t-t_i). \quad (4.7) \]

In Figure 4.4, the effects of multiple presentations of the same item on the intensity level are shown. Again, the same values for \( \mu_1, \alpha_1 \) and \( r_{\text{max}} \) are used.

We now take a closer look at these multi-presentation equations. If \( r_{\text{max}} = \mu_1 \), the new presentations overwrite all older ones and for \( n \) presentations and \( t \geq t_n \) the intensity level is:

\[ r(t) = \mu_t \tilde{r}(t-t_n). \] In this case, only the most recent presentation contributes to the intensity level at any time \( t \). We show this in Figure 4.5.

---

\(^1\) See Appendix A for the derivation.
Figure 4.5: The intensity function for a single-store model before, during and after five presentations \((t_1 = 5, t_2 = 15, t_3 = 25, t_4 = 35, t_5 = 45)\). The parameter values used are: \(\mu_1 = r_{\text{max}} = 4.0\) and \(a_1 = 0.03\).

If we choose \(r_{\text{max}} \to \infty\) (the other extreme case, now there is no saturation at all) we see that the intensity is equal to:

\[
r(t) = \mu \sum_{i=1}^{n} \tilde{r}(t - t_i).
\]  

(4.8)

Another observation is that we can also generalize our claim about the optimal spacing for two presentations to multiple presentations: that for all possible values of \((\mu_1, a_1 \text{ and } r_{\text{max}})\), the maximum intensity level is obtained when the presentation intervals are equal to zero, which is when the presentations are presented in a massed schedule. This follows immediately from Equation 4.7: Because the sum-term within the equation is always positive, the maximum for \(r(t)\) is acquired when \(\tilde{r}(t - t_i)\) is maximal. This is the case if for all presentations \(i, t_i = t\).

The single-store saturation model is therefore unable to describe the spacing effects found in experimental literature. We will therefore in the next section expand this saturation model to a two-store model.

4.2 The two-store model

We will again first describe our two-store model for only two presentations, after which we will show how it works for more than two presentations.

Two presentations

As we have stated in the previous section, we assume that only the first store is bounded by a saturation parameter. Again the total intensity at any time \(t\) is equal to the sum of the intensity levels in both stores \(r(t) = r_1(t) + r_2(t)\). Also, the recall probability is still calculated the same way. For a recall threshold of \(b = 1\) this is: \(p(t) = 1 - \exp(\tilde{r}(t))\). For two presentations the equation for the intensity in the first store is the same as for a single-store model; for \(t \geq t_2\), it can be written as:
\[ r_1(t) = \mu_1 \tilde{r}_1(t-t_1) + \mu_4(1-\frac{\mu_1 \tilde{r}_1(t)}{r_{\text{max}}})\tilde{r}_1(t-t_2), \text{ where } \tilde{r}_1(t) = \exp(-a_1t). \] (4.9)

The point process in Store 2 is still induced by the process in Store 1, so for the intensity in Store 2 we have the convolution of \( \mu_2 r_1 \) and \( \tilde{r}_2 \):

\[ r_2(t) = \int_{t_1}^{t} \mu_2 r_1(z) \tilde{r}_2(t-z) dz, \text{ for } t > t_2 \] (4.10)

We express this equation in our five parameters for the two-store model (\( \mu_1, \mu_2, a_1, a_2 \) and \( r_{\text{max}} \)) and the times at which the presentations occur (\( t_1 \) and \( t_2 \)), which results in the function\(^1\):

\[ r_2(t) = \frac{\mu_1 \mu_2}{a_1-a_2}(e^{-a_2(t-t_1)} - e^{-\alpha_2(t-t_1)}) + \frac{\mu_1 \mu_2}{a_1-a_2}(1-\frac{\mu_1 e^{-a_2 \tau}}{r_{\text{max}}})(e^{-a_2(t-t_1)} - e^{-\alpha_2(t-t_2)}), \text{ for } t > t_2. \] (4.11)

The first part of this equation describes the effect of the first presentation on the intensity in the second store. The second part of the equation denotes the intensity from the second presentation and the only difference is that the initial encoding of the second presentation is bounded by the \( r_{\text{max}} \) parameter. The total intensity in Store 2 is equal to the sum of the intensities of the two presentations in the store. We now again plot a figure, similar to Figure 4.3, in which we show the effect of varying the presentation interval on \( r(t) \) for different retention intervals using the two-store model.

![Figure 4.6](image-url)

Figure 4.6: The effect of the length presentation interval, \( \tau \), on the value of the intensity function for three different retention intervals (10, 50 and 100). The parameter values used are: \( \mu_1 = 4.0, a_1 = 0.03, \mu_2 = 0.02, a_2 = 0.002 \) and \( r_{\text{max}} = 6.0 \).

\(^1\) See Appendix A for details.
Figure 4.6 shows that the highest intensity and thus the highest recall probability is not always reached when the presentation interval is zero. We can identify a different optimal presentation interval for any of the retention intervals. This suggests that, when we employ this two-store model, at least in some cases, two presentations of an item should be presented in a more spaced manner to achieve higher recall probabilities. So this model, contrary to the model with only one store, can describe spacing effects.

We now examine under which circumstances the optimal recall probability is achieved with a more or less spaced schedule (presentation intervals \( \tau > 0 \)) and how this optimal spacing is calculated. Equating the derivative of the recall probability function for two presentations, or equivalently, the intensity function, with respect to \( \tau \) to zero yields:

\[
\tau = \frac{1}{a_1 - a_2} \ln \left( \frac{a_1}{a_2} \left( \frac{\mu_1}{r_{\text{max}}} + \left( 1 - \frac{a_1 - a_2}{\mu_2} \right) e^{-\left( a_1 - a_2 \right) \left( \tau - t_2 \right)} \right) \right). \tag{4.12}
\]

We now, using Equation 4.12, examine for which extreme cases for the retention interval and \( r_{\text{max}} \) the maximum recall probability is reached for \( \tau > 0 \). For multiple presentations we define the retention interval as the period between the last presentation and the moment at which we place the test, which for two presentations has the length \( t - t_2 \).

- If the retention interval is set to zero, this describing an extreme situation in which a subject is tested immediately after the second presentation, from Equation 4.12 follows directly that

\[
\tau > 0 \text{ iff } 1 - \frac{\mu_1}{r_{\text{max}}} < \frac{\mu_2}{a_1}. \tag{4.13}
\]

If \( r_{\text{max}} \to \infty \), then \( \tau > 0 \) iff \( \mu_2 > a_1 \). So even in a two-store model without the saturation, the optimal presentation schedule for two presentations can be a more or less spaced one. This is the case if the transfer rate from the first to the second store exceeds the decline rate in the first store. We then have an expected net increase in points over the two stores combined in time intervals immediately after the first presentation. The total intensity will at a point, either achieve a maximum (if \( a_2 > 0 \)), or increase towards an asymptote if \( a_2 = 0 \). The optimal moment for the second trial to be presented (if we look at a retention interval of 0 and if \( a_2 > 0 \)) is therefore at the moment the intensity curve reaches its peak value. Then, the total intensity of the two presentations combined will be at its highest. This however describes only a limited range of spacing effects and it can never account for the many spacing effects found in experimental literature. Also in almost all of the fits on retention data, the condition \( \mu_2 > a_1 \) did not hold (Cf. Chessa & Murre, submitted). To this we add that for finite \( r_{\text{max}} \), the inequality 4.13 shows that spacing will become more favorable as \( r_{\text{max}} \) decreases.

---

1 Notice that we have to write \( t - t_1 \) in Equation 4.11 as \( t - t_2 + \tau \).
If we choose the retention interval to be extremely large \((t-t_2 \to \infty)\), from Equation 4.12 follows that:

\[
\tau > 0 \text{ iff } \frac{\mu_i}{r_{\text{max}}} > \frac{a_2}{a_1}.
\]  

(4.14)

If we again look at the situation without any saturation \((r_{\text{max}} \to \infty)\), we see that this condition can never be met (as \(a_1\) and \(a_2\) are non-negative). So this model without any saturation cannot describe the positive effect of spacing for long retention intervals.

If we set \(r_{\text{max}} = \mu_1\), the situation where the second presentation fills exactly the same sites in the first store as the first one and is completely overwritten, it is easy to see that, for all retention intervals, the expression for \(\tau\) will result in

\[
\tau = \frac{1}{a_1 - a_2} \ln\left(\frac{a_1}{a_2}\right).
\]  

(4.15)

Now, \(\tau > 0 \text{ iff } a_2 < a_1\). One of our basic assumptions is that the decline rates of preceding stores are greater than the one following it. So under these conditions, an optimal spacing for the two presentations can always be found.

**Multiple presentations**

No additional assumptions are made to expand this model for two presentations to be able to handle an arbitrary number of presentations. Again, the total intensity for an arbitrary number of \(n\) presentations is equal to the total intensity of the first plus the total intensity of second store. The equation for the intensity in the first store is obtained by labeling Equations 4.5 and 4.6. For \(t \geq t_1\):

\[
r_i(t) = \sum_{i=1}^{n} \mu_i^{(i)} \tilde{r}_i(t-t_1), \text{ where}
\]

(4.16)

\[
\mu_i^{(i)} = \mu_i \left(1 - \frac{\sum_{k=1}^{i-1} \mu_i^{(k)} \tilde{r}_i(t_k-t_1)}{r_{\text{max}}}ight).
\]  

(4.17)

The points in the second store still are copied from the first, so Equation 4.10 also holds for multiple presentations. The intensity function summed over two stores is equal to the intensity function of the first store, as given by Equation 4.16, and the intensity function of the second store, which is a generalization of the function given by Equation 4.11 for two presentations.
The full expression for the total intensity function for \( t \geq t_n \) for an arbitrary number of \( n \) presentations becomes:

\[
    r(t) = \mu_1 \sum_{i=1}^{n} \left( 1 - \frac{\mu_1}{r_{\text{max}}} \right)^{a_1} e^{-a_1(t-t_i)} + \frac{\mu_1 \mu_2}{a_1 - a_2} \left( e^{-a_2(t-t_n)} - e^{-a_1(t-t_n)} \right) \\
    + \frac{\mu_1 \mu_2}{a_1 - a_2} \sum_{i=2}^{n} \left( e^{-a_2(t-t_i)} - e^{-a_1(t-t_i)} \right) \left\{ 1 - \frac{\mu_1}{r_{\text{max}}} \sum_{i=1}^{l-i} (1 - \frac{\mu_1}{r_{\text{max}}})^{1-k} e^{-a_1(t_i-t_k)} \right\}. \tag{4.18}
\]

The first sum term describes the intensity in the first store after \( n \) presentations. The second sum term denotes the contribution to the second store of the first presentation and the final sum term gives the contributions of presentations \( i = 2, \ldots, n \) to the intensity function of Store 2, where the term within the curly brackets (following directly from Equation 4.7) contains the total intensity in Store 1 at time \( t_i \).

To study what the effect on the total intensity and on spacing is, we now choose some extreme values for \( a_1 \) and \( r_{\text{max}} \). We will here only present the outcomes of our analytical research on Equation 4.18; the derivations can all be found in Appendix A.

- If \( a_2 = 0 \) and we again take the model without any saturation \( (r_{\text{max}} \to \infty) \), we can identify three situations.
  - First, if \( \mu_2 \leq a_1 \), the maximum intensity (and thus recall probability) at any moment \( t \) with \( t \geq t_n \) is achieved if all presentations are presented in a massed way, with a retention interval as short as possible: \( t_1 = \ldots = t_n = t \). Intuitively we can see that in this case, from the time of the last presentation, there is a net decay of points over the two stores and because there is no saturation, the presentations should all be presented as close to \( t \) as possible to acquire a maximum intensity at \( t \). And because there is no decay in the second store, once the first store is completely empty, no points will be added or decayed in the two stores, so the intensity curve tends to an asymptote.
  - Second, if \( \mu_2 > a_1 \), the maximum intensity at \( t \) is again reached if the presentations are presented in a massed way, but now the retention interval should be as long as possible \( (t_1 = \ldots = t_n \text{ and } t-t_n \to \infty) \). In this case, there is a net increase in the expected number of points in the two stores and we want to profit as much as possible from the increasing intensity function.
  - The third case is when \( \mu_2 = a_1 \). In this special case, the intensity function is equal to \( \mu_1 n \), so that any choice of the presentation times \( t_i \) can be made. We have thus seen that for models without saturation and \( a_2 = 0 \), massed schedules \( (t_1 = \ldots = t_n) \) are optimal.

- If \( a_2 = 0 \) and we take the other extreme value for the saturation parameter, \( r_{\text{max}} = \mu_1 \), we find that an extreme form of spacing would generate the maximum intensity. It is reached if all presentation intervals are of equal length (equidistant) and that they are as long as possible. Because each new presentation overwrites the preceding one in the first store and because there is no decay in the second store, the best way to schedule the presentations is so that each one has the maximum possible time to copy its points from the first to the second store before a new presentation overwrites it.

- If \( a_2 > 0 \) and \( r_{\text{max}} \to \infty \), the optimum presentation schedule will be a massed one. If \( \mu_2 < a_1 \), the same as without any decay in the second store, the retention interval has to be as small as possible, so \( t_1 = \ldots = t_n = t \). If \( \mu_2 \geq a_1 \), we can derive the exact moment at which
all presentations should be scheduled for the intensity to be at a maximum at moment \( t \).

This is for all presentations \( i \):

\[
t_i = t - \frac{1}{a_1 - a_2} \ln \left( \frac{a_1}{a_2} (1 - \frac{a_1 - a_2}{\mu_2}) \right)
\]

(4.19)

We can see that the smaller the value of \( a_2 \) is, the further away the presentations should be presented. Also note that Equation 19 follows from Equation 12 for two presentations.

- The final situation we describe is when \( a_2 > 0 \) and \( r_{\text{max}} = \mu_1 \). Now, each new presentation completely overwrites its predecessor and there is no decay in Store 2. If we calculate what presentation lags generate the highest recall probability, we find schedules where the presentations have smaller presentation lags as they get closer to the moment of the test. So, if \( l_i \) denotes the \( i^{\text{th}} \) presentation lag, that is \( t_{i+1} - t_i \), then \( l_1 > l_2 > \ldots > l_{n-1} \). We call this a ‘shrinking rehearsal’ schedule, in contrast to a schedule of ‘expanding rehearsal’, where the lags get longer for later presentations. Such expanding rehearsal schedules are not found in the extreme cases that we have analytically examined.

### 4.3 Fits on multiple presentation data sets

As with the original model, the Memory Chain Model for multiple presentations was also fitted to known retention data. Here, we describe results of some of these fits. The experiments of which the data was fitted mentioned below are described in Chapter 2.

A two-store model with saturation and \( b = 1 \) was fitted to the data from Glenberg’s (1976) Experiment 1. At first, the model failed to fit all data points. When analyzed, it was found that the model was rejected because of one data point; the data had a clear ‘dip’ for the presentation lag of one intervening item. If this single data point was ignored, the model produced a much better fit and was not rejected. The nature of this dip is unknown, but a replication of Glenberg’s Experiment 1 did not generate this dip. In Table 1, the values for the five parameters are reproduced (Murre & Chessa, submitted).

The model was also fitted to Rumelhart’s data from his Experiments 1 and 2 (Rumelhart, 1967). Over the course of the two experiments Rumelhart conducted to explore the effects of different distributions of learning trials, a total of 19 different distribution schedules were considered. The Memory Chain Model was fitted to the data from the two experiments, again using a two-store model with saturated learning and \( b = 1 \). The model was not rejected on both the experiments. The fact that the model fitted all 19 widely varying learning-and-forgetting curves of the experiments provides evidence that the model captures a lot of the main characteristics of learning from multiple presentations (Murre & Chessa, submitted). Again the values that provided the best fit are shown in Table 4.1.

<table>
<thead>
<tr>
<th></th>
<th>( \mu_1 )</th>
<th>( a_1 )</th>
<th>( \mu_2 )</th>
<th>( a_2 )</th>
<th>( r_{\text{max}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Glenberg (1976)</td>
<td>0.635</td>
<td>0.251</td>
<td>0.158</td>
<td>0.0089</td>
<td>0.860</td>
</tr>
<tr>
<td>Rumelhart (1967)</td>
<td>1.68</td>
<td>0.25</td>
<td>0.093</td>
<td>0.0081</td>
<td>3.70</td>
</tr>
<tr>
<td>Rumelhart (1967)</td>
<td>1.23</td>
<td>0.16</td>
<td>0.041</td>
<td>0.00</td>
<td>3.78</td>
</tr>
</tbody>
</table>

Table 4.1: Parameter values for fits on data from Glenberg (1976) and Rumelhart (1967) (adapted from Chessa & Murre, submitted).
Table 1 shows us that for both fits the best fitting parameter values provided evidence for a saturated form of learning. In all cases, the value found for $r_{\text{max}}$ was in the same order as the $\mu$ parameter. Therefore, subsequent presentations would certainly be ‘saturated’.

The model was also fitted to the data from Zielske’s 1950s experiments (the data was adapted from Simon, 1979). The two- and three-store models without saturation and $b = 1$ were rejected. The two-store model with saturation and $b = 1$ provided a good fit, although one erratic data point was left out of the fit, and the model was not rejected. A three-store model provided a slight improvement to the fit.
Chapter 5

Optimizing second language acquisition

With the Memory Chain Model for multiple presentations we have a mathematical model that describes how multiple presentations of an item are processed in memory. This model can be used for the optimal learning of second language vocabulary. In this chapter, we first describe a framework for a tool that can do just that. We will then in Section 2 describe our implementation of a part of this tool.

5.1 A framework for optimizing second language acquisition

In this section we will describe a framework for an online tool that can be used for the optimal acquisition of a second language vocabulary. When we refer to a user or a subject of the tool with ‘he’ we ask the reader to read ‘he or she’.

Existing tools: SuperMemo and Captain Mnemo

The goal of a tool for optimal language acquisition is to make sure that a user acquires a maximum amount of knowledge of a certain corpus of items within a given time span. A few tools that try to do this have been proposed and developed earlier. One of these learning tools is SuperMemo, based on a model of memory and forgetting, by the Polish researcher Wozniak (Wozniak, 1995)\(^1\). This tool is a computer program that teaches a set of items (for instance second language vocabulary) by presenting the items according to the underlying model. The model makes standard assumptions about the learning and forgetting speed of a user and the program uses optimum presentation intervals and item difficulties to determine the sequence in which the items are presented to the user. Along the way, the program calculates the user-specific retention parameter values by registering the user’s false and correct answers. SuperMemo is produced as a commercial speed-learning software package.

More or less at the same time as the development of SuperMemo, Murre created a similar tool called Captain Mnemo. The underlying model for this tool incorporated existing theories of learning and forgetting explicitly. It assumes that recall during learning and forgetting can be described by a power function (see, among others, Anderson & Schooler, (1991) or Wixted & Ebbesen, 1991). The program uses these power functions to present an item to a user at the moment at which the learning effect of that item is expected to be maximal. More on Captain Mnemo can be found in (Kriele, 1998).

Expanding Captain Mnemo

Captain Mnemo was designed as a first attempt for a tool that provides optimal learning, building on existing theories of learning and forgetting. These theories were not able to describe the spacing effect accurately, as we have argued in previous chapters and are therefore not adequate to calculate optimal presentation moments. The tool we introduce below incorporates the Memory Chain Model for multiple presentations as illustrated in Chapter 4. This model, as we have argued in the previous chapter, is able to describe the spacing effect and with our tool we will therefore be able to calculate optimal presentation moments.

\(^1\) For more information, see also http://www.supermemo.com
Our tool is also designed to be accessed and used online via the Internet, unlike Captain Mnemo and SuperMemo. This has a number of advantages: The primary advantage is that the same tool can then be used by more than one user at the same time and information gathered by the program from multiple users can be used to update the learning model as opposed to the updating of the model from information about one subject only, which is the case with Captain Mnemo and SuperMemo. As a consequence, the program can work with more accurate information to teach the users. A second advantage is that all kinds of information about the performance, workings and usability of the program can be made directly available to the designers and maintainers of the program and the infrastructure of the program can be modified accordingly. Another advantage is that, if the first experiments using the tool by a selective group of people were successful, the experiment could be made public on the Internet. This way we can attract a large number of subjects to get more and more accurate data. More information on advantages and disadvantages of the use of the Internet for data gathering can be found in Janssen, Meeter and Murre (2001).

**Description of an optimization framework: user interface**

Anyone wanting to learn a certain corpus of items (we will assume the items to be second language words and their translations), will log on to the website where the tool has been placed. A diagram of the framework for the tool is shown in Figure 5.1

![Diagram of the framework for a CALL tool using the Memory Chain Model for multiple presentations](image)

If the user is not yet known by the program, he is redirected to a registration page. Here it is required to enter personal information that can be relevant to the learning task. This information includes among other things age, level of education, first language and levels of education in other languages. This information can be used to study correlations between these personal factors and learning and forgetting speeds. A screen dump of an actual implementation of this registration page is shown in Figure 5.2.
After this form has been filled out, the user can choose a login ID and a password to identify himself. He is then redirected to the front-page of the website. On this page he can log in using his personal ID and password so all personal information and previously collected information about past learning sessions are available to the program.

Next comes a page where a user can enter his choices for the current learning session. Here he can enter how much time he wants the learning session to take, how many items he wants to learn and his goals for this learning session. These learning goals can take on a number of forms. For instance a learning goal can be to have a maximum percentage of ‘known’ items at the end of a certain total learning time. Another learning goal can be to know the items up to a predefined percentage within the shortest amount of learning time possible. A goal can also be to have a maximum increase in knowledge of the entire corpus of items within a given total learning time. These goals can all be obtained by calculating presentation schedules with optimal presentation intervals.

After the user has entered its wishes, the learning session can begin. He now interacts with the part of the tool that actually teaches the items. Based on all personal information, difficulty of the items, previous learning sessions, and parameters of the current learning session, the program calculates the sequence in which the words are to be presented.
The first word of this sequence now appears on the screen and the user is asked to type in the correct translation of that item. Figure 5.3 shows a screen dump of an actual implementation of this screen.

![Figure 5.3: The response-screen. The user is asked to give the correct translation of the word shown in the box in the center of the screen ('merhaba'). In the top of the frame, the time bar can be seen.](image)

The answer that is entered is either correct or false. This information is used by the program to update the parameter values of the model and the sequence of words to be presented can be revised accordingly using this new information. This can be done after each item, or after a number of items. After each response, whether correct or false, the user is again shown the item and the correct translation. The user clicks a button to indicate he is done with this item. The program then presents the next item in the (updated) sequence. This process of presenting items continues until a certain time has elapsed or a certain requirement has been fulfilled, depending on what goals the user has entered for the learning session.

**Obtaining parameter values**

In the description above, the interesting step is the calculation of the optimal sequence of words by the model. This sequence is calculated by first calculating the optimal schedules for all individual items in the corpus and then putting all of these optimal item-sequences together. For better understanding of how the sequence is calculated we can look at how the optimal schedule for one item is calculated. To further simplify this, we look at how the optimal interval between two presentations of an item will be calculated.

If someone uses the program for the first time, the program has no knowledge at all about the learning and forgetting rates and the saturation factor of the user, which the model needs to calculate optimal presentation intervals. It therefore makes assumptions about these parameters based on data acquired by previous users. The two-store Memory Chain Model for multiple presentations is fitted to this data to obtain values for the five parameters of this model ($\mu_1$, $a_1$, $\mu_2$, $a_2$, and $b$).
a_2 \text{ and } r_{\text{max}}). \text{ These values are therefore not specific for an individual learner, but rather describe the learning and forgetting of the average learner of this specific corpus of words. The model, with the obtained values, is then used to calculate an optimal presentation interval for the item, which will be described in Section 4.2.

If a sufficient amount of data is available about past learners, the corpus of words is split into more difficult and easier words, and the model is fitted separately to the data from the words in these two groups. The values for the parameters now represent the learning and forgetting rates for the average user for difficult and easy words. The program then uses the different instances of the model to calculate different optimal presentation intervals for items in the two categories. If even more data points are available, the corpus of items can be divided into even more subsets. If the amount of data is limited, but we do want different model versions for difficult and easy items, a choice can also be to vary only one parameter value for the different subsets of items and to keep the remaining four parameter values for all items. The parameters that are the most likely candidates to be varied first are \( a_1 \) and \( \mu_1 \), because they have proved to be the parameters that are the most task-specific (Murre & Chessa, submitted).

After the user has learned using the program for some time, more and more individual information about the learning and forgetting rate is available. This information can be used to calculate the parameter values of the user-specific model, with which the user-optimal presentation intervals and the appropriate item presentation sequence are calculated. Again, if some, but not a lot of this individual information is available, the program can vary only one parameter value for the individual user and calculate the other values using the collective data. Now the most likely candidate to vary is \( \mu_1 \), as it has shown to be the most individual-specific (Murre & Chessa, submitted). Using registration data such as level of education or age, categories of users can be formed that have mutual parameter values specific for that category. The more learning and forgetting data the program acquires, the more personalized the parameter vector that the model uses will be employed. An additional suggestion is that a Bayesian statistical framework could determine the parameter values for the user model using the values obtained by fitting the model to both general and user-specific data. We will however, not describe this here in detail.

**Description: Calculating optimal presentation intervals**

With the five parameter values of a more or less user-specific model, the optimal presentation schedules can be calculated for an item. We have to take into account what the goal of the specific learning task is and what restraints (time, maximum number of presentations of a single item) are in effect. The learning goal tells the program what the actual value is that should be optimized.

We illustrate this with an example. A simple learning goal is to have a maximum recall probability of an item after a retention interval of a set number of intervening items. A possible restraint is to have only two learning presentations, or learning trials. The program then has to calculate what length of the presentation interval results in the highest recall probability. For this purpose the program takes all relevant data as described above and fits the Memory Chain Model to that data. The resulting parameter values will be used to calculate the optimal interval between the two presentations. In this specific example, the calculation can be done analytically. For this take the recall probability equation for a two-store model for two presentations and calculate the derivative of that equation (or of the intensity function, because it is a monotonous transformation) with respect to the presentation interval, \( \tau \). This results in Equation 4.12 and the optimal presentation interval for this learning goal is known.

We further illustrate this example by giving values for the five parameters and the retention interval. This is done for two separate cases in Figure 5.4.
to the. In = 5.0), the optimal presentation lag is longer (more spaced) than in case B (\( \mu_1 = 2.0, a_1 = 0.03, \mu_2 = 0.03, a_2 = 0.02 \) and \( r_{max} = 2.5 \)).

Although the above example provides us with valuable insight into the workings of the model, it is a very specific one that will not often occur in real learning situations. We therefore present another example, which is a generalized version of the previous example. Here we do not limit the number of presentations to two, but now we optimize the recall probability after an arbitrary number of presentations, again for a set retention interval (now defined as the interval between the last presentation and the recall test). Because of the added complexity of this optimization that arises from the multiple presentations, an analytical solution to the optimization problem that can be used in all cases can no longer be constructed. We therefore have to develop a numerical procedure to calculate the optimal presentation schedule. The value that needs to be optimized is of course still the recall probability.

A first concern arises from working with an arbitrary number of presentations. This means that for every additional presentation, another parameter should be added to the calculation. To avoid this, we use a parameterization of the presentation schedule. Presentations are distributed according to some parameterization. There are a number of possibilities for this parameterization. One could be to distribute the presentations equidistantly. This means that all presentation intervals are of equal length. This has the advantage of resulting in only two extra variables, the number of presentations and the presentation interval length. But with this parameterization, schedules using expanding and shrinking rehearsal cannot be realized. To be able to do this we have to come up with another parameterization. We will describe a parameterization used in our implementation, the exponential parameterization. The following equation describes how the \( n \) presentations are scheduled. For the \( k^{th} \) presentation the presentation time \( t_k \) is:

\[
t_k = t_n - T \left( 1 - \frac{k-1}{n-1} \right)^{1/3}, \text{ for } k = 1, \ldots, n,
\]

(5.1)
where \( t_n \) denotes the time at which the last presentation is presented. This parameterization uses two parameters, \( T \) and \( s \). The parameter \( T \) \((T \geq 0)\) sets the length of the entire period in which the presentation are scheduled. It sets the moment of the first presentation, \( t_1 = t_n - T \). The parameter \( s \) \((s > 0)\) defines the extent to which a schedule shrinks or expands. If \( s = 1 \), the schedule is equidistant. If \( s > 1 \), the schedule is an expanding schedule (the presentation intervals become shorter the closer they get to \( t_n \)). If \( s < 1 \), it describes a shrinking schedule. A few examples of these schedules are shown in Figure 5.5.

![Figure 5.5](image_url)

Figure 5.5: Four timelines showing four different distributions of presentations for different values of \( T \) and \( s \). The number of presentations, \( n = 5 \) and \( t = 0 \) is fixed at the moment of the last presentation.

We now can define the optimization procedure. The schedule defines all presentation moments and the intensity function for multiple presentations, which is found in Equation 4.18, has to be maximized for the values of the five model parameters, \( \mu_1, a_1, \mu_2, a_2 \) and \( r_{\text{max}} \), the number of presentations \( n \) and the retention lag, \( t_n - t \), which determines \( t_n \). This can be done by a simple numerical maximization method that iteratively searches for the values of \( T \) and \( s \) that produce the highest retention and thus the highest recall probability.

These two examples are not the only possible ones, as has been argued above. For the other possible learning goals, different optimization procedures could be devised, using the same principles as the ones shown above.

### 5.2 Implementing part of the tool

The actual implementation of a program we described the framework for in the previous section is a very large project. Because of time limitations we implemented only a part of the framework as described above. It is intended as a first step in designing a full-scale online Computer Assisted Language Learning tool.

The project name for the tool we built is ‘OptiLearn’. The tool was built to serve a dual purpose. The first is that, as we have described above, in order for the optimization tool to work there has to be a certain amount of learning data in the database. A starting amount of data is needed to
calculate initial learning parameter values for a certain learning task and derive optimal learning schedules for new users on that task. This data was to be acquired by letting users learn items by interacting with the actual implementation of the optimization tool. A second purpose for constructing a part of this tool was to validate and test the framework and the Memory Chain Model for multiple presentations; that is to test empirically whether the model is indeed able to predict optimal learning schedules.

The overall goal of the constructed tool was to calculate optimal presentation schedules for the learning of second language words and their Dutch translations. We therefore constructed an environment on the Internet in which users can learn the word-pair items and where their learning and forgetting data can be collected. With this data we estimate the five parameter values of the Memory Chain Model for multiple presentations and these values can then be used to calculate the optimal presentation schedules. So our tool uses the first learning sessions only to collect data to calculate the parameter values from after all users have finished learning and no adaptation of the presentation is done. All calculations will be done offline. However, we designed our software in such a way that they could easily be used for an online optimization tool as we described in the previous section.

Our tool is accessible via the Internet. It was constructed to be a part of the Neuromod website\(^1\). On this website, which has been constructed by members of the Neuromod group a number of online memory tools can be found. An advantage of integrating our tool within the Neuromod website is that we could make use of existing knowledge and software that have already been used for the development of other online memory tasks located on their website.

Our implementation is constructed using the open source application server Zope\(^2\). Zope is used to develop all web applications in the Neuromod group. Using Zope, we created an environment that would facilitate users in the learning of second language vocabulary. This user interface was used for our tool, but designed to be able to handle the online optimal learning as described in the previous section as well. For the OptiLearn tool, we also constructed a database. In this database, we can store information about users of the tool, their performance on learning and their optimized learning schedules. As with the design of the user interface, it was constructed to be a database for an actual online learning tool as we have described in the previous section. The database was written in MySQL and it was located on the local server of the Neuromod group.

Zope allows the use of scripts written in the Python scripting language\(^3\), which we used to perform the numerous calculations. The use of Python scripts makes it possible to run the scripts in an online environment, which again makes it relatively easy to expand our tool to a full scale online computer assisted language learning tool. The tool can be found at the homepage of the OptiLearn project: http://www.neuromod.org/mnemo/optilearn.

**Initial learning: the presentation schedule**

The learning data gathered by the program is used to estimate the values of the five parameters of the two-store model: \(\mu_1, a_1, \mu_2, a_2\) and \(r_{\text{max}}\). As described in the previous chapters, the first four parameters describe the learning, induction and forgetting rates of the subjects. User- and task-specific models have the same value for a single presentation as for multiple presentations. So the values for \(\mu_1, a_1, \mu_2\) and \(a_2\) can be extracted from single-presentation data. The \(r_{\text{max}}\) parameter however, defines the limitations on the initial encoding of the subsequent presentations. It is the only parameter that has to be acquired using multi-presentation data. Therefore the program

\(^1\)For more information: http://www.neuromod.org

\(^2\) For more information: http://www.zope.org

\(^3\) For more information: http://www.python.org
collects two types of data. For the values of $\mu_1$, $a_1$, $\mu_2$ and $a_2$ single-presentation recall data and for the value for $r_{\text{max}}$ recall data from more than one presentation.

The single presentation recall data consists of the levels of recall for different retention intervals. These recall levels are measured in percentages of correct responses. As in Glenberg’s experiment (Glenberg, 1976), as described in Chapter 2, the retention intervals in our tool are defined in numbers of intervening items, rather than in seconds or minutes. Note that the measure of intervening items can be rewritten to the measure of time, by multiplying the number of intervening items with the average presentation duration of an intervening item. The users receive one learning trial (consisting of the stimulus and the correct response) and after a variable retention interval they have to reproduce the correct response to the presented stimulus. We use 18 possible retention intervals, ranging from one to 49 intervening items. The response of the users can either be correct (a score of ‘1’) or false (a score of ‘0’). The scores from all users are then used to produce an average recall score for the 18 retention intervals. The 18 different retention intervals are chosen to provide us with as much characteristics of the recall probability curve as possible. Earlier simulation tests showed us that differentiation in the recall scores for shorter retention intervals had a much stronger effect on the parameter values than differentiation in the recall scores for longer retention intervals. Shorter retention intervals therefore provide us with much more information than longer intervals, so the tool uses mostly shorter retention intervals. We also found that after 50 intervening items, the level of recall seems to decline very slowly in the same way as in the retention interval range from 40 to 50 intervening items, thus giving very little extra information. This prompted us to make 50 intervening items the length of the longest retention interval.

Approximately four fifths of the total number of trials the user receives are used to collect this kind of single-presentation data. The remaining portion of the trials is used to obtain data for the estimation of the value of the $r_{\text{max}}$ parameter. We chose this distribution because the single-presentation data is used to fit four parameters whereas the multi-presentation data is used to fit only one parameter. Because the Memory Chain Model has just one parameter, $r_{\text{max}}$, for saturation and this parameter has a constant value for a specific group of subjects and a specific testing situation, the value for $r_{\text{max}}$ determining the saturation for two presentations is the same as for more presentations. It therefore suffices to use data from two presentations. For this data, the program presents the items two times to the user using a relatively short or a relatively long presentation interval, respectively four or fourteen intervening items long. From the earlier tests, we concluded that these values are adequate for our procedure. After a short or long retention interval, which may again be four and fourteen intervening items long, the item is presented a third time, this time functioning as a test trial.

**The items**

So far, we have used the word ‘item’ to describe what is presented to a subject and what the memory representations formed are about. As we have mentioned previously, an item can consist of a picture, a number, an advertisement, etcetera and our tool was designed to be able to work with any corpus of items. For our tool, we choose second language vocabulary learning as our domain. One of the reasons for this is that this could have some obvious real-life applications in second language education. As the second language to be learned, the native language being Dutch, we choose Turkish. This is done for a number reasons: first, we need a language of which not many people had any prior knowledge (this ruled out languages like French, English or Spanish). This interference of prior knowledge should be avoided because first presentations of items are no longer the actual first encounter with the item (an item can be already known). The lack of prior knowledge also helps to justify our assumption that a specific item has the same
difficulty for all users. The second reason is that we need a language that people were interested
to learn, so that enough people would be willing to join in on our experiment to test the tool (as
described in Section 3). This suggests a language like Turkish, for which we think is a basic
interest, instead of using a language like Swahili or nonsense syllables like for instance Glenberg
(1976). A third reason is that at the same time our tool was developed, another experiment
involving second language learning was being conducted within the Neuromod group, using
Turkish-Dutch word pairs. This experiment could be used to cross-validate our findings
(Kruithof, 2003).

The stimulus part of the items therefore consists of Turkish words, while the response part
consists of their Dutch translations. We assumed all items to be of equal difficulty. Apart from the
fact that all Turkish words are assumed to be unknown to all users, this assumption is justified by
the fact that we chose all Turkish words to have an approximately equal number of syllables
(two) and to have all Dutch translations an equally a high frequency of occurrence in Dutch
speaking language, according to de Groot (1988). Also, only words without any special Turkish
characters were used, to facilitate pronunciation of the words and processing of the results.

OptiLearn architecture
The overall architecture of the OptiLearn tool is shown in Figure 5.6.

![OptiLearn architecture diagram](image)

Figure 5.6: A schematic diagram of the OptiLearn tool. The white blocked arrows indicate the
path users follow, while the striped arrows indicate information flows between the different Zope
pages (boxes) and the underlying program modules.

In this tool, once a new user has registered, he can log in to the program and start the learning
session. Once he has done so, the program uses an algorithm implemented in Python to calculate
a new schedule for the user. This schedule is unique to the user and a certain degree of
randomness in the distribution of the different retention and presentation lags is employed. After
the schedule has been created, the user is presented with the first word of the schedule (shown in
Figure 5.3). Because we want the presentation duration of all items to be approximately of equal
length, a time bar is shown in the presentation screen indicating the time the user has left for
entering the correct response. But since we do not want to penalize slow typing users, the time
bar was a mere indication, and the user can enter a late response without any penalties. All data
was checked before processing to remove any data from users that abused this freedom by using
absurdly long response times. After the user has entered his response (the translation of the
second language word), the program checks the database to see if the response is a correct or
incorrect translation. This is then written to the response database, where a tuple \( \text{retention} \)
interval, correctness] for single presentation trials or a tuple [presentation interval, retention interval, correctness] for the two-presentation data is stored, together with information about the current user and session.

Now the correct response is shown to the user, along with the original stimulus part of the item so the user can learn the correct answer or gain confirmation of his correct response. After the user decides that he has seen the stimulus and the correct response for long enough a period, he can click a button on the screen to indicate he wants to be presented with the next item of the schedule. However, if the user provided an incorrect response to the stimulus, the original stimulus is presented again and he is again asked to reply with the correct response. This way, we make sure the user has paid attention to the stimulus before proceeding to the next item and that he has processed the word. After a correct response, the program presents the next item in the presentation schedule. This continues until there are no items left in the schedule, at which point the session is ended.

**Data processing: parameter estimation**

With the data from all subjects gathered in this way we can estimate the values of the five parameters of the two-store Memory Chain Model for multiple presentations. As mentioned above, the data gathered from single trial data is used to estimate the values of the four parameters $\mu_1$, $a_1$, $\mu_2$ and $a_2$. The two-presentation data is then used, together with the values found for those four parameters, to calculate the value of $r_{max}$.

Because of time limitations we designed the method for the parameter estimation only for a recall threshold of $b = 1$.

**Estimating parameters from single presentation data**

The values of the four parameters $\mu_1$, $a_1$, $\mu_2$ and $a_2$ could not be computed from the data in an analytical way because of the complexity of the equations to be solved. We therefore constructed a numerical method for the estimation of the values for these parameters. However, a rough analytical estimator we created could provide us with an initial estimate of the $\mu_1$ and $a_1$ values.

This rough analytical method estimating $\mu_1$ and $a_1$ assumes that the recall data is a realization of a one-store Memory Chain Model. From the properties of the one-store Poisson point process, with the assumption that $b = 1$, we can derive estimators for $\mu_1$ and $a_1$ as follows:

Let $T$ denote the set of all retention intervals examined. We introduce subsets $T_1$ and $T_2$ of $T$, which respectively contain the shortest retention intervals in $T$ and the remaining retention intervals in $T$. Let $Z_1$ and $Z_2$ denote the average length of the retention intervals in respectively $T_1$ and $T_2$. Also, let $V_1$ and $V_2$ denote the average number of false answers given by all subjects for the retention intervals in respectively $T_1$ and $T_2$. Now, the rough estimator for $a_1$ is given by:

$$a_1 = \frac{1}{Z_1 - Z_2} \ln \left( \frac{\ln(V_1)}{\ln(V_2)} \right)$$  \hspace{1cm} (5.2)

and a rough estimator for $\mu_1$ is given by:

$$\mu_1 = -\ln(V_1) - e^{a_1 Z_1}.$$  \hspace{1cm} (5.3)

Our analysis of these estimators based on a simulation study provides evidence for the claim that they are accurate for data that can be explained by a single-store model. However, since we

---

1 See Appendix A for details.
assume that the actual data is a realization of a two-store process, we will merely use the values estimated as starting values for our numerical fitting procedure.

**The numerical fitting procedure**

The data from which we are to extract the values of the four parameters consist of tuples \([\text{retention interval}, \text{correctness}]\), where correctness is either 1 or 0. We do not use our minimization methods directly on this binary data, the actual data to which we will fit the parameters is obtained by taking the average recall scores on the different retention intervals over all users, denoted by \(P(t)\). This results in tuples \([\text{retention interval}, P(t)]\). These recall scores are, as we assume, realizations of a stochastic recall process that is induced by the two-store Memory Chain Model. Again, we assume the recall threshold, \(b\), to be 1. So we need to find the values of the four parameters of the recall probability. To this extent we must establish the minimum of some sort of error measure between the data and the probability function with the values of the four parameters. We decided to use the relatively common least squares error measure rather than another optimization criterion (like maximum absolute difference minimization or maximum likelihood) because it proved to be successful for fitting retention data (see for instance Chessa & Murre, submitted). So, if \(T\) denotes the set of retention intervals examined and \(P(t)\) denotes the average recall score for a retention interval \(t \in T\), the error function is given by:

\[
E = \sum_{t \in T} \left( P(t) - P_{\mu_1, \mu_2}(t) \right)^2. \quad (5.4)
\]

For the numerical minimization of this error, we considered three possible algorithms. These three candidates were:

- **Algorithm 1**: A simple hill-climbing algorithm, developed by Murre (not published). On each iteration, this algorithm tries out all possible changes in the set of parameter values by adding or subtracting to each variable a non-fixed value \(\alpha\), ‘the step size’, or leaving the parameter value unchanged. It then calculates the error value for each of these permutations of the parameter value set. For \(n\) parameters, on each iteration, \(3^n\) different combinations of parameter values must be evaluated. The value of \(\alpha\) is adjusted after each step, increasing if the step resulted in a lower error and decreasing otherwise, so that it functions as a momentum to the searching algorithm. This algorithm searches for the minimal error in a completely deterministic way and continues to do so until a user-defined maximum number of iterations has been reached or the momentum drops below a user-defined level, at which point the algorithm stops and returns the current parameter values. It will however not respond very well to local minima.

- **Algorithm 2**: A version of Caprile and Girosi’s (1990) NonDeterministic Optimization Algorithm, based on adaptive noise, modified by Murre. In each step noise is added to a single parameter according to a uniform distribution. If the change results in a lower error value, the noise parameters are adjusted and the search algorithm will move faster in the chosen direction of the ‘error landscape’. This algorithm continues until the rate of change becomes less than a user-defined value or until a prespecified number of iterations has been reached (‘time-out’). Although this algorithm makes use of stochastic factors and is therefore nondeterministic, it is not unaffected by local minima.

- **Algorithm 3**: A version of Broyden, Fletcher, Goldfarb, Shanno’s (BFGS) Quasi Newton minimization algorithm. This algorithm is a version of a quasi-newton method, which is a type of gradient descent minimization method. Gradient descent algorithms are iterative
algorithms that use the gradient, $\nabla E$, of the error function to minimize the error. In each iteration, the algorithm calculates the direction on the four-dimensional error-landscape in which to go, adjusting the independent parameters according to a step size. This step size is also constantly adjusted to reduce the chance of ending up in a local minimum. The algorithm fits the recall data using the four independent parameters at the same time, in a parallel manner. The implementation in Python was done by T. E. Oliphant\footnote{For more information, see http://pylab.sourceforge.net/}.

These three algorithms were tested by simulating ‘recall data’ according to stochastic recall processes and applying the three algorithms to fit this data. This recall data was generated using the two-store recall probability function for single presentations as described in Chapter 3, with values for the parameters. The recall probability was calculated for over 100 different retention intervals ranging from one to 170 intervening items. All three algorithms were run 216 times, each run using a different set of starting values for the parameters. All three algorithms had first undergone some adjustments to their control parameters to make sure their performance would be optimal for this specific data set.

The performance of Algorithm 1 was characterized by its slowness. The algorithm needed more than a minute to reach a somewhat acceptable solution, which is very long if we want to be able to use this algorithm in an online implementation. Moreover, the algorithm did seem to be affected by the quirky nature of the error landscape and it frequently ended up in a local minimum. Only 66 of the 216 starting sets of parameter values resulted in a ‘correct’ minimization and the correct results all originated from the starting values that were closest to the true values.

Algorithm 2, although being slightly faster, performed much worse, even with additional adjustments to the values of the control parameters. The algorithm almost always ended up in a local minimum, and only one set of starting values resulted in a correct result.

The performance of Algorithm 3 was much better and an important feature is that it produced the outcome very fast. An average of 1.3 seconds was needed to complete the computations. Its performance in producing correct answers was almost as good as that of Algorithm 1: 41 of the 216 sets of starting values resulted in correct estimations of the parameter values, it therefore proved to be not entirely unaffected by the difficult error landscape and it frequently ends up in local minima.

The test showed that Algorithm 1 came up with the right answer somewhat more often than Algorithm 3, but since Algorithm 3 can be run approximately 40 times in the amount of time Algorithm 1 needs to complete a single run, we chose to use Algorithm 3, using multiple sets of starting values. As noted above, the rough estimates of the values of $\mu_1$ and $a_1$ can be used to base the set of starting values on.

To determine how many different data sets we would actually need and to take a closer look at our fitting procedure, we ran further tests on our BFGS-powered fitting procedure. These tests consisted of simulating the learning of a variable number of ‘users’ and then testing our fitting procedure on their data. The results of these tests suggested that the BFGS algorithm needed more than one set of starting values to begin the optimization for coping with local minima. For the starting values for $\mu_2$ and $a_2$, we did not have any prior information, we chose 0. The parameters $\mu_1$ and $a_1$ both have three possible values: the value estimated with the rough estimator and that value plus and minus ten percent. This results in a total of nine starting sets of parameter values to be used. Our tests proved this to be sufficient: usually only two or three of the
nine outcomes was a local minimum. The same tests showed us that there exists a tradeoff between the number of subjects and the number of items learned by the subjects for the accuracy of our fitting procedure. Approximately 25 data points per retention interval seemed to be the minimum for producing a satisfactory fit.

**Estimating the saturation parameter**

After these four parameters have been estimated, \( r_{\text{max}} \) remains to be estimated. As we have argued, the value of \( r_{\text{max}} \) is the same for two presentations as for more than two presentations. Besides the values for \( \mu_1, a_1, \mu_2 \) and \( a_2 \), to be able to calculate the value of \( r_{\text{max}} \) we only need the average recall score, \( P(t) \), for a single combination of a presentation interval, \( t_2-t_1 \) or \( \tau \), and retention interval, \( t_2-t \). In the equation for the recall probability after retention interval \( t_2-t \) for two presentations, \( r_{\text{max}} \) is the only unknown variable, so its value can be calculated. To do this, we rewrite the recall probability equation for two presentations, yielding:

\[
 r_{\text{max}} = \frac{1}{\mu_1} \ln(1 - P(t)) + \frac{\mu_1 \mu_2}{a_1 - a_2} \left( e^{-a_1(t-t_2)} - e^{-a_2(t-t_2)} \right) e^{-a_1 \tau} 
\]

While we only need one such combination to determine the value for \( r_{\text{max}} \), we decided to examine more of these combinations of presentation and retention interval and their recall score. As noted above, the subjects are presented four different combinations: short presentation and short retention lag, short presentation and long retention lag, long presentation and short retention lag, long presentation and long retention lag. The definitive value of \( r_{\text{max}} \) is calculated by taking the average of the four values of \( r_{\text{max}} \) we acquired in this way.

In Figure 5.7 the overall architecture of our estimation procedure for the four single-presentation parameters and \( r_{\text{max}} \) is shown.

![Figure 5.7: Schematic view of the estimation procedure for the four single-presentation parameters and \( r_{\text{max}} \). The boxes represent the different modules, the arrows the data flow and data and resulting values are bold.](image)

**Data processing: calculating optimal schedules**

With the values for the five parameters we can calculate optimal presentation schedules. We decided to implement an optimization method for an arbitrary number of presentations, with the variable to be optimized being the recall score after a certain retention lag, as described in the
second example of Section 1 of this chapter. We use our previously described parameterization of the presentation schedule as shown in Equation 5.1. Given the values for $\mu_1, a_1, \mu_2, a_2$ and $r_{max}$, together with the value for the recall threshold, $b$, we can calculate the recall probability for any given number of presentations and retention interval. The parameters that need to be optimized are the scheduling parameters, $s$ and $T$. For this we will use a simple numerical maximization method, the hill climbing method as described above (Algorithm 1). The fact that for this optimization problem, we can use such an algorithm and that it performs fast and accurate enough lies in the problem characteristics. First off, we only need to optimize the value of two parameters and the search landscape does not contain any local minima in it as the individual error functions with respect to both $s$ and $T$ have only one maximum. For the actual program that was the implementation of this method, we again assumed $b = 1$. The entire optimization procedure is shown in Figure 5.8. To test this implementation, we designed the experiment illustrated in the next chapter.

![Figure 5.8](image_url)

*Figure 5.8: Schematic view of the procedure that calculates the presentation schedule corresponding with the (optimal) parameter values. Again, the boxes represent the different modules, the arrows the data flow and data and resulting values are bold.*
Chapter 6
Experiments

6.1 Testing the implementation: Experiment 1

To test the implementation as described in the previous chapter, we designed an experiment. This first experiment was designed to acquire the recall data for a specific type of items used. This recall data could then be used for the estimation of the model parameters, which in turn can be used to calculate optimal presentation schedules for that specific type of items again as described in the previous chapter. The results from this experiment prompted us to conduct a second experiment. In this section we describe Experiment 1 in a formal manner:

As in Glenberg’s (1976) experiment, a continuous paired associate design was used. Each trial the subject is presented with one half of the pair (the stimulus part) and is asked to give the correct response. This way we constructed a cued recall experiment. The reason that we chose our setup roughly the same as Glenberg lies in the fact that in his results, a spacing effect was found.

Subjects
The subjects were 30 persons of at least 16 years old and with Dutch as their native language. None of the subjects received a reward for participating in this experiment. They were contacted by email and asked to take part in an experiment in which they would have to learn Turkish words. None of the subjects had any significant knowledge of the Turkish language before the experiment. They were told they could do the experiment in their own time, using any computer with Internet access, but that they had to fully be concentrated and undisturbed.

Materials and Design
Each subject received 100 trials (presentations and tests). For each subject an algorithm constructed a unique order of trials. The algorithm created each specific order using some degree of randomness. The 100 trials were filled with two different lag sequences: The first sequence (‘P-T’) consisted of a single trial used as a presentation and one that served as a recall test. The second sequence (‘P-P-T’) consisted of two presentation trials and one recall trial. From these sequences single presentation data can be extracted by considering the second presentation as a test trial. First, approximately eight P-P-T sequences were assigned to an order. The remaining trials were filled with P-T sequences. The remaining trials were filled with filler items, which were not used in the data analysis.

There were four different P-P-T schedules: the presentation lag as well as the test lag could either four or fourteen intervening items. The P-T schedules had 18 different versions; here the test lag could be 1,2,3,4,5,6,7,8,9,11,13,15,17,19,24,29,39 or 49 intervening items. The fact that schedules using longer intervals are harder to fit into a limited order of trials caused that the numbers of exemplars for each schedule could not be kept same for each subject and some schedules appeared more frequently than others, as will be shown in Section 6.2. The schedules, however, where distributed as randomly as possible, in order to eliminate any effects that the distribution of schedules might have.

The items Turkish-Dutch word pairs as described in Section 5.2. The exact Turkish words and their translations can be found in Appendix B.
**Procedure**
The subjects were instructed to start an Internet browser and visit the website specifically designed for the experiment. After registration, subjects were given one trial as a test, to get acquainted with the method of the experiment. After this the subjects received their 100 paired associate trials as described in Section 5.2.

**6.2: Results from Experiment 1**
The average time it took the subjects to complete the experiment was 27 minutes. All 30 subjects were completed their 100 items. A total of 1258 single-presentation data points (consisting of a recall score on the test trial) were collected, an average of 42 per subject. Except for the retention intervals of length 4 and 14 intervening items the average number of data points per retention interval was 59.5. Because the ‘P-P-T’ schedules also provide us with single presentation data as well as with two-presentation data, the number of data points for a retention interval of four items is 199 (from the ‘P-P-T’ and ‘P-T’ schedules) and for the retention interval of fourteen intervening items we have 107 data points. From this raw data the program calculated the average recall scores for the different retention intervals. These can be seen in Figure 6.1 represented by the dots.

![Figure 6.1: The average recall scores for the single-presentation items as observed from Experiment 1. The dots represent the data points. The curve is the fitting model with a recall threshold of $b=3$.](image)

A total of 241 two-presentation data points were collected, distributed over the four P-P-T schedules. Table 6.1 shows the number of observed actions and the average recall scores at the test presentation for the four combinations of presentation interval ($\tau$) and retention interval (here denoted as $t-t_2$). We discuss these results below.
Table 6.1: sample sizes and average recall scores for the four combinations of presentation interval and retention interval as resulted from Experiment 1.

<table>
<thead>
<tr>
<th>$\tau$</th>
<th>$t-t_2$</th>
<th>sample size</th>
<th>average recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>4</td>
<td>67</td>
<td>0.761</td>
</tr>
<tr>
<td>4</td>
<td>14</td>
<td>67</td>
<td>0.446</td>
</tr>
<tr>
<td>14</td>
<td>4</td>
<td>56</td>
<td>0.704</td>
</tr>
<tr>
<td>14</td>
<td>14</td>
<td>51</td>
<td>0.471</td>
</tr>
</tbody>
</table>

Discussion

At the same time we conducted our Experiment 1, a similar experiment was done within the Neuromod group (Kruithof, 2002). This experiment examined both single-presentation and multi-presentation retention curves for the same type of items we have used, the Turkish-Dutch word pairs. In the data analysis of that experiment, the Memory Chain Model was fit to the data and this produced a fit where the value of the recall threshold was not our assumed default value of $b = 1$. Kruithof found $b = 3$, suggesting that our data too should be fitted using this higher value for the recall threshold because of the similarity of the items. As our method for parameter estimation we described in the previous chapter assumes $b = 1$ and is not equipped to work with any other value for $b$, we could not estimate the parameters using our implemented program. Fitting with our procedure indeed produced unreasonable values for the $r_{\text{max}}$ parameter. Although our experiment was initially designed to test the method for parameter estimation and schedule optimization as described in the previous chapter, we decided to do some analysis on the data using a different method. To analyze the data, we turned to the same method that was used for the fitting of Kruithof’s data. This was done using the Microsoft Excel solver. This solver was used to simultaneously minimize the chi-square statistic for both the single presentation recall data and the two-presentation data. In using the $X^2$ measure, we were able to make claims about the validity of the fits.

For the calculation of the value of the level of significance, $\alpha$, that corresponds with the minimized $X^2$ value the number of examined retention intervals is 23 (19 single-presentation and four two-presentation ‘intervals’) and the number of free variables is six: $\mu_1, a_1, \mu_2, a_2, r_{\text{max}}$ and $b$. As expected, fitting the retention data from Experiment 1 in this way with $b = 1$ did not produce an acceptable fit.1 With the minimal value of $X^2$ being 29.15 this resulted in a value for alpha of 0.047. However if we assumed a recall threshold of $b = 3$ the model fitted very well to the data. A recall threshold of $b = 2$ also produced a very good fit. Higher values for the recall threshold did not produce a valid fit. Together with the minimal $X^2$ value and the value of $\alpha$, the resulting values for the five parameters for $b = 3$ and $b = 2$ are shown in Table 6.2.

Table 6.2: Fitting scores and resulting parameter values for $b = 2$ and $b = 3$ on the data from Experiment 1.

<table>
<thead>
<tr>
<th>$b$</th>
<th>$X^2$</th>
<th>$\alpha$</th>
<th>$\mu_1$</th>
<th>$a_1$</th>
<th>$\mu_2$</th>
<th>$a_2$</th>
<th>$r_{\text{max}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>22.40</td>
<td>0.215</td>
<td>3.90</td>
<td>0.347</td>
<td>0.147</td>
<td>0.00260</td>
<td>4.22</td>
</tr>
<tr>
<td>2</td>
<td>17.92</td>
<td>0.394</td>
<td>2.80</td>
<td>0.403</td>
<td>0.138</td>
<td>0.00421</td>
<td>126</td>
</tr>
</tbody>
</table>

The recall probability function that corresponds with $b = 3$, is also plotted in Figure 6.1. The two-store Memory Chain Model only fits our data with a recall threshold of $b = 2$ or $b = 3$. For these two fits, we have two very different values for $r_{\text{max}}$. For $b = 3$, the value is very close to that of $\mu_1$, suggesting that the first store is almost saturated directly after the first presentation. For

---

1 With condition $\alpha > 0.05$
a recall threshold of $b = 2$, the value for $r_{\text{max}}$ is very high, suggesting that hardly any saturation occurs.

We designed Experiment 1 to be analyzed with our method described in Chapter 5. However, the sparse multi-presentation data do not permit us to make more specific claims from this analysis as described in this section.

For instance the occurrence of a spacing effect cannot be determined. Table 1 shows us that for a retention lag of 14 intervening items, the longer presentation lag results in a higher recall score, suggesting a spacing effect. In the case of a retention interval of four intervening items, the shorter retention interval results in a higher recall score.

To make more valid claims about the value of the recall threshold and the value of the saturation parameter and to collect evidence for the occurrence of the spacing effect in our type of experiment, we need to gather more of this kind of data. For this we designed Experiment 2, which will be described in the next section.

### 6.3 Description Experiment 2

As this experiment was designed to help us in analyzing the results from Experiment 1, it uses the same items and testing method as the first experiment. This time however in order to get more relevant data and in order to make claims about the value of $r_{\text{max}}$, we decided to gather more multi-presentation data from all subjects and not only present an item once or twice, but more often. The subjects received sequences of six presentations of the items. Because of the use of continuous paired associate items, each presentation after the first is also a test moment and this would provide us with a lot of data points to produce a good fit. The presentation schedules were also designed to investigate whether, in the case of our specific experiment material and design, a spacing effect occurred. This experiment was partly modeled after Rumelhart’s (1967) experiment as described in Chapter 2. Again, we first describe the experiment in a formal manner.

#### Subjects

The subjects were 53 psychology students, none of which had participated in Experiment 1. They were asked to participate in an experiment in which they would learn Turkish words. This group was tested in a closed test room with seven computers with Internet access located at the Faculty of Psychology of the University of Amsterdam. These subjects received a reward for their participation in the experiment. This reward was not linked to their performance in any way.

#### Materials and Design

Each subject received 204 trials (presentations and tests). Contrary to Experiment 1, in this experiment the order of the trials was the same for each of the subjects. The 200 trials were filled with items belonging to one of six possible lag sequences as used by Rumelhart (1967). The fact that these sequences consist of six presentations, does not allow for a semi-random distribution of these schedules. The six lag sequences, denoted by r9 - r14, are shown in Table 6.3. The lengths of the inter-presentation lags are again denoted as intervening items. As noted above, the sequences were chosen to distinguish between spaced (gray filled rows) and massed presentation schedules. This was done for three different lengths of the final lag between the fifth and the last presentation: a short, medium length and long lag (respectively 2, 16 and 32 intervening items).

---

1 These six schedules are the same as the schedules 9 - 14 in Rumelhart’s Experiment II, (Rumelhart, 1967).
Table 6.3: Description of the six lag sequences used in Experiment 2.

<table>
<thead>
<tr>
<th>Schedule type</th>
<th>Final lag length</th>
</tr>
</thead>
<tbody>
<tr>
<td>r9</td>
<td>32</td>
</tr>
<tr>
<td>Spaced</td>
<td>Long</td>
</tr>
<tr>
<td>r10</td>
<td>10</td>
</tr>
<tr>
<td>Massed</td>
<td>Long</td>
</tr>
<tr>
<td>r11</td>
<td>16</td>
</tr>
<tr>
<td>Spaced</td>
<td>Medium</td>
</tr>
<tr>
<td>r12</td>
<td>2</td>
</tr>
<tr>
<td>Massed</td>
<td>Short</td>
</tr>
<tr>
<td>r13</td>
<td>2</td>
</tr>
<tr>
<td>Spaced</td>
<td>Short</td>
</tr>
<tr>
<td>r14</td>
<td>2</td>
</tr>
<tr>
<td>Massed</td>
<td>Short</td>
</tr>
</tbody>
</table>

Each sequence consisted of six presentations of a single item (five presentations functioning as both a learning presentation and a recall test and one final recall test). Each sequence had four occurrences in the total of 204 trials. The remaining trials were filled with filler items that were not used in any data analysis, but served only as intervening items. Although all subjects were given the same order of sequences and filler trials, the actual items filling the trials varied between subjects to eliminate any possible effects which variable item difficulties might have. Again, in this experiment, a continuous paired-associate task was used. In this experiment, we used the same Turkish words and their translations as in Experiment 1 but other items were added. These additional words were selected to have the same frequency of occurrence in Dutch speaking language according to de Groot (1998) as the words used in Experiment 1. These words can be found in Appendix B.

Procedure

The same procedure was used as in Experiment 1. In this experiment however, the subjects received a total of 204 trials.

6.4 Results and discussion Experiment 2

Each schedule was presented four times to each of the 53 subjects, resulting in 212 observations per schedule and for each of the 36 data points. In Table 6.4, the average recall scores for each of the presentations are shown for the six presentation schedules r9-r14.

Table 6.4: Average recall scores on each presentation for the six presentation schedules resulting from Experiment 2.

<table>
<thead>
<tr>
<th>Schedule type</th>
<th>Final lag length</th>
</tr>
</thead>
<tbody>
<tr>
<td>r9</td>
<td>0.830</td>
</tr>
<tr>
<td>Massed</td>
<td>Long</td>
</tr>
<tr>
<td>r10</td>
<td>0.981</td>
</tr>
<tr>
<td>Long</td>
<td>Spaced</td>
</tr>
<tr>
<td>r11</td>
<td>0.844</td>
</tr>
<tr>
<td>Medium</td>
<td>Spaced</td>
</tr>
<tr>
<td>r12</td>
<td>0.835</td>
</tr>
<tr>
<td>Medium</td>
<td>Massed</td>
</tr>
<tr>
<td>r13</td>
<td>0.858</td>
</tr>
<tr>
<td>Short</td>
<td>Massed</td>
</tr>
<tr>
<td>r14</td>
<td>0.839</td>
</tr>
<tr>
<td>Short</td>
<td>Spaced</td>
</tr>
</tbody>
</table>

The value of 0.00472 for the average recall that occurs twice in the first column denotes the fact that on two occasions the first presentation of a word was translated correctly. Although no knowledge of Turkish was assumed, the frequency with which this occurred allows for us to ignore these values and uphold this assumption.

We have also plotted the recall scores in Figure 6.2. In Figure 6.2-A the recall scores for the three spaced schedules are shown and in Figure 6.2-B we have plotted the recall scores for the massed schedules. Because the schedules are exactly the same for the three massed schedules and for the three spaced schedules for the 2nd up to the 5th presentation, we used the average of the recall score
for these presentations (represented by the diamonds). The squares in both figures show the recall scores on the final test trial after a short, medium and long final lag.

Figure 6.2: Results from Experiment 2. Figure 6.2-A shows the recall scores for the three spaced schedules. Because the schedules are exactly the same for the 2\textsuperscript{nd} up to the 5\textsuperscript{th} presentation, we use the average of the recall score for these presentations (represented by the first four points). The last three points represent the recall scores for short, medium and long final lag. Figure 6.2-B shows the same for the three massed schedules.

With these data we can make an ad hoc observation about the spacing effect. As we can see in Table 6.4, for each of the three lengths of the final lag, the spaced schedule resulted in a higher recall score, although the effect is much larger for longer lengths of the final lag. This is consistent with Rumelhart’s findings (Rumelhart, 1967) and this shows us that our type of items and method of presentation does indeed yield a spacing effect, so that optimization of presentation schedules for this specific case should in fact be possible.

For further analysis, we fitted a two-store Memory Chain Model to these data using the same method as for the analysis of Experiment 1. As with Figure 6.2, we took the average of the data.
from the first four recall tests for the massed and the spaced groups as the data points. The Excel solver was used to minimize the value of the sum of the $X^2$ errors between the recall score and the value of the recall probability for each of the data points. This was done for different values of $b$.

We found that only for strongly elevated recall thresholds, $b \geq 4$, the model is accepted for $\alpha > 0.05$. In Table 6.5 we show the values of the model parameters and $\alpha$ that produced these best fits for a number of different values of the recall threshold parameter, $b$.

<table>
<thead>
<tr>
<th>$b$</th>
<th>$\mu_1$</th>
<th>$a_1$</th>
<th>$\mu_2$</th>
<th>$a_2$</th>
<th>$r_{\text{max}}$</th>
<th>$\alpha$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2.81</td>
<td>0.12</td>
<td>0.045</td>
<td>0.0018</td>
<td>5.45</td>
<td>2.7E-05</td>
</tr>
<tr>
<td>3</td>
<td>4.01</td>
<td>0.10</td>
<td>0.045</td>
<td>0.0021</td>
<td>6.90</td>
<td>0.011</td>
</tr>
<tr>
<td>4</td>
<td>5.17</td>
<td>0.096</td>
<td>0.045</td>
<td>0.0021</td>
<td>8.27</td>
<td>0.095</td>
</tr>
<tr>
<td>5</td>
<td>6.31</td>
<td>0.090</td>
<td>0.044</td>
<td>0.0020</td>
<td>9.59</td>
<td>0.21</td>
</tr>
<tr>
<td>6</td>
<td>7.43</td>
<td>0.085</td>
<td>0.044</td>
<td>0.0019</td>
<td>10.88</td>
<td>0.27</td>
</tr>
<tr>
<td>7</td>
<td>8.55</td>
<td>0.082</td>
<td>0.043</td>
<td>0.0018</td>
<td>12.14</td>
<td>0.27</td>
</tr>
<tr>
<td>8</td>
<td>9.65</td>
<td>0.079</td>
<td>0.043</td>
<td>0.0017</td>
<td>13.38</td>
<td>0.23</td>
</tr>
<tr>
<td>9</td>
<td>10.76</td>
<td>0.076</td>
<td>0.043</td>
<td>0.0016</td>
<td>14.61</td>
<td>0.19</td>
</tr>
</tbody>
</table>

Table 6.5: Alpha and model parameters values producing the best fit on data from Experiment 2 for different values of the recall threshold.

Discussion

The main question that arises is what these high values for the recall threshold mean. As has been detailed in Chapter 3, the recall threshold denotes the minimum number of representations that should be retrieved from the memory stores at the moment of testing for a subject to be able to produce a correct response. Almost all of the single-presentation data sets that were used to test the Memory Chain Model, as described in Chapter 3, could be fitted with a recall threshold of one, the default value. However, the single-presentation retention data from the experiment of Kruithof (2002) were fitted with an elevated recall threshold of $b = 3$ and the quality of this fit was excellent ($\alpha = 0.90$). We hypothesize that a psychological explanation for this non-default value can be found by looking at the type of items used in recall experiments: In the retention data sets described in Chapter 3 that produced the best fit with $b = 1$, the stimuli and the response part of the items were already known to the subject or very simple: a word in the subject’s native language, a digit, etcetera. The use of these uncomplicated items can account for the fact that only one memory representation has to be retrieved in order for a subject to give a correct response. In Kruithof’s experiment, however, unknown and relatively difficult items (Turkish words) were used as stimuli. This raised difficulty of the items results in a higher recall threshold, because not only the association between stimulus and response has to be retrieved, but also the appropriate previously unknown Turkish word that matches the association in the Dutch translation. This could be an explanation for the higher value of the recall threshold. A more detailed (neuro)psychological description of this explanation for the elevated recall threshold is currently being developed.

Because of the use of the same type of items in our two experiments, the above explanation could also at least partly account for our findings that our data can only be fitted with higher recall thresholds, although further research on the relation between item difficulty and recall threshold should be conducted to validate this intuition.

However, the very high values of $b$ that we find in Experiment 2 ($b \geq 4$), cannot be fully accounted for by assuming that the use of Turkish items produces this slight rise in the recall threshold. We cannot account for these multiple values of $b \geq 4$ with psychological assumptions about an elevated recall threshold. We therefore have to take a closer look at how fitting the data results in these values. Looking at Table 6.5, we see that the model is accepted for different recall
thresholds with almost equal values for $\alpha$. The fact that there is not one set of values for the model parameters that describe the data from Experiment 2 best suggests that the model is not fully able to describe the learning and forgetting curves for multiple presentations that are found in Experiment 2.

For a more detailed account of what characteristics of the recall probability curves the model exactly fails to capture, we again turn to the results of the best fits of our Experiment 2 as shown in Table 6.5. We see that the higher the value of $b$, the higher the values for $\mu_t$ and $r_{\text{max}}$ are. This is to compensate for the drop in recall probability that is caused by the higher $b$. In fitting the data, the $b$, $\mu_t$ and $r_{\text{max}}$ parameters are used to slightly alter the shape of the recall probability curves to try to produce a better fit. Higher values for $b$, $\mu_t$ and $r_{\text{max}}$ seem to produce a steeper gradient of the recall probability curves between the first two recall test trials of a schedule.

In comparing the recall probability curves of the best fitting model with $b = 3$ to the actual data from Experiment 2, we find that the model seems to be unable to describe two phenomena in the data. First, the last data point of the massed schedule for a long retention lag cannot be accounted for. Second, the model seems to be unable to account for the fast rise in average recall scores from the first to the second test trial, especially in the spaced schedules. We also find evidence for the same sort of effect in the analysis of our Experiment 1, when we look at the relatively high recall scores for twice-presented items shown in Table 6.1, in comparison to the recall scores as depicted in Figure 6.1. Although we do not have enough specific data to fully support the claim, in Experiment 1, a second presentation of an item seems to result in a higher rise in recall probability than a first presentation. Either some sort of boosting effect seems to take place at a second presentation, or the first presentation is weakened by some process. These effects are currently not assumed in our Memory Chain Model for multiple presentations as described in that chapter.

In the next chapter we will suggest a possible psychological explanation for these effects as well as a suggestion to elaborate the Memory Chain Model so that it might be able to better describe recall probability curves for multiple presentations.
Chapter 7

Conclusions, discussion and suggestions for further research

7.1 Summary and results

In this concluding chapter we will summarize our work, which we have described in the previous chapters. We will also discuss the results from the two experiments from Chapter 6 in more detail and give suggestions for further research.

In Chapter 1, we noted the value that Computer Assisted Language Learning (CALL) could have in education and how understanding and modeling the spacing effect is one of the main issues that needs to be solved in order to achieve an optimal learning tool. In Chapter 2 we elaborated on the spacing effect and described both the experimental literature on the subject as well as existing theories trying to explain it and we concluded that an overall model for learning and forgetting is needed. We gave a description of the Memory Chain Model in Chapter 3. In Chapter 4, we further expanded this model, so that it is able to describe learning and forgetting for multiple presentations of an item. We showed that the expanded model was indeed able to predict the beneficial effects of spaced learning, by using both consolidation of memories and a saturation effect. We described the behavior of the model in different extreme situations.

In Chapter 5 we described a framework for a user-model-based Computer Assisted Language Learning tool. A tool constructed according to this framework can have several advantages over the existing learning tools, some of which we noted in Chapter 5. As a start, we implemented a part of this framework, the OptiLearn tool. This tool, an online learning environment for the presentation of paired-associate items, was first tested and then used for our two experiments. It can be used by other experimenters to research online language learning. With minor adjustments, other associative learning experiments that use different types of items, such as pictures, can be done with the tool. These experiments can be done online, using the worldwide range of the Internet to do experiments with a very large number of subjects, with obvious advantages.

The OptiLearn tool is created to be a first step in constructing a CALL-system that uses information from a lot of different subjects to calculate optimal presentation schedules according to our framework. Future work on the tool, so that it would become a fully functional CALL-system has been noted in Chapter 5. Summarizing it should include:

- Dynamical adaptation of optimal scheduling by constantly updating the model using the user’s response.
- The ability to derive, and work with variable item difficulties.
- A Bayesian framework in which personal and universal data can be combined into a model that can be used for an individual model.
- The ability to work with different learning goals, learning times and learning through multiple sessions.

Using the OptiLearn tool, we have conducted two experiments, to test both the workings of the experimenting environment and the validity of the Memory Chain Model for multiple presentations as described in Chapter 4. These experiments and their results are described in Chapter 6. From these experiments we can draw a number of conclusions.
The first conclusion, a very obvious one, is that we can indeed do language learning experiments using our OptiLearn tool. Experiment 1 was done online using the Internet, and subjects did not have to show up at a specific time and place, which was experienced by both the subjects and the experimenter as a clear advantage over the usual means of experimenting. A disadvantage is of course that the environment and the conditions in which the subjects do the same experiment is different for each subject and that this may affect our data. This problem could indeed be present in our Experiment 1, where a relatively small number of subjects are used. Future researchers using the OptiLearn tool should use the advantage of the lower threshold for participation in online experiments to gather a larger number of subjects, thereby minimizing any of the negative effects of the variable learning environment. For further readings on this topic, see Janssen, Meeter and Murre (2003).

As can be found in Chapter 6, the results of Experiment 1 presented us with more questions than answers and prompted us to conduct Experiment 2. From this data, alongside the data from a simultaneously conducted experiment from Kruithof (see Murre & Chessa, submitted) we concluded that the Memory Chain Model for multiple presentations as described in Chapter 4 could not adequately describe the multi-presentation retention data from our experiments. We found that the model uses a higher value for the recall threshold parameter, instead of \( b = 3 \), to account for the fast rise in recall probability that is observed after the second presentation, especially in spaced schedules. This high value of \( b \) cannot be explained by neurobiological or psychological assumptions, so we have to assume that other processes occur in learning from multiple presentations that are not yet modeled in the model we presented.

The unexpected rapid rise in recall probability that occurred at the second presentation should be examined further, to find out under which conditions it occurs. If the existence of this ‘boosting effect’ is indeed established for a broad range of multi-presentation learning situations, the model should be altered in such a way that this phenomenon can be described. This should be done by an addition to the model that is (neuro)psychologically plausible. In the next section we present some possible explanations for the boosting effect found and give some possible additions to the model.

### 7.2 Expanding the model

After Experiment 2 terminated, we had some informal interviews with some of the participants about their experiences. We were especially interested in how the second presentation was experienced. Some participants told us that indeed they thought that they learned more from a second presentation than from any other. At the first testing moment, on the second presentation, the subjects felt reminded of the fact that they were supposed to remember the items. Although the nature of the interviews was very informal and no real conclusions can be drawn from it, this could indicate that a second presentation is indeed processed differently from other presentations. The question is what causes this difference.

The model as we have presented it in Chapter 3 and 4 handles each presentation in exactly the same way. The only factors that determine the total gain in intensity at the time, \( t \), of a presentation are the initial encoding, the store capacity and the current intensity level, respectively denoted by \( \mu_1 \), \( r_{\text{max}} \) and \( r(t) \). The interviews seem to suggest that in the brain there is some sort of mechanism that does distinguish between the encoding of a first presentation and a second presentation. In papers on learning and forgetting in honeybees by the neurobiologist Menzel, he claims that stimuli that are presented multiple times cause different chemical reactions in the brain of the bees than stimuli that are presented only once. In these papers he also argues...
that the honeybee is a suitable model for studying cognitive processes such as learning and
forgetting. (Menzel et al., 2001; Menzel, 2001). If Menzel’s conclusions and our own intuition
that a second (of third, fourth, etc.) presentation is treated differently from a first presentation are
indeed true, our model should be expanded accordingly to sufficiently describe this.

We now give a suggestion for such a possible expansion. We assume that, when an item is
presented and no memory representations are yet present in the first store (which is the case on a
first presentation), the memory is marked as being a ‘first presentation’. This presentation adds
points to the first memory store in the normal way. However, if on a presentation, there are
memory representations of the presented item present in any of the stores (in this case we have
multiple presentations) the presentation adds an extra number of points to the first memory store
on top of the $\mu r(t)/r_{\text{max}}$. This ‘superadditive’ boosting effect could be controlled by an extra
parameter that determines how much extra is added. If we call this parameter $\beta$, with $\beta \geq 1$, the
effective initial encoding of such a subsequent presentation would become: $\beta \mu r(t)/r_{\text{max}}$. Another
solution could be to have the boosting effect be dependent on the intensity level in the first store
on the moment of the presentation: the less representations are present in the first store the
stronger the boosting effect could be. This because the boosting effect seems to occur mainly in
the spaced schedules than in the massed ones.
Of course these suggestions are very roughly outlined and should be followed upon only after
other experiments have proven the existence of this boosting effect. Here, we have merely offered
suggestions for further research.
Literature


Appendix A

Proof for Equation 4.7

We prove Equation 4.7 by induction. We first show that the result holds for \( n = 2 \) trials. The intensity function for the situation (given by Equation 4.3) was written as Equation 4.4 in the text which is equivalent to Equation 4.7 for \( n = 2 \).

We now proceed by claiming that Equation 4.7 holds for \( n \) trials, of which the validity will be shown for \( n+1 \) trials (the ‘induction step’). The contribution of trial \( n+1 \) at presentation time \( t_{n+1} \) to the initial encoding is equal to:

\[
\mu_1 \left[ \frac{\mu_1 \sum_{i=1}^{n} (1 - \frac{\mu_i}{r_{\text{max}}})^{n-i} \tilde{r}(t_{n+1} - t_i)}{r_{\text{max}}} \right]
\]

The intensity function at time \( t \geq t_{n+1} \) is therefore equal to:

\[
r(t) = \mu_1 \left[ \mu_1 \sum_{i=1}^{n} (1 - \frac{\mu_i}{r_{\text{max}}})^{n-i} \tilde{r}(t - t_i) \right] + \mu_1 \left( 1 - \frac{\mu_1}{r_{\text{max}}} \right) \sum_{i=1}^{n} \left( 1 - \frac{\mu_i}{r_{\text{max}}} \right)^{n-i} \tilde{r}(t_{n+1} - t_i) \tilde{r}(t - t_{n+1})
\]

The second term of this expression is equal to:

\[
\mu_1 \tilde{r}(t - t_{n+1}) - \mu_1 \frac{\mu_1}{r_{\text{max}}} \sum_{i=1}^{n} (1 - \frac{\mu_i}{r_{\text{max}}})^{n-i} \tilde{r}(t - t_i)
\]

By adding this expression to the first term for \( r(t) \) (i.e. the intensity for \( n \) trials), we obtain:

\[
r(t) = \mu_1 \tilde{r}(t - t_{n+1}) + \mu_1 \left( 1 - \frac{\mu_1}{r_{\text{max}}} \right) \sum_{i=1}^{n} (1 - \frac{\mu_i}{r_{\text{max}}})^{n-i} \tilde{r}(t - t_i)
\]

\[
= \mu_1 \tilde{r}(t - t_{n+1}) + \mu_1 \sum_{i=1}^{n} (1 - \frac{\mu_i}{r_{\text{max}}})^{n-i} \tilde{r}(t - t_i)
\]

\[
= \mu_1 \sum_{i=1}^{n+1} (1 - \frac{\mu_i}{r_{\text{max}}})^{n-i} \tilde{r}(t - t_i)
\]

which proves the result.

Derivation of Equation 4.11 from Equation 4.10.

For the intensity in the second store we have the convolution of \( \mu_2 r_1 \) and \( \tilde{r}_2 \):

\[
r_2(t) = \int_{t_2}^{t} \mu_2 r_1(z) \tilde{r}_2(t - z) dz \text{ for } t > t_2.
\]

We split this integral into two parts, corresponding with the intervals \([t_1, t_2]\) and \([t_2, t]\):
\[ r_2(t) = \int_{t_1}^{t} \mu_2 r(z) e^{-a_2(t-z)} \, dz + \int_{t_2}^{t} \mu_1 r(z) e^{-a_1(t-z)} \, dz \]

For the first integral, we use the single-store intensity function for a single presentation, as described in Chapter 3. For the second integral, we use Equation 4.9. Equation 4.11 then results from straightforward elaboration of the integrals.

**Analytical research on Equation 4.18**

- If \( a_2 = 0 \) and \( r_{\text{max}} \to \infty \), Equation 4.18 can be rewritten as:
  
  \[ r(t) = \mu_1 (1 - \frac{\mu_2}{a_1}) \sum_{i=1}^{n} e^{-a_2(t-t_i)} + n \frac{\mu_1 \mu_2}{a_1} \]

  If \( \mu_2 < a_1 \), \( r(t) \) increases as the sum-term increases. For all presentations \( i \), this term is maximal if \( t_i = t \), suggesting a massed schedule.

  If \( \mu_2 > a_1 \), \( r(t) \) increases as the sum-term decreases, because it is multiplied with a negative term. The intensity function is at a maximum if the sum-term approaches 0, which, for all presentations \( i \), is the case if \( t_i - t_i \) is as large as possible.

  If \( \mu_2 = a_1 \), the intensity function is equal to \( \mu/n \) which is independent of the \( t_i \).

- If \( a_2 = 0 \) and \( r_{\text{max}} = \mu_1 \), Equation 4.18 can be rewritten as:

  \[ r(t) = \mu_1 (1 - \frac{\mu_2}{a_1}) e^{-a_2(t-t_i)} + \frac{\mu_1 \mu_2}{a_1} \sum_{i=2}^{n} (1 - e^{-a_2(t_i-t_i-1)}) \]

  for all \( i = 2, \ldots, n-1 \), we calculate the partial derivative:

  \[ \frac{\delta r}{\delta t_i} = \frac{\mu_1 \mu_2}{a_1} e^{-a_2(t_i-t_i-1)} a_1 + \frac{\mu_1 \mu_2}{a_1} (-e^{-a_2(t_i-t_i-1)}) a_1 \]

  By setting these derivatives to zero and elaborating the results we get that for each presentation \( i \) the optimal interpresentation lag is given by:

  \[ t_i = \frac{1}{2} (t_{i-1} + t_{i+1}) \]

  Thus, the optimal moment for a presentation is exactly between the previous and next presentation, resulting in an equidistant schedule.

- If \( a_2 > 0 \) and \( r_{\text{max}} \to \infty \), we can rewrite Equation 4.18 to:

  \[ r(t) = \mu_1 \sum_{i=1}^{n} e^{-a_2(t-t_i)} + \sum_{i=1}^{n} \frac{\mu_1 \mu_2}{a_1 - a_2} (e^{-a_2(t-t_i)} - e^{-a_2(t_i-t_i)}) \]

  We take the partial derivatives:
\[
\frac{\delta r}{\delta t_i} = \mu_i a_i e^{-a_i(t_{i+1} - t_i)} + \frac{\mu_{i+1} \mu_2}{a_i - a_2} \left( a_2 e^{-a_2(t_{i+1} - t_i)} - a_1 e^{-a_1(t_{i+1} - t_i)} \right).
\]

Equalizing these to zero yields:

\[
(a_1 - a_2)t_i = \ln \left\{ \frac{\frac{\mu_2 a_2}{a_i - a_2} e^{(a_i - a_2)t}}{a_1 \left( \frac{\mu_2}{a_i - a_2} - 1 \right)} \right\}
\]

which can be elaborated to Equation 4.19 for the optimal moment for presentation \(i\):

\[
\tilde{t}_i = t - \frac{1}{a_1 - a_2} \ln \left\{ \frac{a_1}{a_2} \left( 1 - \frac{a_1 - a_2}{\mu_2} \right) \right\}
\]

Now, we see that if \(\mu_2 < a_1\) the term within the curly brackets is negative. This will lead to the extreme situation where for all \(i\): \(t_i = t\). Otherwise, if \(\mu_2 > a_1\), \(t_1 = \ldots = t_n < t\).

- For the case where \(a_2 > 0\) and \(r_{\text{max}} = \mu_1\), rewriting Equation 4.18 yields:

\[
\begin{align*}
\sigma(t) &= \mu_1 e^{-a_1(t_{i+1} - t_i)} + \frac{\mu_1 \mu_2}{a_1 - a_2} \sum_{i=2}^{n} \left( 1 - e^{-a_i(t_{i+1} - t_i)} \right) e^{-a_2(t_{i+1} - t_i)} + \frac{\mu_1 \mu_2}{a_1 - a_2} \left( e^{-a_2(t_{i+1} - t_i)} - e^{-a_1(t_{i+1} - t_i)} \right)
\end{align*}
\]

We take the partial derivative with respect to \(t_i\) and we get:

\[
\frac{\delta \sigma}{\delta t_i} = \frac{\mu_1 \mu_2}{a_1 - a_2} a_i e^{-a_i(t_{i+1} - t_i)} e^{-a_2(t_{i+1} - t_i)} + \frac{\mu_1 \mu_2}{a_1 - a_2} \left( 1 - e^{-a_i(t_{i+1} - t_i)} \right) a_2 e^{-a_2(t_{i+1} - t_i)}
\]

\[
+ \frac{\mu_1 \mu_2}{a_1 - a_2} a_i e^{-a_i(t_{i+1} - t_i)} e^{-a_2(t_{i+1} - t_i)}
\]

Equaling this equation to zero and after simplification we get the following equation denoting the optimal lag between presentations \(i\) and \(i+1\) in terms of the model parameters and the preceding lag \(t_{i-1}\):

\[
t_{i+1} - t_i = -\frac{1}{a_1 - a_2} \ln \left\{ \frac{a_2}{a_1} + \frac{a_2}{a_1} e^{-a_1(t_{i+1} - t_i)} \right\}
\]

According to this relation, the optimal presentation lags form a ‘shrinking rehearsal’ schedule, which we will prove below.

We use the property that the logarithmic function is strictly concave. Since \(a_1 > a_2\), the argument of the logarithm in the equation above is a convex combination of the value 1 and \(\exp( -a_1(t_{i+1} - t_i) )\). Strict concavity of \(\ln\) implies that
This means that

$$(t_{i+1} - t_i) < \frac{1}{a_1 - a_2} - a_1 (1 - \frac{a_2}{a_1})(t_i - t_{i-1})$$

and so:

$$(t_{i+1} - t_i) < (t_i - t_{i-1})$$

For all $i = 2, \ldots, n-1$, this denotes that every presentation lag is smaller than its preceding lag: QED. This result holds for any choice of the parameter values, with $a_2 > 0$ and $r_{\text{max}} = \mu_1$.

**Analytical rough estimator: Equations 5.2 and 5.3**

The input data consists of a set of recall scores $R_{i,t}$ for different retention intervals $t$ belonging to some finite set $T$. For every retention interval $t \in T$, we define the error-rate:

$$\hat{v}_i = \frac{1}{k} \sum_{i=1}^{k} (1 - R_{i,t})$$

for $k$ scores at $t \in T$.

From the single-store model with recall threshold $b = 1$, we derive that:

$$v(t) = 1 - p(t) = e^{-\mu e^{-at}}.$$  

$\hat{v}_i$ is an estimator for $v(t)$. From here on we will use the relation

$$-\ln(v(t)) = \mu e^{-at} \text{(**)}$$

From this relation we will now obtain two inequalities. Let $T_1$ and $T_2$ denote the partition of $T$. From (***) follow the two equalities:

$$\prod_{\mathfrak{t} \in T_1} \ln(v(t))^{-1} = \prod_{\mathfrak{t} \in T} \mu e^{-at} \text{ and } \prod_{\mathfrak{t} \in T_2} \ln(v(t))^{-1} = \prod_{\mathfrak{t} \in T_2} \mu e^{-at}$$

The left and right terms can be written as geometrical means. We then get:

$$\Phi_1 = \mu e^{-az_1} \text{ and } \Phi_2 = \mu e^{-az_2} \text{, where}$$
\[ \Phi_i = \left( \prod_{t \in T_i} \ln(v(t)^{-1}) \right)^{1/2}, \quad i = 1, 2 \text{ and} \]

\[ Z_i = \frac{1}{|T_i|} \sum_{t \in T_i} t, \quad i = 1, 2. \]

For the estimators, we divide (***) by (****) and we get:

\[ \frac{\Phi_1}{\Phi_2} = e^{a(Z_2 - Z_1)} \Rightarrow \hat{a} = \frac{1}{Z_2 - Z_1} \ln \left( \frac{V_1}{V_2} \right) \]

and for \( \mu \), from (***) follows that:

\[ \hat{\mu} = V_1 e^{\hat{a}Z_1} \]

where \( V_1 \) and \( V_2 \) are estimators for \( \Phi_1 \) and \( \Phi_2 \), obtained by substituting \( \hat{v}_i \) in \( \Phi_1 \) and \( \Phi_2 \) for \( v(t) \).
Appendix B

Dutch-Turkish word pairs, used as the paired-associate items for Experiment 1 and 2.

<table>
<thead>
<tr>
<th>Stimulus</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>kasap</td>
<td>slager</td>
</tr>
<tr>
<td>kunduz</td>
<td>bever</td>
</tr>
<tr>
<td>hata</td>
<td>fout</td>
</tr>
<tr>
<td>pancar</td>
<td>biet</td>
</tr>
<tr>
<td>mutfak</td>
<td>keuken</td>
</tr>
<tr>
<td>renk</td>
<td>kleur</td>
</tr>
<tr>
<td>zarar</td>
<td>verlies</td>
</tr>
<tr>
<td>aslan</td>
<td>leeuw</td>
</tr>
<tr>
<td>levha</td>
<td>bord</td>
</tr>
<tr>
<td>beden</td>
<td>lichaam</td>
</tr>
<tr>
<td>kitap</td>
<td>boek</td>
</tr>
<tr>
<td>potin</td>
<td>laars</td>
</tr>
<tr>
<td>burun</td>
<td>neus</td>
</tr>
<tr>
<td>tava</td>
<td>pan</td>
</tr>
<tr>
<td>sokak</td>
<td>straat</td>
</tr>
<tr>
<td>elma</td>
<td>appel</td>
</tr>
<tr>
<td>silgi</td>
<td>gum</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Stimulus</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>damar</td>
<td>ader</td>
</tr>
<tr>
<td>vapur</td>
<td>schip</td>
</tr>
<tr>
<td>kural</td>
<td>regel</td>
</tr>
<tr>
<td>turp</td>
<td>radijs</td>
</tr>
<tr>
<td>kral</td>
<td>koning</td>
</tr>
<tr>
<td>horoz</td>
<td>haan</td>
</tr>
<tr>
<td>zemin</td>
<td>vloer</td>
</tr>
<tr>
<td>diz</td>
<td>knie</td>
</tr>
<tr>
<td>dost</td>
<td>vriend</td>
</tr>
<tr>
<td>sucuk</td>
<td>worst</td>
</tr>
<tr>
<td>tepe</td>
<td>berg</td>
</tr>
<tr>
<td>dolap</td>
<td>kast</td>
</tr>
<tr>
<td>erkek</td>
<td>man</td>
</tr>
<tr>
<td>hekim</td>
<td>arts</td>
</tr>
<tr>
<td>kirpi</td>
<td>egel</td>
</tr>
<tr>
<td>tutum</td>
<td>gedrag</td>
</tr>
<tr>
<td>lahana</td>
<td>kool</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Stimulus</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>tavan</td>
<td>zolder</td>
</tr>
<tr>
<td>ders</td>
<td>vak</td>
</tr>
<tr>
<td>destek</td>
<td>steun</td>
</tr>
<tr>
<td>kurt</td>
<td>wolf</td>
</tr>
<tr>
<td>kazan</td>
<td>ketel</td>
</tr>
<tr>
<td>insan</td>
<td>mens</td>
</tr>
<tr>
<td>defter</td>
<td>schrift</td>
</tr>
<tr>
<td>kemer</td>
<td>riem</td>
</tr>
<tr>
<td>kulak</td>
<td>oor</td>
</tr>
<tr>
<td>kase</td>
<td>kom</td>
</tr>
<tr>
<td>kent</td>
<td>stad</td>
</tr>
<tr>
<td>armut</td>
<td>peer</td>
</tr>
<tr>
<td>kalem</td>
<td>pen</td>
</tr>
<tr>
<td>sinir</td>
<td>zenuw</td>
</tr>
<tr>
<td>araba</td>
<td>auto</td>
</tr>
<tr>
<td>yasak</td>
<td>verbod</td>
</tr>
<tr>
<td>marul</td>
<td>sla</td>
</tr>
</tbody>
</table>