A neural modelling approach to investigating general intelligence

by

Daniel Rasmussen

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2010

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

One of the most well-respected and widely used tools in the study of general intelligence is the Raven's Progressive Matrices test, a nonverbal task wherein subjects must induce the rules that govern the patterns in an arrangement of shapes and figures. This thesis describes the first neurally based, biologically plausible model that can dynamically generate the rules needed to solve Raven's matrices. We demonstrate the success and generality of the rules generated by the model, as well as interesting insights the model provides into the causes of individual differences, at both a low (neural capacity) and high (subject strategy) level. Throughout this discussion we place our research within the broader context of intelligence research, seeking to understand how the investigation and modelling of Raven's Progressive Matrices can contribute to our understanding of general intelligence.

Acknowledgements

Thank you to my supervisor, Chris Eliasmith, and the other members of the Centre for Theoretical Neuroscience for their invaluable assistance throughout this research.

Dedication

To my family for getting me here, and Gill for her constant support and encouragement.

Contents

List of Figures

Chapter 1

Introduction

1.1 General intelligence

Intelligence is, by its very nature, complex and difficult to circumscribe. Nevertheless, some definition must be agreed upon in order to provide a place to begin. In 1994 a group of leading researchers in the study of intelligence, from a broad range of disciplines, signed their names to this definition:

> Intelligence is a very general mental capability that, among other things, involves the ability to reason, plan, solve problems, think abstractly, comprehend complex ideas, learn quickly and learn from experience. It is not merely book learning, a narrow academic skill, or test-taking smarts. Rather, it reflects a broader and deeper capability for comprehending our surroundings—"catching on", "making sense" of things, or "figuring out" what to do. (Gottfredson, 1997)

This definition is necessarily ambiguous, but it succeeds in capturing in broad strokes what we mean by general intelligence. It is a desire to better understand this "catching on", "making sense", and "figuring out" that motivates the work presented here; what is it, how does it arise in the brain, and how can we model it in a computational framework.

1.1.1 History

An important aspect of the above definition is that intelligence involves performance across a broad range of tasks. A person who performs very well at only one task we might call highly skilled or expert, but the label of highly intelligent is reserved for those who

demonstrate an ability to excel in a number of different areas. This was first stated formally by Spearman in 1904. Spearman observed that individuals that performed well on one test of intellectual ability tended to perform well on other tests of intellectual ability. This is not a surprising observation, but Spearman's contribution was to develop techniques to quantify this tendency, eventually leading to his invention of factor analysis. What he had discovered was that there was a common factor underlying performance across all of these tasks, and he called this factor g and coined the term "general intelligence".

A central debate in intelligence research is whether g represents a single cognitive component or a network of more fine-grained abilities. Spearman leaned toward the former, suggesting that g was attributable to a general "mental energy", while the latter was first championed by Thomson (1939). Thomson's theory was that human mental ability depended on many low-level processing components, and that different intelligence tests sampled from different sets of those components. To the extent that those sets overlap, performance on those tests will be correlated, and this is what gives rise to g. The problem is that if the tests are sufficiently overlapping it is impossible to tell the difference between Thomson's conception of g and Spearman's conception of g. In order to differentiate them, researchers need to find non-overlapping groups of these components that can be tested experimentally. The attempt to find such divisions—and debate over whether or not they are valid—continues to the present day and defines much of intelligence research.

1.1.2 Components of general intelligence

We will discuss just one of the most general and well accepted divisions of intelligence, as it is most relevant to our research. In 1987, Cattell introduced the idea of fluid versus crystallized intelligence (g_f versus g_c). Fluid intelligence represents novel problem-solving ability; it is the general capacity to be presented with a new situation or problem, make sense of it, and act effectively. Crystallized intelligence represents the accumulation and use of knowledge; it is the insight and ability gained throughout a lifetime. For example, when an accomplished musician plays a piece of music, they are using their crystallized intelligence. If they became stranded on a desert island and managed to build an instrument out of coconuts, that would be demonstrating fluid intelligence.

This distinction could also be understood through its parallel with inductive versus deductive reasoning. Inductive reasoning is the ability to extract general conclusions from specific examples—to achieve new insight beyond the facts that are given. Deductive reasoning is the ability to work out the necessary conclusions of a body of knowledge—to find the implicit knowledge contained within the explicit. For example, if I know that Andy is taller than Bob, Bob is the same height as Charlie, and Charlie is taller than Dennis, I could deductively reason that Andy is taller than Dennis. Suppose I also know that Andy is older than Bob, Bob and Charlie are the same age, and Dennis is the youngest. If I notice

the parallel between these two sets of facts and extract the general hypothesis that people get taller as they get older, that would be a demonstration of inductive reasoning. This illustrates the relationship between the two concepts: fluid intelligence/inductive reasoning involve the creation of new insight beyond what was known previously, whereas crystallized intelligence/deductive reasoning involve the use and manipulation of a body of knowledge.

These examples also illustrate that the division between fluid and crystallized intelligence is not clear-cut. When the musician builds a new instrument out of coconuts he is indeed engaging in novel problem solving, but he is certainly helped in this task by his past knowledge of how instruments work. Nevertheless, this is the most successful division of g that has been developed since Spearman introduced it; it is in common usage today and continues to receive experimental and theoretical support (Gray et al., 2003; Perfetti et al., 2009).

This thesis focuses on fluid ability; our goal is to understand and model this process of generating new solutions when confronted with a novel problem. Our motivation for choosing to focus on fluid intelligence rather than crystallized is that most cognitive modelling in this area has been focused on crystallized ability, trying to understand how to apply previously created abilities/knowledge to the current problem. As mentioned, intelligent behaviour almost always involves a combination of both aspects, and so a better understanding of fluid intelligence is a valuable addition to the overall study of intelligent behaviour.

1.2 Intelligence testing

We apply our research in the domain of intelligence testing, and so provide this brief overview of the field.

1.2.1 History

The theory of intelligence has always progressed hand-in-hand with attempts to measure and quantify that intelligence. The earliest intelligence testing was done by Galton (1892), who was studying the heritability of intelligence. He attempted to measure the intelligence of a person based on their general success in life, and then did the same for their relatives and looked for correlations—a new concept—between them. He is often faulted for his results, which perhaps overemphasized the importance of being a white aristocrat, but his methods are not dissimilar to modern research into heredity. Even the ideas that intelligence could be measured and passed on to children were great steps, at a time when evolution was still heavily debated.

One problem with Galton's research was his measure of intelligence; selecting "eminent" and "illustrious" men in society, although perhaps sharing a loose correlation with intelligence, is a very indirect method of assessing intelligence as we now understand it. Binet (1905) was the first to apply more scientific principles to intelligence testing. Binet's work was with school children, attempting to identify gifted or handicapped individuals. His methods laid the template for all future intelligence tests: he developed a collection of tests across various domains and levels of complexity, and collected standardized scores on those tests (norms) for schoolchildren at various grade levels. He could then compare a given student's score to the average score for a child of their age, and determine whether they were ahead or behind of what was expected (taking into account standard deviations in intelligence). Binet's tests—later called the Binet-Simon and then Stanford-Binet intelligence scales—became very popular, and remain in use (in revised form) today.

1.2.2 Controversy

Several controversial issues arise in the context of intelligence testing, and although the measurement of intelligence is neither the basis nor goal of our current research, we address these issues here in order to clarify our position.

The first topic often raised is that there is no such thing as general intelligence, that everyone has unique strengths and to try to assign a number to that complexity is impossible. In other words, it is impossible to measure intelligence. This is an argument that cannot really be responded to, as it presupposes that all the evidence to the contrary is meaningless. Intelligence as we have been discussing it here, a general capacity to "do well" at things, has been shown to be reliably quantifiable for over a century. The body of scientific evidence is overwhelmingly in support of the position that people do differ in intellectual ability, and that these differences are quantifiable. However, saying that these differences are quantifiable does not imply that measured IQ is a complete description of a person's ability. Intelligence is quantifiable in the same way in which we can quantify global climate; we can reliably measure overall levels and trends, while still allowing for the complex variability of weather in any one area.

The second idea is that we may be able to measure intelligence, but that it is just a meaningless number and bears no real relation to everyday life. Again this is a position that is contradicted by the majority of scientific evidence. There are strong correlations between measured intelligence and a broad range of life outcomes, such as school performance, social status, income, job performance, and crime rate (Neisser et al., 1996). Note that these are only correlations, they do not necessarily establish causative links (i.e. they cannot tell us in a straightforward manner whether intelligent people do better in school or people who do well in school become more intelligent). However, they do tell us that these variables

are reliably linked—whatever it is that we are measuring with intelligence tests, that thing also has an important impact on our lives.

A third concern is that the measurement of intelligence is inherently culturally biased, and so is more a measure of the environment we grow up in rather than innate ability. This is a valid concern, and one which all intelligence tests must struggle with. It will always be impossible to eliminate all biases from a test, but test makers do make use of increasingly advanced methodology to minimize their effect. There are two main approaches. The first is to minimize the culture-specific content of the test. This has led to the increasing prevalence of abstract visual problem solving tests; these tests rely less on language and accumulated knowledge, and therefore are less dependent on the culture in which the subject is raised. The second approach is to better understand the cultural differences, so that their effects can be better understood. Work in this area involves establishing norms for different cultural groups, as well as genetic research. Through these two approaches cultural effects can be minimized, so that scores better reflect individual rather than group differences.

The fourth concern is that measuring intelligence implies some type of IQ determinism. In other words, if intelligence is this quantitative, measurable thing then it must determine success in a fixed way, and it thereby diminishes the importance of other personality traits such as hard work. However, this is not a claim that most intelligence researchers would make. Even Binet (1905) recognized that intelligence scores were highly plastic and only an approximation; in other words, various factors can cause intelligence scores to change over time, and a person's intelligence score does not completely predict their performance on any task. As intelligence testing has progressed, researchers' appreciation of these variations has only increased. Intelligence tests are quite accurate at determining how, in general, a person will perform relative to their peers; few researchers would attempt stronger claims than that.

The fifth concern is the social implications of intelligence testing. These concerns have more to do with how intelligence results are used, rather than the tests themselves. Examples include using intelligence tests in a job hiring process, or to divide students into low achievers and high achievers. Given the limitations outlined in the previous paragraphs, there are serious concerns about a person being "misclassified" based on such simplistic use of an intelligence score. However, the fault in this case lies with the tool-user, not the tool. Intelligence tests can be quite useful measures, but they should not be used as a magic wand by people who do not understand their limitations. A similar concern is the use of intelligence tests to measure group differences. Unfortunately, tests have been used in the past to support inaccurate, racist claims, and this has cast a shadow over any research in this area. However, this does not imply that intelligence testing itself is flawed. There are many interesting research areas in group differences that would help us better understand people and societies, and the expansion of knowledge in this area should be

pursued scientifically, not avoided.

1.2.3 Current work in intelligence testing

Intelligence testing has come a long way since Galton. Tests benefit from what has been learned about the structure of human cognition, allowing them to more accurately target the faculties they intend to test. Tests are also now subjected to far more rigorous analysis to ensure validity, reliability, and to establish norms. For example, in a 1989 review of norming studies for just one test of general intelligence, Raven's Progressive Matrices, the author cited over a dozen different studies (Raven, 1989), each involving thousands of subjects, and these studies continue to the present day (see Van De Vijver (1997); Bors and Stokes (1998); Brouwers et al. (2009)).

In addition, intelligence is usually assessed by a battery of such tests. This serves two purposes: first, it helps get a more complete picture of the subject's capabilities by employing tests targeted at various domains, and second, it helps ensure that any weaknesses in one test (such as a slight cultural bias) have less impact on the overall score, being balanced out by other tests.

These types of tests are now widely used in many different areas: education, research, workplace, military, and more. The SAT, MCAT, GRT, and LSAT are all examples of tests that follow in this tradition, attempting to assess the subject's underlying general ability to "do well". Despite the complaints often levelled against them, intelligence tests are an important part of how we assess individuals' capabilities (and, it must be said, a much less biased technique than interviews or reference letters). As such, the ongoing effort to better understand these tests—what they are testing, and how they test it—is an important area of research in understanding human ability.

1.3 Raven's Progressive Matrices

We apply our research to a specific intelligence test, Raven's Progressive Matrices, which we introduce here. A more in-depth analysis is undertaken in Chapter 3.

1.3.1 Description

The first version of the RPM was developed by John C. Raven in 1936. It caught on quickly; as early as 1940, Cattell was discussing it as a good "culture-free" test of intelligence. Over

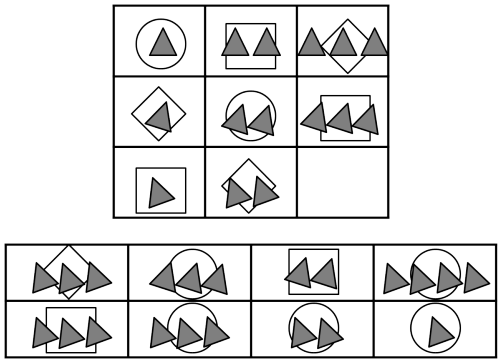

Figure 1.1: An example Raven's-type matrix.

the years the test's popularity has continued to increase (Brouwers et al., 2009), and first John C. Raven and then his son John Raven have continued to maintain the test.[1]

The RPM consists of a series of items wherein subjects must determine which answer completes the pattern presented in the matrix. An example matrix is shown in Figure 1.1. To solve these matrices the subject needs to examine the first two rows or columns and discover the rules which link those cells. They can apply the rules to the last row/column to determine which answer (among the 8 possibilities given) belongs in the blank cell. All the matrices can be solved equivalently using row-wise or column-wise patterns, although experimental analysis has shown that subjects almost always proceed in a row-wise fashion (Carpenter et al., 1990), perhaps due to a reading bias.

The matrices are arranged so that they become increasingly complex as subjects proceed through the test. They are designed to cover the full spread of ability of their target subjects; almost everyone should get the early items correct, while very few should solve the later ones. The tests are also administered with a time limit (although some studies may remove this component), so subjects must manage their time on each item.

There are three different versions of the RPM. The Standard version was developed first, and is used to assess intelligence in school-age children. However, one important aspect of intelligence tests is that they are only good at assessing subjects that fall within their range. If subjects do too well (getting almost all the items correct), then there is no way to differentiate them. Similarly, if subjects do too poorly, getting very few items correct, then little information has been gained. The Advanced and Coloured versions were developed to cover these two ends of the spectrum, respectively. The Advanced is used to assess adults and above-average adolescents, while the Coloured is used to assess young

[1]Initially this involved fairly dramatic changes, such as adding, deleting, or rearranging items, but since the 1970s it has consisted primarily of renorming every few years and publishing updated manuals.

children or subjects with cognitive deficits. All the tests follow the structure described above, they differ only in the complexity of the matrices. In our research we focus on the Advanced version of Raven's Progressive Matrices. Our goal is to understand the functioning of a normal, mature human brain, and so the Advanced is best targeted at the cognitive abilities we are interested in.

1.3.2 Why the RPM?

There are several reasons why we have chosen to apply our research to the RPM, rather than one of the many other intelligence tests available.

First, the RPM is one of the most popular tests; it is widely used in clinical settings, as well as for research purposes, where assessing a subject's general mental ability is often necessary (such as in fMRI studies). This popularity provides two benefits: it increases the value of better understanding the test (as any gains in knowledge are immediately applicable to a broad range of work), and it ensures that there is a broad base of previous research into the behavioural and cognitive mechanics of the test from which to build.

Second, the RPM is an abstract, nonverbal test, meaning that it does not rely on language for input or output, and does not require linguistic processing in order to solve. This is helpful, because those are two complex topics that we can put aside in our model. Understanding the cognition behind this type of problem solving is complicated already, without adding in the complexity of language processing.

Third, the RPM is highly correlated with g (Marshalek et al., 1983). This is important because what we are truly interested in is general intelligence; we are only interested in intelligence tests insofar as they provide insight into general intelligence. The high correlation between the RPM and g is the best indication that whatever the cognitive faculties necessary to solve the RPM are, they are also central to general intelligence.

Fourth, the RPM taps into dynamic, fluid rather than crystallized intelligence (see Section 1.1.2). In other words, it does not require a large base of accumulated knowledge in order to solve. This has two advantages: it means that the test is less culturally biased (i.e. it depends more on the innate characteristics of the subject and less on the environment in which they were raised), and it allows us to focus more on modelling the actual problem solving processes rather than dealing with memory systems.

In summary, the RPM is one of the best and most well-studied tools we have available to probe the nature of general intelligence, and it allows us to focus on dynamic problem solving ability. It is for these reasons that we decided to apply our research in this domain.

1.4 The current project

Broadly speaking, our goal is to better understand the cognitive processes involved in general intelligence by recreating those processes in a biologically plausible neural model. There are two parts to this task. First, we seek a better understanding of the high-level functions that underly g (focusing on fluid intelligence/inductive reasoning). What are they? How is it that people go about solving these problems? Why do some people perform better than others? Second, we seek to understand how those functions are implemented in the brain. That is, we do not just want to be able to solve these problems, we want to understand how the human brain solves these problems. How do neurons carry out these tasks? What kinds of constraints does this impose on the higher level cognitive behaviour? How do these functions relate to the underlying neuroanatomy?

To this end we have built a computational model, implemented in simulated spiking neurons, that is capable of carrying out these kinds of operations. In order to examine the effectiveness of this model in the real world problems of general intelligence, we have applied it to the domain of Raven's Progressive Matrices. The success of the model is demonstrated in its ability to successfully solve these matrices, as well as in the interesting insights and predictions into these aspects of general intelligence that the model provides.

Chapter 2 delves more into the neural aspects of general intelligence, which have been largely absent from the discussion up until now, while Chapter 3 discusses previous theoretical and experimental work on Raven's Progressive Matrices. Chapter 4 describes the model we have developed; how it is implemented at both a low and high level, and how that relates to the underlying neuroanatomy. In Chapter 5 we present results from the model, and then we conclude in Chapter 6.

Chapter 2

The Biological Basis of Intelligence

Ever since the scientific enquiry into intelligence began, researchers have been trying to understand why some people are smarter than others. Differences in brain structure or function certainly underly these differences, but the complexity of the brain makes it difficult to identify specific differences related to general intelligence.

2.1 How do brains differ?

2.1.1 General factors

Brain size

The problem with studying the brain is that although it is a very structured system, it is difficult to separate out particular sections for study. Unlike da Vinci, who was able to dissect the human hand in exquisite detail and so learn its parts, the dissection of a brain offers much less insight into its functional components. Thus the early work in this area was focused on broad measurements of the brain.

The most obvious such measure is head size (the logic being that bigger brains require bigger heads). This work was again pioneered by Galton (1888). Galton made a rough calculation of head volume, and then looked for correlations with intelligence (in this case, measured by students who did/did not receive honours when graduating from Cambridge). This is obviously a very crude method of measuring brain size, but nevertheless he found significant correlations between the two variables.

A better measure of brain size was employed by Broca (1861), who weighed the brains of cadavers and compared the weight to the age and occupation of the subject. Broca found

that skilled workers tended to have heavier brains than unskilled workers, again indicating a link between brain size and intelligence. He found similar results when measuring skull size, lending weight to Galton's results. The fact that these two researchers were able to find any relation at all is impressive, given how noisy their measures of brain size were.

The advent of Computed Tomography (CT) scans and Magnetic Resonance Imaging (MRI) greatly increased the resolution with which the volume of the brain could be estimated, thereby allowing more accurate comparisons between size and g. In a review of 28 such studies, containing 1389 subjects, Rushton and Ankney (2009) find an average correlation of 0.4. This is quite a significant correlation, especially considering that we are only looking at whole brain volume, as there are certainly large parts of the brain not significantly involved in general intelligence.

MRI also allows us to look more closely at specific components of the brain. Narr et al. (2007) examined the relationship between g and grey matter volume (the area of the brain dense in neuron bodies), white matter volume (the area dense in connections between neurons), and cerebrospinal fluid volume. Correlations with grey and white matter volume were significant ($r = 0.36$ and $r = 0.26$, respectively), while CSF volume was not. It is not surprising to find this correlation with grey/white matter, given that we know overall brain volume (of which grey/white matter are a part) correlates with g. However, CSF is also a part of brain volume, and the lack of correlation there illustrates the point that it is not simple volume that is important, but the amount of "thinking hardware" (grey and white matter).

Brain structure

Recent advances in neuroimaging, such as Diffusion Tensor Imaging (DTI), allow researchers to look in detail at white matter tracts, allowing them to see how the brain is connected. This allows us to investigate the idea that the organization of a brain is important in addition to its size.

This work is still in its early stages, but a strong and consistent result is that the human brain is organized in a "small world" fashion (Sporns and Zwi, 2004). This means that there are relatively densely connected local clusters, with occasional cluster-cluster connections. This has two advantages: the highly connected clusters allow each module to be highly parallel and robust, and the cluster-cluster connections allow modules to communicate while keeping the average connection distance low (desirable because long-range connections are more noisy and expensive to maintain).

These results tell us about the average structure of the human brain, but they do not shed light on why some brains are smarter than others. However, a reasonable hypothesis would be that given that the brain is organized in this "small world" fashion, which seems

to be a good thing, those brains that are more "small world" will be more intelligent. Li et al. (2009) investigated this idea by creating individual network maps of subjects' brains. They could then analyze those maps to extract graph-theoretic properties, and compare those properties to measured IQ. They found that higher numbers of clustered connections, shorter average path length, higher global efficiency (reflecting how well the network can transfer information), and higher local efficiency (a measure of how robust each cluster is to the removal of nodes) were all correlated with higher intelligence. These results are relatively new and so not as robust as those discussed in the previous section, but they do support the idea that the structure of the brain is important as well as its size.

Brain activity

The advent of functional neuroimaging allows researchers to investigate the relationships between general brain activity and intelligence. The primary discovery in this area is seemingly counterintuitive: higher intelligence is correlated with *less* overall activation on a given task. This was first reported by Haier et al. (1988), who used Positron Emission Tomography (PET) to examine subjects' neural activation when solving Raven's Progressive Matrices. They found that subjects who scored higher tended to have lower average activity while carrying out the task. A similar result was found in a number-based rule finding task using Electroencephalography (EEG) by Neubauer and Fink (2003).

These results at first seem strange; the obvious assumption would be that bigger, harder working brains would be *more* active and thereby perform better on intelligence tests. However, when viewed in light of the structure results discussed in the previous section, we see that intelligence has more to do with efficiency than brute force. This theory is supported in the functional domain by another study by Haier et al. (1992), who found that as subjects improved at a cognitive task (in this case, Tetris), those who improved more were those who showed a larger decrease in overall cortical activity. Importantly though, although these subjects showed a larger decrease overall, they showed increased regionalization of activity; in other words, they moved from high, diffuse cortical activity to high, localized cortical activity (which appears as less activity overall). This fits nicely with the "small world" organization; higher performing subjects organize their activity into tight, densely connected modules, rather than spreading it inefficiently across the whole network.

These three sections—brain size, brain structure, and brain activity—now present a clearer picture of the broad factors determining general intellectual ability. First of all, bigger is better: larger brains have more processing power (of some form) available to them, and thereby perform better on tests of intelligence. Second, smarter brains make better use of their resources, both through a better structured neural architecture and by utilizing that architecture more efficiently while carrying out a task.

2.1.2 Specific factors

In the previous section we looked at general ways in which brains differ, and the impact that might have on performance. However, there are also many theories that attribute differences in performance to very specific brain functions (and the associated neuroanatomy). Essentially, the contrast is whether general intelligence is general because it involves the brain generally, or whether it is general because there is a specific component underlying all intelligent performance. Likely both perspectives are true and impact intelligence in some way, so in this section we will review some of the most prominent results of the latter theory.

Working memory

The most popular single factor said to underly general intelligence is working memory. Working memory is the ability to store and manipulate information on a short-term basis in order to carry out a task. For example, when computing $1234 + 5678$ in our heads, most people will move right to left, adding 4 and 8 to get 12, then storing the 2 and carrying the 1 over into the next computation. Those intermediate results need to be maintained until the whole computation is complete, and this is the task of working memory.

Common tests of working memory are memory span and n-back tasks. In memory span, subjects are presented with sequences of numbers of different length, and then quickly probed in some way to determine whether they have remembered the sequence correctly (e.g. by presenting another sequence and asking whether it is the same or different). Subjects who can remember longer sequences correctly are said to have greater working memory capacity. This task may be complicated by requiring the subject to simultaneously perform some other task, such as mental arithmetic; this increases the emphasis on processing, rather than pure capacity (Turner, 1989). In the n-back task, subjects are presented with a continuous stream of digits, and they must indicate when the current digit is the same as the one presented n digits ago. This forces the subjects to retain and constantly update the previous n digits, demonstrating both the storage and processing component of working memory.

Kyllonen and Christal (1990) carried out a large ($n = 2144$) study investigating the relationship between working memory and reasoning ability. They found a remarkably strong correlation ($r > 0.8$) between the two factors, and suggested that complex reasoning was strongly mediated by working memory capacity. In other words, almost all complex reasoning tasks depend upon this underlying ability to store and manipulate information, and individuals' ability to carry out the latter is what ultimately determines their performance on the former.

Subsequent studies employing more advanced methodologies (Süßet al., 2002) have continued to find a strong correlation, although typically closer to $r = 0.6$ (Conway et al., 2003). These studies have also emphasized the complex nature of working memory. In particular, it is more than just capacity (e.g. remembering a telephone number), it involves the active processing of what is remembered in order to accomplish a goal. This raises the question of whether we should forget about the memory component and focus on these executive processes instead. In an fMRI study, Smith and Jonides (1999) found different brain regions responsible for storage versus processing of working memory, emphasizing the separation between these roles. Unsworth and Engle (2005) investigated this further and found that pure working memory capacity had no relation to problem difficulty in the RPM; subjects with high working memory capacity did not have a particular advantage over low capacity subjects as problems became more difficult. However, we know that working memory ability in general *is* important to the RPM (Carpenter et al., 1990). In combination, these results suggest that it is the executive aspect—the ability to manipulate the contents of working memory—that must be responsible for the correlation between working memory and general intelligence. However, these studies do not disagree with the claim that working memory, defined as the general capacity to store and manipulate information on a short term basis, is crucial to general intelligence, they simply differ on where the emphasis should be placed.

Processing speed

Processing speed is the idea that intelligence is ultimately dependent on a general quickness of mental activity. The idea is that if subjects can process information quicker, more rapidly pursue ideas, and quickly share information across the brain, then that quickness will appear as a higher general intelligence. This ability is usually measured using very simple reaction time trials, such as presenting two shapes and asking the subject to determine whether they are the same or different. The idea is to eliminate any higher level cognitive processing and get at the speed of these fundamental components.

These tasks show a consistent correlation with tests of intelligence (Kail, 2000). Proponents also argue that processing speed is the underlying factor responsible for the observed correlations between working memory and intelligence—faster processing leads to enhanced manipulation of the contents of working memory, which in turn leads to the gains in general intelligence (Fry and Hale, 1996). Thus the work discussed above showing a relationship between working memory and intelligence can be interpreted instead as demonstrating the importance of processing speed. However, it is difficult to establish these causal claims; it could also reasonably be said that improved working memory allows for faster processing.

It is not clear what mechanism in the brain underlies this processing speed. One theory is that it is due to increased myelinization, which is a process that occurs in the brain

wherein the axons of neurons become sheathed in a white fatty material (hence "white matter"). This material promotes the conductance of action potentials along the axon; increased proportion of myelinated to unmyelinated axons could lead to a general increase in speed of neural communication, and therefore faster processing speed. In support of this idea is the parallel development of processing speed, intelligence, and myelinization in children—as one increases, so too do the others (Fry and Hale, 1996). We could also hypothesize a relationship between processing speed and the small world architecture described in Section 2.1.1; perhaps those with a more efficient network architecture are able to process information more quickly.

Attention

This theory proposes that it is the ability to attend to the correct input or choice amongst competing data that determines intelligence. This is closely related to the executive functions of working memory, in that often the role of this attention is to decide what information should be stored in or output from memory. However, the emphasis here is not just on input and output, but on input and output in the face of interference. Phrased differently, it is the ability to ignore information that is not relevant to the current task.

The Stroop task is commonly used to assess this selective attention ability. In this task a word is quickly flashed on the screen, and subjects must identify the colour of the font. Normally subjects can do this without problem, but if the word is actually the name of a conflicting colour (e.g. the word "blue" in green font) then subjects have much more difficulty, taking longer to respond and making more errors (Stroop, 1935). This is interpreted as a problem of attention: subjects are failing to attend to the colour of the word because they are being distracted by its content. A significant negative correlation has been found between intelligence and the Stroop effect; in other words, subjects that are less affected by this interference also have higher measured intelligence (Matzel and Kolata, 2010). This supports the idea that selective attention is important to intelligence.

Gray et al. (2003) investigated this effect using a modified n-back task, where digits were chosen to be intentionally confusing. For example, in a 3-back task the sequence 5 6 4 3 5 is tricky because the second 5 is *almost* three digits away from the previous 5. This increases the attentional demands of this working memory task, because the subject must ignore the second 5 even though it may be sending out strong recognition signals. Gray et al. found that this attention-heavy version of n-back better differentiated high and low intelligence individuals than the standard version, again demonstrating that there is an important connection between the two factors.

2.1.3 Summary

These results demonstrate the complex nature of intelligence, and the associated difficulty in finding the link between intelligence and the brain. There is good support for the claims that brain size, grey matter, white matter, network properties, processing speed, working memory, and attention are all making significant contributions to intelligence. This is further complicated by the interdependent nature of these claims: executive functions may be related to attention, processing speed to white matter, and so on.

The human brain is not a neatly organized system, and so it is unlikely that any single component underlies intelligence. We embrace this complexity and take it as an abundance of data which can be used to help build and evaluate any model of intelligent behaviour. We will return to this discussion in Chapter 4 to examine how the components of the model relate to these factors, and in Chapter 5 we will see how we can model individual differences in brain size, executive function, and attention.

2.2 Why do brains differ?

In the previous sections we discussed *how* brains differ, now we will address *why* they differ—where do those differences come from? There is a large amount of work in this field, but we will only cover the broad trends here.

2.2.1 Brain development

Brains change dramatically over our lifetime. From birth to around age six they undergo incredibly rapid growth, so much so that it is difficult to quantify the changes due to their instability. By age 6-8 the brain has largely settled, although it will continue to grow, adding volume and neurons, until age 20. After age 20 the brain enters a steady decline, losing about 10% of its neurons (85000 per day) and 12% of its volume by age 90 (Pakkenberg and Gundersen, 1997).[1] If we look at different aspects of the brain in more detail, we see that grey matter peaks around age 4 and then declines steadily, while white matter increases until age 20 and then remains relatively stable (Pfefferbaum et al., 1994).

What do these neuroanatomical changes mean in terms of cognitive performance? Staff et al. (2006) investigated the relationship between changes in volume and changes in g,

[1] Humans do continue to generate new neurons in certain brain areas as adults (Gould, 2007). However, the rate of this neurogenesis is quite low; the dominant trend is still an overall decrease. It is unknown what effect neurogenesis has functionally, although its predominance in the hippocampus has suggested a role in memory/learning (Zhao et al., 2008).

and found, as we might expect, that larger decreases in volume were associated with larger decreases in g. Interestingly, this effect was found predominantly in the white matter, even though, as just mentioned, white matter volume is relatively steady compared to grey matter. One possible explanation of this seeming contradiction is that the decreases in grey matter are a more normal part of cognitive development and do not significantly impact cognition, whereas the white matter, which is supposed to remain stable, has a more dramatic effect on performance when this stability is disturbed.

When looking in more detail at these changes in intelligence, researchers have found that verbal, crystallized ability remains unchanged or even increases with age; it is only the dynamic, fluid ability that decreases and causes the overall decrease in measured intelligence (Kaufman et al., 1989). The neural mechanisms of these changes are not well understood. It has been found that frontal brain areas, usually associated with fluid reasoning, decrease in volume more with age while temporal areas, usually associated with crystallized reasoning, are relatively unchanged (Resnick et al., 2003), but these are very general claims and no specific analysis has been done comparing localized brain changes to different patterns of intelligence.

One final point is that although there are significant changes in neurophysiology and intelligence, relative intellectual performance is very well preserved with age. In other words, subjects that are clever at age 20 will still be clever at age 60 (assuming no adverse pathological effects); they may not be as clever as they were 40 years ago, but relative to their age group they will perform about the same. Deary et al. (2000) illustrated this stability across a 66 year period, comparing scores of 101 subjects on a standard intelligence test at age 11 with scores on the same test at age 77. Although the average score increased significantly (at age 11 general intelligence is still on the upswing), the relative rankings were highly correlated ($r = 0.73$).

2.2.2 Genetic influences

What causes these changes in intelligence? This is a complex question, and closely tied to one of the most debated questions of the human mind: nature or nurture. In other words, are these changes preprogrammed in our genetic code, or are they a result of the environment we live in? The answer is certainly both, and we will now explore some of the major influences in either camp.

The influences of genetics on intelligence are difficult to determine, as they are difficult to separate from environmental influences. For example, when Galton (1892) was attempting to determine whether childrens' intelligence was inherited from their parents, he measured the children's intelligence, measured the parents', found a correlation, and concluded that intelligence was inherited. However, the obvious problem is that the parents raised the children in a particular way, which very likely influenced the course of their

cognitive development. Thus the observed correlation could have nothing at all to do with shared genes, and everything to do with the fact that parents create for their children an environment that reflects their own intellectual tendencies.

Because of these difficulties, modern heritability research has focused on twin and adoption studies. These represent natural experiments manipulating the degree of shared environment/DNA, so by testing the correlation between the intelligence of these pairs researchers can estimate the importance of the two factors. For example, two adopted children raised in the same family share a very similar environment, but dissimilar DNA—if their intelligence scores are highly correlated, then that indicates that environment is the dominant factor in determining intelligence. On the other hand, monozygotic twins raised apart share 100% of their DNA, but have different environments; correlations between their intelligence scores reveal the influence of DNA on intelligence. Dizygotic twins or monozygotic twins raised in the same family can be used to provide intermediary measurements. These techniques are not perfect, as they assume that environment and genetics are independent when they almost certainly are not, but they are the best method of studying these relationships.

The results of these studies largely confirm Galton's results: general intelligence is highly influenced by genetics, with correlations ranging between 0.5 and 0.9 (Deary et al., 2010). In addition, the influence of genetics increases with age: twins will show increasing correlation between intelligence scores as they get older (Bartels et al., 2002), and adopted siblings will become increasingly dissimilar with age (Gray and Thompson, 2004). This is despite the fact that humans are constantly accumulating more environmental factors over time.

These conclusions are in line with the characteristics of neural development discussed in the previous section. The stability of test scores over many years, in the face of what we must assume are significant environmental differences, suggests a strong genetic component to intelligence. In addition, the increasing influence of genetics corresponds with the stabilization of intelligence scores in children as they mature.

Finally, the genetic influences on test scores are also reflected in genetic influences on brain structure. Using similar studies, researchers can measure the neuroanatomical similarity between twins or adopted siblings, and thereby estimate how much of the brain's structure is under genetic control. These results are largely in line with the intelligence research, indicating that the brain is highly influenced by genetics. In addition, this influence is strongest in the very brain areas thought to be associated with higher level thought or intelligence (Thompson et al., 2001).

2.2.3 Environmental influences

The heritability research indicates fairly convincingly that intelligence is strongly determined by genetics. However, there are also significant environmental influences whose effects cannot be ignored. For example, the Flynn effect is a commonly observed trend, across a broad range of cultures, wherein intelligence scores have been steadily rising ever since reliable tests were first developed. The magnitude of these gains is usually around 3 IQ points per decade, although in one study over a 30 year period using a Raven's type test gains were as high as 7 points per decade (Flynn, 2009). The causes of these gains are unknown, but it seems unlikely that such rapid and widespread changes could be attributed to evolutionary processes; therefore, we must conclude that environmental effects are having a significant impact on intelligence.

There are a number of links between environment and intelligence. One of the strongest is with socioeconomic status: children of more privileged parents tend to score higher than children in less privileged families (Neisser et al., 1996). This is true both within a given culture, such as the United States, and between cultures, such as between wealthy and developing nations. These results tend to raise controversy, because they are incorrectly assumed to imply inherent differences between these groups (i.e. that wealthy people are inherently smarter than poor people). However, it is worth noting that the variation in intelligence within a given group is greater than the variation between them (Gray and Thompson, 2004); in other words, there are more important factors determining intelligence than group differences.[2] In addition, it is unlikely that socioeconomic status itself is causative—having more money and opportunities does not, in itself, make someone smarter. Rather, it is what one can do with that money and opportunity that is producing an effect.

One obvious such factor is education. Since schooling is meant to enrich young minds, it would be somewhat disappointing to find that it had no impact on intelligence. Fortunately, this is not the case—years of education is significantly correlated with intelligence score (Kaufman et al., 1989). Estimates of the effect place it around 2.5 IQ points per year of education (Winship and Korenman, 1997) . One obvious concern is that the causal effect has been reversed here; what if people that are smarter stay in school longer, rather than the other way around? However, in the above results this problem was minimized by controlling for the measured IQ of subjects before schooling. In other words, among subjects that started out with the same IQ, those with more schooling ended up with higher intelligence. This is not perfect, as it is possible that preexisting differences had not yet become apparent at that age, but it does lend weight to the theory that education can lead to an increase in intelligence.

[2]This is true even if we assume the group differences to be genetically based: the genetic variability within a group is more significant than the variability between groups.

Another important factor is diet. It is well known that a poor diet has consequences for normal development, resulting in decreased height and other negative effects; it is no great surprise that these effects extend into the cognitive domain. Malnutrition can result in mental retardation, as well as more subtle decreases in intelligence (Kretchmer et al., 1996). What is more debated is whether, in relatively well-nourished populations, more subtle dietary adjustments can improve intelligence scores. It seems that such effects, if they exist, are very minor. In a review of nutrition and intelligence information in several affluent countries over the past century, Flynn (2009) found that there was little relation between the two variables: intelligence continued to rise even when nutrition fell (such as during a depression), and the greatest gains in intelligence were found in the wealthier classes (who were already well-fed to begin with) or were equal regardless of different economic status (and the assumed differences in nutrition).

The broadest environmental factor is the idea of generally "stimulating" environments. This is the idea that individuals living in cognitively demanding environments respond by developing greater cognitive ability, much as someone living in a physically demanding environment develops stronger muscles. This is suggested by Flynn (2009) to be the leading cause of the Flynn effect—the increasing flow of information and knowledge in the modern age has led to increasing general intelligence. It also may be behind the decreased discrepancy in intelligence scores between urban and rural populations: modern travel, communication, and the spread of technology have reduced the differences between the two environments (Neisser et al., 1996). More concrete evidence of this factor can be found in several studies in which children were placed in specially designed enriched environments, and showed significant and longterm increases in measured intelligence (Kretchmer et al., 1996).

2.2.4 Summary

Thus both nature and nurture make important contributions to individual differences in intelligence. In this section we have not come much closer to understanding the causal relationships, we have simply pointed out the many factors involved. These factors themselves interact in complex ways: wealthy individuals have more opportunity to attend school for longer, well-nourished children get more out of school, and so on. In addition, there are complicated relationships between genes and environment that are only beginning to be understood. For example, certain genes may only become expressed in well nourished individuals, or there may be genetic predispositions that cause an individual to gravitate toward intellectually stimulating environments. In a study comparing heritability of intelligence and socioeconomic class, Turkheimer et al. (2003) found that there was very little genetic influence on the intelligence of subjects in low socioeconomic classes, while the variability in IQ of high socioeconomic class subjects was almost entirely attributable to

their genes. This emphasizes the close interaction between nature and nurture; subjects' scores can be improved up to a point by removing negative environmental influences, but once those needs have been met genetics are the dominant factor.

In summary, there is no clear answer to the question of why some brains are smarter than others. There are both general and specific differences in brain structure and function, and many different causes that may contribute to those differences. There are two important lessons to take from this chapter: first, intelligence arises in the brain, and the complexities of human cognition cannot be understood without understanding their basis in the biology; second, no cognitive factor operates in isolation, their effects combine in complex ways to produce what we call general intelligence. These lessons emphasize the importance of developing realistic neural models of cognition; such models are uniquely able to reflect the underlying dynamics of neural behaviour as well as the complex interactions between high level cognitive components.

Chapter 3

Raven's Progressive Matrices

The RPM is the most frequently used single test of general intelligence (Van De Vijver, 1997), second only to the Wechsler Intelligence Scales, which are actually batteries of tests including a Raven's-type matrix reasoning task. The popularity of the test is due to its minimal cultural influence, ease of administration, and high correlation with g. In an analysis of a broad range of intelligence tasks, the RPM was found to be the most highly correlated with g, and is in fact used as a benchmark to evaluate the g-loading of other tests (Marshalek et al., 1983). Even more so than g, the RPM is an excellent indicator of fluid intelligence. This is demonstrated in Figure 3.1, which graphically depicts the correlations between a broad range of intelligence tests. Tests closer to the centre are more highly correlated with g, and tests closer together are more highly correlated with each other. The RPM can be seen to be both highly correlated with g and associated with other measures of fluid rather than crystallized intelligence. In both psychometrics and neuroimaging, the RPM is the standard for assessing g_f (Perfetti et al., 2009; Kane and Engle, 2002; Gray et al., 2003).

Thus the RPM has widespread use, a long history of internal and external evaluation used to verify and improve its validity, and a close association with the underlying mechanisms of general and fluid intelligence. For these reasons it makes an excellent vehicle for our investigations.

3.1 Examples

The easiest way to understand the RPM is by looking at some examples, and for this reason we have created some Raven's-like matrices, seen in Figures 3.2 and 3.3. We present these rather than the true Raven's matrices, as the dissemination of the real items could have negative consequences for the test's validity.

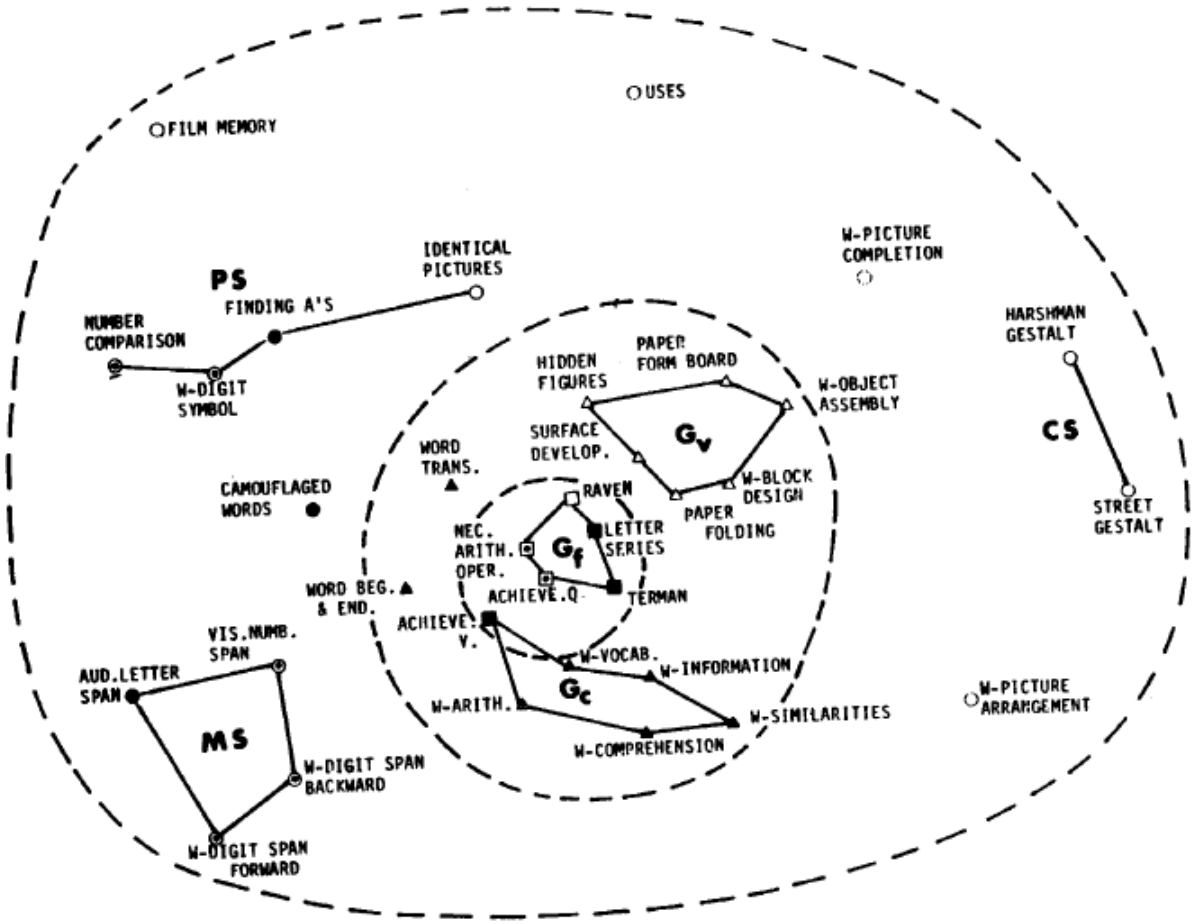

Figure 3.1: Various tests of intelligence, grouped according to their correlations with g and each other. Obtained from Marshalek et al. (1983).

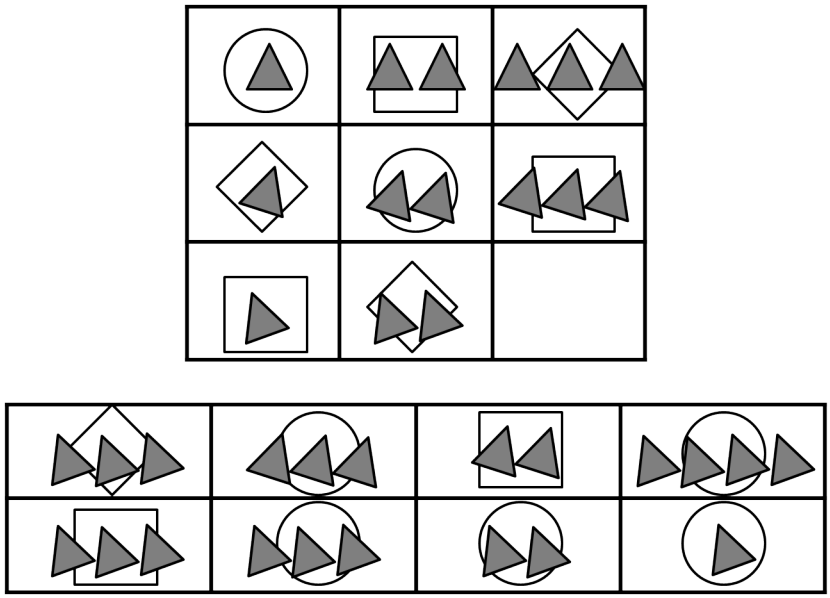

Figure 3.2: A simple Raven's-style matrix

Figure 3.2 is an example of a simple item in Raven's Advanced Progressive Matrices. The subject's task is to determine which of the eight answers along the bottom belong in the blank cell. Most subjects will be able to identify the correct answer (number 6) without difficulty. However, let us examine in more detail what cognitive steps the subject went through to arrive at that answer.[1] The subject needs to:

1. parse the image into its component parts; for example, they need to recognize that there is one big empty circle and one smaller shaded triangle in the top left cell ($\text{cell}_{1,1}$).

2. guess which objects correspond to each other (i.e. which objects are they going to try to find a rule for). For example, in this matrix it is natural to think that the big, empty shapes correspond to each other, and the small, shaded triangles correspond to each other.

3. find a rule for the corresponding items; if they were looking at the shaded triangles, they might notice that the number of triangles is increasing by one across each row.

[1] Of course not all subjects solve a given matrix in the same way; we try to refer to the general information that needs to be extracted/manipulated, rather than the specific method of accomplishing those goals. Any subject that correctly solves the matrix (and not through chance) will need to extract at least equivalent information to what we describe here.

4. use the rules to select an answer—for example, they might generate a hypothesis as to what the answer should look like based on their rules, and then choose the answer that is closest to their hypothesis.

Note that this is almost certainly not a serial process as it has been presented here. If the subject gets to step 3 and then cannot find a good rule, they may go back to step 2 and try to find a different correspondence. They may think they have found a good rule, but then when they go to select an answer they cannot find any that match their hypothesis, forcing them to go back and reevaluate their rules. They may even need to go all the way back to step 1 and reparse the image, for example as lines rather shapes. Thus there is a back and forth movement, with information from each step being used both forwards and backwards. In addition, many of these steps can be occurring at a subconscious level. The visual processing in particular can come so naturally that subjects do not even notice themselves doing it, and if they do it is not as well articulated as has been laid out here.

This process is demonstrated in Figure 3.3. In this case it will likely take several tries before the subject discovers that the image needs to be parsed into lines, and that the top lines (above the horizontal bar) correspond to each other and the bottom lines correspond to each other. Only then can the subject discover the two rules: along the bottom, a line is present in the third cell of a row if it was present in the first and the second cell; along the top, a line is present in the third cell if it was present in the first or second but not both.

These examples also demonstrate why the RPM is such a popular intelligence test. First, it requires very little instruction or background, rather it engages the automatic impulse to complete the pattern by filling in the blank cell. It also does not require any special knowledge in order to solve; rules such as "number of triangles is increasing by one" represent fairly simple principles, making it unlikely that a particular educational or cultural background would be needed in order to discover them. And finally, although the RPM has a very simple format it is able to assess a broad range of dynamic problem solving ability. This simplicity in form and underlying principles combined with complexity of reasoning is what makes for a good test of general intelligence.

3.2 Why are some problems harder than others?

The problems in the RPM are arranged in increasing difficulty, with easy items at the beginning and difficult items at the end. This ordering is determined experimentally, by looking at human error rates on each item. The order needs to be determined in this way, because it is not completely understood *why* some problems are hard and others are easy. Recall that these matrices are the tools used to test general intelligence; if we understood

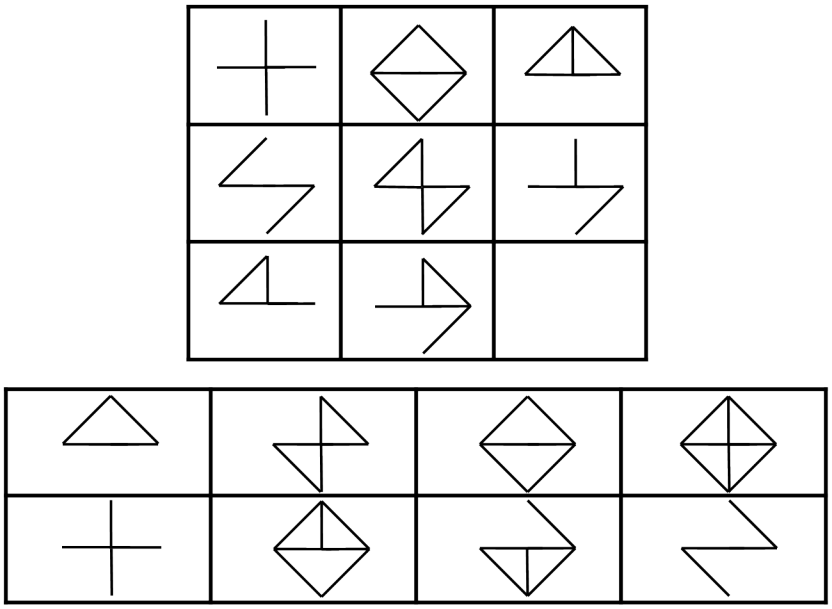

Figure 3.3: A more difficult Raven's-style matrix

exactly how they did that, that would imply that we also understood general intelligence (which we obviously do not). However, the converse is also true: by analyzing what makes some matrices more difficult than others, we can gain insight into the abilities underlying general intelligence.

The most obvious factor is the number of rules. Problems that require six rules in order to solve are going to be more difficult than problems that require only one rule. There are a number of theories as to the specific mechanism underlying this difficulty. Carpenter et al. (1990) associate it with working memory: with more rules involved, there is more information that needs to be juggled in the mind. Christoff et al. (2001) place more emphasis on attention/executive control, suggesting that with more rules there is more simultaneous processing going on. This is a little problematic, because most evidence points to the fact that rules are processed serially rather than in parallel, but it does seem reasonable that there would be some added difficulty involved in sorting out the simultaneous input.

Another factor in problem difficulty is the type of rule that needs to be found. Not all rules are equally easy to determine: "number of triangles increases by one" is relatively easy, while "each shape can be a triangle, a circle, or no shape at all" is more difficult to discover. This was first suggested by Carpenter et al. (1990), and has since been suggested in different forms by Mackintosh and Bennett (2005) and DeShon et al. (1995) (different rule types,

but the same underlying claim that rule type helps determine problem difficulty). In combination with the previous point, this is essentially a quantity/quality distinction; intelligence is taxed by both the number of computations that need to be carried out and the individual difficulty of those computations.

A more low-level factor is the visual complexity of the matrix objects. This was put forward by Meo et al. (2007), who found that error rates go down if the matrix is made up of easy to identify or non-overlapping elements, even if the number and complexity of rules on those elements are exactly the same. These manipulations affect the difficulty of the initial visual parsing (step 1, see Section 3.1), as well as correspondence finding (step 2) since familiar items may contain innate correspondence cues. Similar work was done by Primi (2002), who also found that increasing the visual complexity of matrix items increased their difficulty. Primi also directly manipulated the difficulty of correspondence finding by adding misleading visual cues that would cause subjects to guess the wrong correspondences, and again found that this increased the difficulty of the matrices. These results demonstrate that we should not neglect the importance of these low-level perceptual processes in high-level reasoning tasks.

A fourth difficulty factor is suggested by the work of Kirby (1983). He found that by changing the order of items in the test he could manipulate error rates. That is, the difficulty of an item is dependent not only on the item itself, but on which items have been seen previously. For example, if the subject encounters an item that requires a completely different rule type than what they have seen before, they are less likely to find the correct rule. If they recently solved a matrix using a similar rule, they are more likely to find the correct answer. Similar results were found by Verguts and De Boeck (2002). These studies demonstrate another important aspect of intelligence: the ability to generalize and learn from results so that previous computations can be applied to the current problem.

3.3 Why do some people score better than others?

Those subjects that score the highest are those that make the fewest mistakes; therefore, there is a close relationship between factors affecting problem difficulty and factors affecting individual performance—those who score better will be those who are better able to deal with the sources of difficulty.

Unsurprisingly then, there are strong correlations between working memory and RPM performance (Kyllonen and Christal, 1990). Carpenter et al. (1990) found that RPM performance correlated with the ability to manage more subgoals simultaneously in working memory. Applied to the RPM, this suggests that subjects with better working memory will be better able to deal with increased numbers of rules. Neurophysiological studies have also found that during the RPM many brain areas thought to be associated with working

memory become active (Prabhakaran et al., 1997). However, when looking specifically at working memory capacity, Unsworth and Engle (2005) found no correlation between capacity and problem difficulty or number of rules. This suggests that the working memory correlations have more to do with the executive processes—the ability to manipulate the contents of working memory—rather than the number of items that can be stored. Nevertheless, there seems to be strong evidence that working memory ability in general is important to individual performance on the RPM.

Another important aspect of RPM performance is the ability to learn. Verguts and De Boeck (2002) demonstrated that subjects do learn during the course of the test; when manipulating the order the items were presented in, they found that subjects were more likely to try to solve an item using strategies that had been successful on previous items. Therefore, subjects that are able to more quickly pick up on these trends and learn the underlying principles of the RPM should have enhanced performance. This was demonstrated by Williams and Pearlberg (2006) and Tamez et al. (2008), who investigated the relationship between the RPM and a three-term contingency learning task, where subjects had to learn arbitrary relationships of the form $A + B = C$. Both studies found that RPM performance correlated with the ability to learn these contingencies, suggesting that those who were able to pick up on the contingencies more quickly were also able to benefit more from learning during the course of the RPM.

Considering that visual complexity was an important part of matrix difficulty, we would expect that subjects with enhanced visual processing would demonstrate superior performance. Interesting data in this area can be found in the study of autistics, since a common effect of autism is enhanced visual performance. Dawson et al. (2007) found that although autistics tend to perform below average on more verbal intelligence tests, they perform above average on the RPM. Soulières et al. (2009) provided support for these results in an fMRI study, finding that controls and autistics employed the same basic network, but autistics had relatively enhanced activation in extrastriate (visual processing) areas, and relatively decreased activation in frontal regions. This suggests that autistics are not approaching the problem in a completely different way, they simply have different capacities; therefore we would expect similar differences between normal subjects with high versus low visual processing ability. However, there have not been any studies testing this hypothesis, and so we can only extrapolate from the studies of autistic reasoning.

One aspect of individual performance that is not apparent in the matrix items themselves is the strategy used by the subject to solve the problem. Using eye tracking and verbalization, Vigneau et al. (2006) demonstrated that there were two distinct strategies used by subjects. The first they call "response construction"; subjects examine the matrix, come up with a hypothesis, and only then look at the answers to find a match for their hypothesis. The second strategy is "response elimination"; rather than generating a complete response, subjects look back and forth between matrix and answers, using a

process of elimination to narrow down the 8 possibilities until only one remains. Vigneau et al. also found a strong correlation between strategy used and overall score: those who used response construction rather than elimination tended to score higher. The same strategies and contrasting performance have been found on other tests that share a similar format (Bethell-Fox et al., 1984; Haverty, 2000). It is not clear at the moment which is the causative factor: do less intelligent (i.e. lower scoring) subjects resort to the response elimination strategy because it is easier, or does a preexisting preference for the worse strategy (response elimination) cause a lower score? In light of the individual differences described above, it seems that the former is more likely. Subjects come in to the test with differing cognitive ability, and it seems reasonable that this would motivate their choice of strategy. For example, the response construction strategy places greater demands on working memory, because the subject must hold the complete hypothesis in mind before moving to the answers. Therefore, subjects with lower working memory ability will not be able to carry out the response construction strategy, and will be forced into response elimination. It should be possible to investigate this further by systematically changing these variables, such as working memory load, and seeing if that causes subjects' strategies to change—an interesting avenue for future work.

In summary, just as there are many factors contributing to problem difficulty, there are many factors contributing to individual performance. These factors range from the low level, such as working memory and visual processing, to the high level, such as problem solving strategy. These results provide us with an abundance of data against which to compare the performance of our model. The better a job we have done of modelling human cognition, the more of these results we should be able to account for. We make these comparisons in Section 5.2.

3.4 Previous models

Despite the importance of the RPM both in itself and as a window into general intelligence, the abundance of experimental research contrasts sharply with the lack of theoretical results. This is less surprising when we consider the general theme of the discussion so far, which is complexity. The abilities investigated here are interesting because of their complicated causes and effects, but that necessarily comes tied to a lack of clarity and consensus as to how they might be bound together into an overall framework. Nevertheless, there has been some interesting work in this area, which we will summarize here.

In all of these cases we present a very simplified description of the inner workings of the model. Rather than the details of their implementation, we wish to focus on the theoretical motivations and consequences of these models.

3.4.1 Evans

Having said that there is little theoretical modelling work in this area, there is a related problem in Artificial Intelligence referred to as analogical reasoning. Broadly speaking, this consists of investigating how to automate the solution of "a is to b as c is to ?". In other words, how is a similar to b? What is the transformation between them? And what will the result be of applying that transformation to c? This is an important problem in AI, because it allows for generalization. For example, suppose we have created a visual recognition/natural language processing system that knows all about cubes—it can recognize them, move them around on command, and state facts about them ("the blue cube is on top of the green cube"). Now if we can only make the program understand the analogy between cubes and pyramids, we can have a system that does all the same things for pyramids without having to redevelop those capabilities in the new domain.

This is essentially the same problem we are trying to solve in the RPM. We want to know what the transformations are between the cells of the top two rows, and then what the result will be of applying those transformations to the third row: $\text{cell}_{1,1}$ and $\text{cell}_{1,2}$ are to $\text{cell}_{1,3}$ as $\text{cell}_{3,1}$ and $\text{cell}_{3,2}$ are to ?. However, the focus of the two domains are different: the AI analogy program tends to see analogy largely as a search problem, the challenge being to find the path from a to b or c, whereas we are interested in the process itself and how it relates to general intelligence. Nevertheless there are interesting overlapping ideas, in particular in order to understand the history of the current research.

We will focus on the ANALOGY model, developed by Evans (1968), as its emphasis on geometric reasoning makes it the most direct ancestor of later models of the RPM.[2] Evans was interested in solving geometric analogy problems, which can essentially be thought of as a 2x2 matrix problem where a=$\text{cell}_{1,1}$, b=$\text{cell}_{1,2}$, c=$\text{cell}_{2,1}$, and the subject is given 5 possible answers to fill in the blank cell. However, in the problems he was interested in the only rules used were basic geometric transformations (translation, scaling, reflection, and so on); there were no abstract rules such as are present in the RPM.

Evans' solution uses the same basic approach as all the models we will discuss in this section. The first step is to transform the model from images into a format that can be processed by the algorithm. In Evans' case this meant hand-coding descriptions of each figure as a set of attributes (e.g. the type of shape, its location, its endpoints). He also hand-built a library of transformations. The algorithm then went through all the possible transformations (rules) between the representation of a and the representation of b, finding the ones that apply. For example, if a contains a square at (1,1) and b contains a square at (1.5,1) then the algorithm would recognize that this met the criteria of the "translate to the right" rule. Once this is complete the model has a set of transformations that map

[2]See Hall 1989 for a broader review of AI approaches to analogy.

a to b. To pick an answer, it calculates similar sets for c and each of the possible answers, and then picks the answer whose transformation set is most similar.

Evans' approach was sound, and indeed all future models (including our own) have followed this basic idea of breaking the cells down into attributes and then determining the rules that map those attributes on to each other. However, the main criticism of this model, and indeed the criticism which will be levelled against all subsequent models, is that one can quite reasonably ask whether the algorithm is solving the matrix, or whether it was solved already by the person setting up the problem and specifying the rules. It would be analogous to giving someone a matrix and saying "each of the triangles in these cells can grow or shrink in size, move left or right, or disappear; figure out which of those is being applied in this matrix". There is some problem solving element involved, but it is certainly not the same problem as subjects are solving during the RPM. In addition, Evans makes no effort to map his algorithm on to human cognitive processes. This is not a criticism, so much as pointing out the difference in the AI approach; his is a computational solution, and tells us very little—nor is it intended to—about general intelligence.

3.4.2 Hunt

Hunt (1973) was the first to take these analogy techniques and apply them to the RPM. His approach was essentially the same: break the items down into sets of features, and then look through a library of known rules to find those that explain the observed differences between adjacent cells in a row. However, the addition of the more complex rules found in the RPM as compared to Evans' geometry problems forced Hunt to realize that not all matrices could be solved in the same way: those that involved abstract rules, such as "each row contains a square, triangle, and circle" require different input processing and computations than the visual rules, such as the geometric transformations Evans used. This insight, that there are qualitative differences in RPM problem solving, has been born out by more recent work.

Unlike Evans, Hunt was interested in how his model might provide insight into human cognitive processes. This forced him to recognize the main flaw in his model: it is only "as powerful as the operators available to it". In other words, everything depends on this library of preprogrammed rules. This is a problem in two different respects. Algorithmically, it is a problem for the same reason mentioned above; one must question whether the model is solving the matrix, or whether all the "solving" was done when coming up with that library of rules. The second problem was revealed due to Hunt's efforts to use this system as a model of how humans solve the problem. If RPM performance is ultimately dependent on this library of known rules, then the RPM is measuring crystallized intelligence—the ability to acquire and use knowledge. However, we know that the RPM is primarily a measure of dynamic, fluid ability (see Section 3). Hunt saw this as evidence

that the RPM was a flawed test that did not measure what it was intended to. However, all future work has continued to indicate that the RPM is an excellent indicator of g_f; therefore, it seems more reasonable to conclude that the problem is not with the RPM, but with Hunt's model. Human performance cannot be dependent on a library of known rules.

3.4.3 Carpenter, Just, and Shell

The work of Carpenter et al. (1990) has had an impact on almost all modern RPM research. Unlike the previous two models, Carpenter et al. were explicitly interested in understanding the human cognitive processes involved, and employed modelling as a means to that end. Thus their primary contribution to the understanding of the RPM was their analysis of rule types (see Section 3.2) and the relationship with working memory (see Section 3.3), but they also developed the most fully fledged cognitive model of the RPM.

Carpenter et al.'s model is based on a production system. Production systems are a popular technique in cognitive modelling, and involve a collection of if-then rules combined with a central memory area. The if-then rules (called productions) are triggered when certain conditions are met in that memory area, and can then operate on that memory (inputting, outputting, or manipulating). The modified memory can then trigger more rules, thereby accomplishing complex computations. This system naturally lends itself to the type of processing described in the previous two models. First, the description of the matrix (the same type of hand-coded attribute based representation) is input into memory. Productions are then triggered by predefined patterns in those attributes, and add the corresponding rules into memory.[3] Finally, once all the rules are in memory they can be used to select an answer.

The main advance in this model is its firmly cognitive basis. This allows Carpenter et al. to use experimental data to guide their model, and match its outputs to human outputs. For example, their eye-tracking data indicated that subjects began with a cell to cell comparison and used that to build up row-wise rules, so that is the strategy they built into their model. It also allowed them to test their predictions on individual performance. For example, they predicted that rule type was an important factor in problem difficulty and therefore individual performance, and they were able to show that by increasing or decreasing their model's library of known rules they could increase or decrease its performance on the RPM. They also manipulated the number of rules that could be maintained in the model's memory (analogous to human working memory ability) and showed similar

[3]This brief description makes this task sound much simpler than it is. In reality it is a complicated system of looking for patterns between cells, then generalizing those patterns across rows, and then seeing if those generalize to the whole matrix.

effects on performance. These techniques, regardless of the specific results, demonstrate how modelling can be used to provide insight into cognitive processes.

There are three main disadvantages to this model. The first is, as always, the dependence on a library of known rules. The problems of this have already been discussed: it raises the question of whether the solution has been preprogrammed into the input, and it does not reflect the fluid ability we know is central to the RPM. This reliance also contributes to the second difficulty, which is an unrealistic inflexibility. Their model will always get the same problems right, and always make identical errors—in sharp contrast to human data (Bors, 2003). In addition, their ability to model individual differences involves adding or removing large chunks from the model's abilities, for example removing the knowledge of a certain rule or allowing the model to remember 3 rules instead of 4. This seems to be an unrealistic depiction of the smooth gradient of individual differences observed in humans. We see such differences in the crystallized domain, where one person can know things another does not, but in the fluid domain it is rare to find such a sharp divide where one person has an ability and the other does not. This is related to the third problem with the model, which is that it is fixed at a very high level. It is unclear how it maps onto lower level processes, such as the hardware of the human brain. This limits the model's scope: it can tell us about high level aspects of intelligence, but low level factors (such as those that might give rise to a more realistic range of individual differences) remain a mystery.

3.4.4 Lovett, Forbus, and Usher

The most recent work on modelling the RPM has been done by Lovett et al. (2007, 2010). They have taken a different approach, focusing on low-level visual processing rather than high-level reasoning. This is an area that has been largely neglected, primarily because researchers who choose to work on the RPM tend to be more interested in cognitive problems than vision problems. However, as discussed in Section 3.2 visual reasoning is a crucial aspect of the RPM, so it is good to see work being done in this area.

Lovett et al.'s solution is based on an image processing technique that can take an image, such as a cell of a Raven's matrix, and extract the structural information (e.g. that it consists of an open shape with an object inside of it). Essentially this is automating the translation between image based and attribute based representations, which has been done by hand in all other models. Their model then looks for structural similarities between these representations, allowing it to identify objects that have been rotated or scaled. In addition, the lack of such transformations can be used to infer more advanced transformations; for example, if there a circle in $\text{cell}_{1,1}$ that has no match in any of the other cells, and a square in $\text{cell}_{1,2}$ and triangle in $\text{cell}_{1,3}$ with the same problem, then they can guess that the circle, square, and triangle correspond to each other. Together, this allows the model to create

descriptions of the transformations (rules) linking each row. It can then compare these transformations between the first two rows to see if they represent general rules for the matrix. If it does find that it has a good rule, it then tries each of the eight possible answers to see which will complete the third row in such a way that its transformations fit with the general matrix transformation.

The main limitation of this model is that it is unclear how it maps onto human cognition. This is primarily a computational solution, used to demonstrate that these image processing and structure mapping techniques can be used to accomplish complex tasks. The structure mapping component is based on the work of Gentner (1983) and so is at least motivated by human cognition, but the image processing and the actual problem solving that moves from structured representation to answer selection are purely computational solutions. This limits the insight this model can provide into the higher level aspects of human intelligence; its contributions are primarily in the visual processing domain. In addition, Lovett et al.'s model is designed and tested on the Standard version of the RPM rather than the Advanced. Because the more difficult problems in the Advanced version tend to be more abstract and less based on visual transformations, it is possible that their model will run into difficulties. However, this is something that would be interesting to see investigated further.

Chapter 4

A new model of the RPM

4.1 Motivation

Our work fills the important gap outlined in the above research. Specifically, we investigate the inductive process of rule generation: how do subjects extract general rules from the information presented in a Raven's matrix, and what does that tell us about human cognition?

There are three primary motivations for this work. First, understanding rule generation is important to understanding the RPM. We discussed above how previous models have relied on an unexplained library of known rules, and why that causes problems for their explanatory power—namely, it disagrees with the experimental research showing that the RPM measures dynamic, fluid ability rather than crystallized knowledge. In our work we provide just such a dynamic method of rule generation, thereby enhancing our knowledge of the cognitive abilities underlying the RPM.

Second, this work provides insight into general intelligence. We know that fluid intelligence is a central component of general intelligence, and we also know that the RPM is an excellent measure of fluid intelligence. This tells us that whatever cognitive abilities are involved in the RPM are important to fluid intelligence. However, the central element of RPM performance, actually coming up with the rules that govern the matrix, has up until now been largely a mystery. Therefore by providing an account of those abilities, we gain important insight into the underlying components of general intelligence.

The third motivation for this research is to demonstrate the value of neural modelling. This approach allows us to provide a unified description from high level behaviour down to neural implementation, adjust the model's behaviour in a way that reflects human data, and map the results and predictions from this model onto the human brain. The success

and insight gained by this model, which will be examined in Chapter 5, demonstrate the advantages of this approach to understanding complex cognitive systems.

4.2 Scope

With these grand goals in sight, we must also keep in mind the limitations of our model. Recall in Section 3.1 we described 4 broad components to the RPM task: visual parsing, correspondence finding, rule generation, and answer selection. In our work we have chosen to focus on the third component, rule generation. Answer selection is also modelled, but although still inspired and constrained by the biology, it is not modelled at the neural level. The visual parsing and correspondence finding, which we will capture under the general heading of visual processing, are not a part of our model—the input is converted by hand into a text-based format. This is not unique to our model, but rather represents the norm: Evans (1968), Hunt (1973), and Carpenter et al. (1990) all rely on hand-coded input.[1] Carpenter et al. downplay this component, stating that visual processes "are not a primary source of individual differences". This is not the tack we wish to take. As discussed in Section 3.2, we believe that visual processing is an important part of RPM performance. The vision system is not simply input to the cognition system, rather they have complex interactions with one another. Even state-of-the-art computer vision systems, making no attempt to be biologically plausible, would be hard-pressed to extract the necessary information from a Raven's matrix, not to mention the added complexity of integrating that processing with a problem solving system. Thus we exclude visual processing from our model not because it is unimportant, but because the complexities involved would prevent us from making any progress in understanding rule generation, the component we are most interested in. However, we acknowledge that this would be an interesting direction for future research, and an important addition to our understanding of the RPM; the work of Lovett et al. (2010) represents an interesting step in that direction.

4.3 Background methods

There are two theoretical frameworks which form the basis of our work. The first is Vector Symbolic Architectures (VSAs), which we use to encode the matrix information in a structured, mathematical format. The second is the Neural Engineering Framework (NEF), which we use to translate those mathematical forms into neural representations. We will discuss these techniques briefly here (see Gayler 2003/Plate 2003 and Eliasmith and Anderson 2003 for more details, respectively).

[1] Even the Lovett et al. (2010) model, which is focused on automating the visual processing, requires the user to manually segment the image into its basic elements.

4.3.1 Vector Symbolic Architectures

VSA is a label for a family of techniques that use high dimensional vectors to represent structured, symbolic information. For example, in the sentence "the dog chases the ball" there are three symbols: "dog", "ball", and "chasing". However, a sentence is more than just a collection of words, it has structure—"the dog chases the ball" is very different from "the ball chases the dog". Thus if we want to translate this information into a mathematical format we need to preserve not only what elements are present, but how they are related to each other. This is what VSAs allow us to do.

The first component of representation is vocabulary: how to represent "dog", "ball", "chase", and so on. We do this by associating a random high-dimensional vector with each word. For example, "dog" will be $\begin{bmatrix}0.3 & 0.4 & 0.1 & \dots\end{bmatrix}$, "cat" will be $\begin{bmatrix}0.2 & 0.6 & 0.4 & \dots\end{bmatrix}$ and "chase" will be $\begin{bmatrix}0.7 & 0.2 & 0.1 & \dots\end{bmatrix}$. If we represent each word as a 1-dimensional vector between 0 and 1 with a precision of 0.1, then we can store a maximum of 11 words in the vocabulary ($\begin{bmatrix}0.0\end{bmatrix}$, $\begin{bmatrix}0.1\end{bmatrix}$, $\dots$ $\begin{bmatrix}1.0\end{bmatrix}$) before we run out of unique vectors. By adding another dimension we increase the number of unique identifiers to 121. However, in reality we will have far fewer useful identifiers, because the vectors need to be sufficiently far apart that they can be distinguished in a noisy environment (where $\begin{bmatrix}0.5 & 0.5\end{bmatrix}$ will look the same as $\begin{bmatrix}0.5 & 0.6\end{bmatrix}$). Thus the number of words that can be stored in the vocabulary will be proportional to the dimension of the vectors and the noise and precision required in the particular task.

The next problem is how to combine these vectors together to represent structures. For this we require two operations: a superposition operation that combines vectors into a set, and a binding operation that ties two vectors together. The important aspect of the first operation is that it creates a new vector that is similar to each of its inputs (we define the similarity of two vectors as their inner product). The important aspect of the second operation is that it creates a new vector that is different from each of its inputs, but from which those original inputs can still be recovered. Different VSA implementations are defined by their choice of these operators.

We follow the Holographic Reduced Representation implementation (Plate, 2003) in using vector addition as our superposition operation and circular convolution as our binding operation.[2] Vector addition is simply the element-wise addition of the two vectors, which can be thought of as producing an average of the two. Circular convolution is more

[2]In true HRRs all vectors and operations on those vectors are normalized. However, sometimes in our model we relax this constraint in order to simplify the computations. This is not critical to the system's performance, given the noise inherent in neural vector representations.

complicated, defined as

$$\begin{aligned} C &= A \otimes B \\ &\text{where} \\ c_j &= \sum_{k=0}^{n-1} a_k b_{j-k \bmod n} \end{aligned} \tag{4.1}$$

Circular convolution can be thought of as the multiplication to superposition's addition, as it shares many of the same properties; it is commutative, associative, and distributive (Plate, 2003). Circular convolution meets our two criteria: it produces a vector which is not similar to A or B, yet we can also recover A or B from C. We do so using the idea of a pseudoinverse. The pseudoinverse of A, A', is defined as

$$a'_i = a_{-i \bmod n}$$

This has the useful property that $A \otimes A' \approx I$, where I is the identity vector. We can use this as follows:

$$\begin{aligned} C &= A \otimes B \\ C \otimes B' &= A \otimes B \otimes B' \\ &\approx A \otimes I \\ &\approx A \end{aligned} \tag{4.2}$$

In other words, if we have C containing A and B, and we want to recover A, then we convolve C with the inverse of B and the result will be approximately equal to A (and vice versa to recover B).

One final component of VSAs is cleanup memory. This is a standard technique used to counteract the information loss during VSA operations. The idea is that if we know the vocabulary of vectors used in the system and we have a noisy vector v, we can compare v to each of the vectors in the vocabulary and if it is similar to one of them output that clean version rather than v itself. For example, suppose we are looking at a note written by someone with messy handwriting. When we look at the first letter it may take us a bit of puzzling to figure out that it is a "T". But then when we move on to the next letter, we do not store that messy version of "T" in memory, we store the nice clean knowledge of "T" we have built up over time. This is analogous to what is going on in cleanup memory. Thus we can take the approximation of A produced in Equation 4.2 and pass it through cleanup memory to remove the approximation and end with the true representation of A.

These are the building blocks that allow us to encode structured information. For example, to encode "the dog chases the ball" we could create a vector $A = \textit{subject} \otimes \textit{dog} + \textit{object} \otimes \textit{ball} + \textit{verb} \otimes \textit{chase}$. This is different from "the ball chases the dog" because

$subject \otimes dog$ is different from $subject \otimes ball$, although they will be similar to some extent because they both involve chasing ($verb \otimes chase$). We can also encode hierarchical structure, such as combining an independent clause and a dependent clause in the phrase "the dog chases the ball while at the park". We might encode this as $C = ic \otimes A + dc \otimes B$ (where B represents "while at the park").

We can also extract information from this sentence. For example, if we had C and wanted to know the independent clause, we would calculate

$$\begin{aligned} C &= ic \otimes A + dc \otimes B \\ C \otimes ic' &= (ic \otimes A + dc \otimes B) \otimes ic' \\ &= ic \otimes A \otimes ic' + dc \otimes B \otimes ic' \\ &\approx A \otimes I + noise \\ &\approx A \end{aligned}$$

We can then pass A through cleanup memory in order to remove the noise and make A suitable for further computations. For example, we could determine the subject by calculating $A \otimes subject' \approx dog$, or we could determine what role "dog" held in the sentence by calculating $A \otimes dog' \approx subject$.

These examples illustrate how we can encode and decode structured information using high dimensional vectors. The details of how we translate from input (in these examples, English phrases) into vector form are not important—what we are interested in is that the two operations of binding and superposition allow us to combine information into new vectors while still preserving the information in and relationships between those vectors. We will discuss in Section 4.4 how we apply these techniques to the RPM.

4.3.2 Neural Engineering Framework

VSAs allow us to represent information in a structured mathematical format, but we also need a way to represent those vectors and carry out the VSA operations using neurons. This is the role played by the NEF: it allows us to represent information in simulated spiking neurons, as well as perform transformations on those representations.

First, a brief description of neuron biology, to clarify the terminology in use here. The main body of the neuron is the soma, which receives current as input along its dendrites. The input causes voltage to build up within the soma in a nonlinear fashion, as current is also leaking out through various channels. Once the voltage reaches a certain threshold the neuron undergoes a very short sharp change in voltage called a spike, or action potential. The action potential travels down the neuron's axon until it reaches a synapse—an interface between the axon and a dendrite of another cell. When the action potential reaches the

synapse it causes chemicals, called neurotransmitters, to be released, which travel to the dendrite of the post-synaptic cell and induce a current. This post-synaptic current then travels down the dendrite to the soma of the next cell, and the process continues. This is a very simplified version of neuron behaviour, but is sufficient to understand the neural modelling employed here.

Representation

The NEF represents information in a distributed manner, using the combined information from a population of neurons to represent a value (in this case, the high dimensional VSA vectors). There are two important components to representation: encoding a value into spikes, and decoding spikes into a value. In encoding, each neuron in the population will respond to a given input by emitting spikes at different times. To decode, we use the fact that those spikes are not random—they are a function of the input and the neuron's unique properties. Therefore if we know the population's neural properties, we can reconstruct the represented value by examining how each neuron is firing.

To encode a vector $x(t)$ (in realistic spiking neurons all computations occur over time, t) into the spike train of neuron a_i we define

$$a_i(x(t)) = G_i \left[\alpha_i \tilde{\phi}_i x(t) + J_i^{bias} \right] \tag{4.3}$$

G_i is a function representing the nonlinear neuron characteristics. It takes a current as input (the value within the brackets), and uses a model of neuron behaviour to output spikes. In our model we use Leaky Integrate and Fire neurons, but the advantage of this formulation is that any neuron model can be substituted for G_i without changing the overall framework. α_i, J_i^{bias}, and $\tilde{\phi}_i$ are the parameters of neuron a_i. α_i is a gain on the input; it does not directly play a role in the encoding of information, but rather is used to provide variety in the firing characteristics of the neurons within a population. J_i^{bias} is a constant current arising from intrinsic processes of the cell or background activity in the rest of the nervous system; it plays a similar role to α_i, providing variability in firing characteristics. $\tilde{\phi}_i$ represents the neuron's preferred stimulus, that is, which inputs will make it fire more strongly. This is the most important factor in the neuron's firing, as it is what truly differentiates how a neuron will respond to a given input. In summary, the activity of neuron a_i is a result of its unique response (determined by its preferred stimulus) to the input $x(t)$, passed through a nonlinear neuron model in order to generate spikes.

To decode the spikes from neuron a_i into the represented value $\hat{x}(t)$ we define

$$\hat{x}(t) = \sum_i h(t) * a_i(x(t))\phi_i \tag{4.4}$$

where $*$ denotes standard (not circular) convolution. Essentially this is modelling the current that will be induced in the post-synaptic cell by the spikes coming out of a_i. $a_i(x(t))$ are the spikes generated in Equation 4.3. $h(t)$ is a model of the post-synaptic current generated by each spike; by convolving that with $a_i(x(t))$ we get the total current generated by the spikes from a_i. ϕ_i are the optimal linear decoders, which are calculated analytically so as to provide the best linear representation of the original input $x(t)$. The equation for determining the decoders is as follows:

$$\begin{aligned} \phi &= \Gamma^{-1}\Upsilon \\ \text{where} & \\ \Gamma_{ij} &= \int a_i(x)a_j(x)dx \\ \Upsilon_j &= \int a_j(x)f(x)dx \end{aligned} \tag{4.5}$$

Note that in this case we have removed the temporal aspect, t. This is because we use only the average firing rate of the neuron for a given input, rather than the specific timing of the spikes. However, since the two measures (firing rate and spike timing) arise from the same neural model, we can use decoders calculated based on rates and apply them in the temporal case. $f(x)$ is the function to be performed on the represented value; when we want to decode the value represented by the population without modification, as is usually the case, then $f(x) = x$. Recall earlier we said that since each neuron's firing is a function of its input and its unique characteristics, if we know those characteristics and how a neuron is firing we can reconstruct what the input must have been. The optimal linear decoders represent that calculation; they map the current generated by the neuron onto the input that must have caused that current. Obviously this is only an approximation, as the noise and uncertainty make it impossible to reconstruct the input exactly. This is why we use population based representations; by summing these results across multiple neurons we average out these errors, leading to increased overall accuracy.

Linear transformations

We have now defined how to represent values in populations of spiking neurons. However, we also need to be able to carry out transformations—the VSA operations—on those values. The first transformation is a scale on the input (e.g. if we wanted to calculate $y = 2x$). Assuming that we have two populations A and B which we want to represent the x and y values, respectively, then this amounts to a question of how to set the connection weights between A and B. Recall that the spiking of neuron b_j is a result of its nonlinear response (G_j) to the input current. However, now that input is no longer a direct value, but is instead the output from population A. The output of population A is given by

Equation 4.4, so to calculate the firing of population B we can take that and substitute it in for $x(t)$ in Equation 4.3:

$$\begin{aligned} b_j(x(t)) &= G_j\left[\alpha_j\tilde{\phi}_j(\sum_i h(t) * a_i(x(t))\phi_i) + J_j^{bias}\right] \\ &= G_j\left[\sum_i h(t) * a_i(x(t))\alpha_j\tilde{\phi}_j\phi_i + J_j^{bias}\right] \\ &= G_j\left[\sum_i h(t) * a_i(x(t))\omega_{ji} + J_j^{bias}\right] \end{aligned} \tag{4.6}$$

In other words, the input current of neuron b_j is equal to the output current of all the a neurons connected to b_j, multiplied by the connection weights. This is the standard procedure for neural networks, but in most cases the connection weights, ω, need to be learned. The NEF formulation has allowed us to analytically determine the connection weights: $\omega_{ji} = \alpha_j\tilde{\phi}_j\phi_i$.

So far the value has not actually been transformed, the output of A is simply being passed directly to B. In order to calculate the transformation $y = 2x$ we simply multiply the connection weights by 2: $\omega_{ji} = 2\alpha_j\tilde{\phi}_j\phi_i$. More generally, if we want to multiply the vector x by the matrix C, we can calculate the weights as $\omega_{ji} = \alpha_j\tilde{\phi}_jC\phi_i$.

The second transformation we want to do is to combine two values (i.e. $z = x + y$). This is almost identical to Equation 4.6, except that now the input current is coming from two populations instead of one:

$$c_k(x(t) + y(t)) = G_k\left[\sum_i h(t) * a_i(x(t))\omega_{ki} + \sum_j h(t) * b_j(y(t))\omega_{kj} + J_k^{bias}\right] \tag{4.7}$$

where $\omega_{ki} = \alpha_k\tilde{\phi}_k\phi_i$ and $\omega_{kj} = \alpha_k\tilde{\phi}_k\phi_j$. We have simply added a second, analogous term for the second input population. We could do the same thing to combine arbitrary numbers of inputs.

The final step is to combine the previous two elements to compute arbitrary transformations of the form $z = C_1x + C_2y$. This is simply Equation 4.7, except $\omega_{ki} = \alpha_k\tilde{\phi}_kC_1\phi_i$ and $\omega_{kj} = \alpha_k\tilde{\phi}_kC_2\phi_j$. Thus with these techniques any linear transformation can be computed using simulated spiking neurons.

Nonlinear transformations

Notice that the linear transformation discussion has been solely about encoding from value to spikes, there has been no discussion about decoding from spikes to values. This is

because the decoding is unaffected by these linear transformations. Recall that the optimal linear decoders map from output current to the input value that caused that current. These manipulations are only changing the input current—the mapping remains the same. However, it is also possible to compute different mappings, making it possible to approximate nonlinear functions such as multiplication.[3] This involves manipulating $f(x)$ in Equation 4.5.

There are a number of methods to perform multiplication ($z = xy$) using spiking neurons, but we will focus on the technique used in our research. This involves a two stage process, using an intermediate population M. The M populations takes x and y as input and combines them into a two dimensional value $m = [x\ y]$. This is a linear transformation, and so is accomplished as described above. Then the decoders of M, instead of mapping onto the same two dimensional space as the input, map onto a one dimensional space by setting $f(m) = m_1 * m_2$ in Equation 4.5. In general terms, whereas before the decoders were used to directly translate output current into the value that caused that current, now they are adding a transformation to that translation. The decoders are still only linear weights and so can only approximate a nonlinear transformation such as multiplication, but they can do so sufficiently well for our purposes. Together with the linear transformations discussed in the previous section, this is sufficient to carry out all the VSA operations.

Nengo

Our model is developed using Nengo, a standardized software implementation of the Neural Engineering Framework. Nengo is designed to automate the computation involved in the NEF, providing the modeller with high-level commands so that the focus can be on the overall network architecture and behaviour. For example, in Nengo the modeller can create two populations of neurons, create a connection between them that computes a specific transformation, and then run the whole network. The Nengo software will take care of generating the neurons, calculating optimal linear decoders, calculating connection weights, and generating spikes as the network runs. Nengo also provides a scripting interface, which allows us to describe models in a high level language and then see them implemented in a graphical interface (Stewart et al., 2009b).

We chose to implement our model in Nengo for a number of reasons. First, it saves on both development time and error rates by utilizing code that has already been empirically tested on a number of models. Second, the GUI provides us with an intuitive high-level description of our model, and allows us to make modifications on the fly without returning back to the code. Third, by using the standard Nengo interface we make it easy to integrate our model with any other systems developed in Nengo, or to share our model with other

[3]Multiplication is particularly relevant for VSAs, because circular convolution can be computed by performing an element-wise product in the frequency domain. This is discussed in Section 4.4.2.

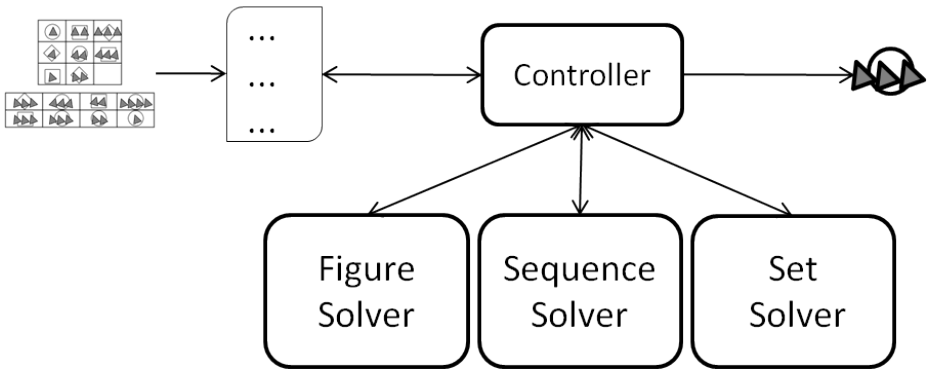

Figure 4.1: General model architecture

NEF researchers; this facilitates the development of integrated models of various brain systems.

4.4 Model

We will begin with a high level description of our model, and then clarify how each component operates. A general picture of the model's structure is shown in Figure 4.1. The rule generation systems are the three modules shown along the bottom. Each one is implemented in neurons, and takes the matrix information as input and returns the rules it has generated as well as its hypothesis as to what belongs in the blank cell. The controller is responsible for coordinating these systems, implementing the higher level problem solving strategy (e.g. which module to try first, and how to deal with their output). The controller is also responsible for inputting the matrix data and outputting the model's final response. It is not implemented in neurons, but is constrained by what we know of the underlying biology. [4]

4.4.1 Rule types

The three rule generation modules are designed to find different types of rules. The different types are distinguished by the type of information that constitutes a rule. Matrices containing the three types of rules are shown in Figure 4.2. Figure rules involve the first two cells being combined in some way to form the third; what needs to be learned is the

[4] Although the controller is not implemented at the neural level in this model, there do exist realistic neural models to perform these tasks (see Stewart et al. 2010).

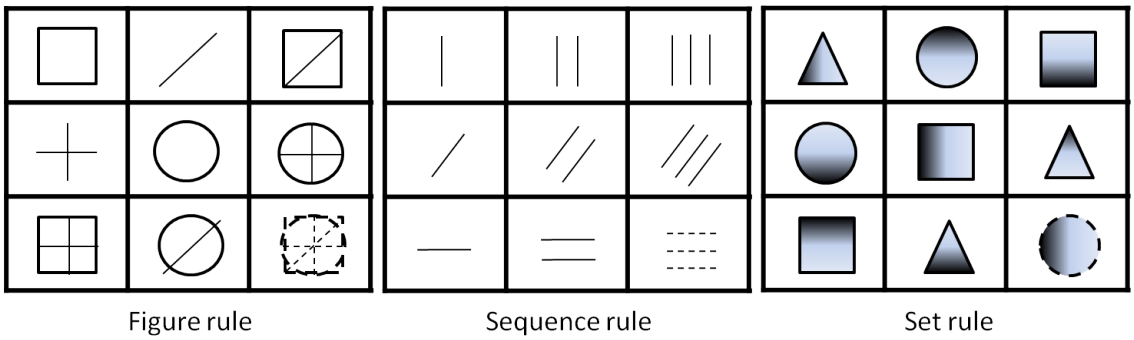

Figure 4.2: Matrices demonstrating the rule types associated with each rule generation module.

operation being used to combine the two cells. For example, in the expression $A + B = C$, what is being learned is the $+$. Sequence rules involve an orderly progression—increasing in number, rotation, scaling, and so on. These sequences are defined by an iterative transformation applied to the previous item in the sequence; for example, the sequence $1, 2, 3$ is defined by the operation $+1$ ($1 + 1 = 2$, $2 + 1 = 3$, and so on). The 1 is the rule being learned in this case. Set rules involve a set of features that are present in each row/column, usually with their order being shuffled. In this case the rule being learned is the items that make up the set.

We need three different modules to find these rules because they represent three fundamentally different types of computations. However, these are not arbitrary distinctions we have created just to make our model work, they are supported by experimental work on the RPM. There have been several studies investigating the distinction between spatial and analytic problem types. Spatial problems involve rules that can be extracted directly from the visual features of the matrix, without the need for further processing. Examples of this include figures being superimposed on each other or rotating/scaling (this encompasses the figure and sequence types used in our model). Abstract problems require that the matrix data be extracted into a higher level space. For example, there is nothing in the visual characteristics of a circle and a square that indicate that a triangle should follow; it is only when the subject steps back and notices that it is not the shapes themselves that are important but the shapes as generic items in a set that they can solve the problem.

Hunt (1973) was the first to make this distinction when he found that he needed two different algorithms to solve spatial versus analytic problems. However, the fact that an algorithm requires it does not prove that humans also have two separate systems, especially when there is no attempt to base the algorithms on human cognition. Kirby (1983) found experimental evidence for this distinction in human subjects by creating

matrices with two "correct" answers—one if the subject employs a spatial rule, and another if the subject is using an analytic rule. He showed that he could cause subjects to use one strategy or the other depending on the verbal instructions used while administering the test and the order the items were presented in. This demonstrates that humans can use at least two different types of reasoning to solve matrix problems, and switch between them. More evidence comes from DeShon et al. (1995), who found that they were able to selectively impair performance on analytic versus spatial type problems by forcing subjects to verbalize while solving them. The theory is that when verbalizing, subjects are focusing their mental efforts in the abstract, linguistic areas of the brain—this causes interference when they attempt to use spatial reasoning, but not in the abstract case. Regardless of the interpretation, this study provides evidence that there are qualitative differences between reasoning types. There is also evidence that these types of reasoning activate different areas in the brain. Prabhakaran et al. (1997) found that spatial RPM problems have relatively enhanced activation in right hemisphere regions, while analytic problems showed an opposite emphasis on left hemisphere regions. More recently, Golde et al. (2010) investigated sequence versus set type problems, and found that the former showed activation in the premotor cortex while the latter activated more anterior prefrontal cortex regions.

There is no specific evidence for the three-way distinction made here, because it is a new suggestion that came out of our efforts to model these systems at the neural level. However, these studies show that the degree and type of distinction we have made is consistent with the experimental and neurophysiological evidence. For example, the fMRI studies show that there are different brain regions associated with different problem types—as in the architecture we present here—rather than the same region being used in different ways. In addition, the parallels between the spatial/analytic and figure/sequence/set distinctions mean that our results are consistent with previous classifications and their associated evidence. The mapping between the two is not perfect; in particular, sequence problems seem to sit somewhere in the middle between spatial and analytic. However, the spatial/analytic distinction is only a general classification, and does not completely explain the differences between items (Vigneau and Bors, 2008). Therefore we should not be too concerned if our classification, in particular a more detailed classification, does not perfectly match with previous work. An attempt to further investigate and clarify the distinctions between these theories would be an interesting avenue for future work.

4.4.2 Sequence solver

We begin with the sequence solver module, as it was the first to be developed. Each of these sections will follow a similar format; we will begin with a description of how we use VSAs to accomplish the task, and then outline how those methods are implemented at the

neural level.

VSA techniques

Recall that with sequence rules we are trying to find the iterative transformation that defines the sequence. Since we are working with VSAs, our iterative operation is going to be circular convolution; the unknown is the vector that when convolved with an item gives us the next item in the sequence. In other words, we want to find T such that $cell_{1,1} \otimes T = cell_{1,2}$ and $cell_{1,2} \otimes T = cell_{1,3}$. If we want to find $A \otimes T = B$, we can solve for T as $T = A' \otimes B$. However, we do not just want to calculate T for a single pair of A and B vectors, we want a general T for the whole matrix. To accomplish this we treat all adjacent pairs of cells as a set of A and B vectors, and extract a general transformation from that set of examples. Neumann (2001) has shown that we can accomplish this by calculating

$$T = \frac{1}{n} \sum_{i=0}^{n} A'_i \otimes B_i \tag{4.8}$$

In other words, we simply calculate a T for each pair of cells and take the average.

Humans cannot instantly calculate and average transformation vectors for the whole matrix as in the above calculation, so we modify it slightly in order to calculate the transformations sequentially:

$$T_{i+1} = fT_i + l(A'_i \otimes B_i) \tag{4.9}$$

where f is our forgetting rate and l is our learning rate. We can see that this is just a standard learning rule if we set $f = 1 - l$:

$$\begin{aligned} T_{i+1} &= (1-l)T_i + l(A'_i \otimes B_i) \\ &= T_i - lT_i + l(A'_i \otimes B_i) \\ &= T_i - l(T_i - A'_i \otimes B_i) \end{aligned}$$

and if we set $l = 1/(i+1)$ then we will end up calculating the average just as in Equation 4.8. The reason for using the more general formulation of Equation 4.9 will become clear in the neural discussion.

Thus we can calculate a general rule for the whole matrix by sequentially calculating rules for adjacent pairs of cells. This exemplifies the inductive process of extracting general conclusions from specific examples. To illustrate this process let us examine how the system would solve the sequence matrix in Figure 4.2. To begin, A_0 is the vector representation of one line and B_0 is the vector representation of two lines (we will ignore the orientation of the lines just to keep this example simple, but they are treated in exactly the same way). T_0 is zero, so $T_1 = A'_0 \otimes B_0$ which if we were to translate into English would be something

like "increase number of lines by one". However, this rule will be very noisy, affected by the random nature of VSA vectors and their computations as well as the neural noise. In addition, we do not have enough information to distinguish the correct rule; an equally plausible T_1 would be "transform everything into two lines". In the next step A_1 is the representation of two lines and B_1 is three lines. The result of calculating $A'_1 \otimes B_1$ is then combined with the previous computations (T_1). This will cause the information the two transformations have in common—increasing by one—to be reinforced, while extraneous information that the two transformations do not share, such as the specific number of lines, is not. Thus with each new pair the correct rule will increasingly emerge from the background noise, until by the end the system has generated a general rule that describes the transformation common to the entire matrix. The system can then select an answer by applying that transformation to $\text{cell}_{3,2}$ ($cell_{3,2} \otimes T = cell_{3,3}$), generating a hypothesis as to what belongs in the blank cell.

It is helpful to examine a case where this system fails, as it illustrates both how this module works and why we need different modules for different problem types. If we applied this module to the figure matrix in Figure 4.2 then A_0 would be a square, B_0 would be a diagonal line, and the transformation would be something like "squares become diagonal lines". A_1 would be a diagonal line, B_1 would be a square with a line through it, and the transformation would be "diagonal lines become squares with diagonal lines through them". These transformations do not really have anything in common—at best they both contain the idea of becoming something involving a diagonal line—and so when they are added together they just become random noise rather than anything being reinforced. This is even more apparent when we look at the next step, when A_2 is a cross, B_2 is a circle, and the transformation is "crosses become circles". In this case it is very clear that there is nothing in common between these transformations, so no clear rule is going to emerge. Thus by the end of the matrix the transformation will be essentially random noise, and the system will not be able to generate an accurate prediction for the blank cell.

Note that what distinguishes the first from the second example is that in the first there is a common relationship between the vectors while in the second there is not. In Section 4.3.1 we said that all the vocabulary vectors are generated randomly; however, we now see that that cannot be true, otherwise there would be no more relationship between "one" and "two" than there is between "one" and "triangle". Our model assumes that people's mental representations of numbers contain innately the idea that they exist in a sequence—that two comes after one, three after two, and so on. Thus our vector representation of "two" is not a random vector, but is built based on the vector for "one" so that there will be a relationship between the two vectors. One concern is that in doing so we have simply built the solution in to the vector representations. This is to some extent true—the representation of "two" does contain the information that it is one more than "one". However, we are still only working with VSA vectors; that information is not stated as a

fact, it is bound up in the vector representation of "two". The problem solving being done is to extract that relationship from the vector. It would be analogous to saying that when someone solves $3+2=5$ they are not really doing math, they are just recognizing that 5 is 2 more than 3. That *is* math, just as recognizing that two lines is one more than one line is what is required to solve this problem. These types of items are defined by the fact that they are a sequence, and that information must be encoded some way in people's mental representations of the items in those sequences.

Neural implementation

There are two main components to the neural implementation. The first is a network to calculate the transformation for a particular pair of vectors ($A'_i \otimes B_i$). The second is an integrator to maintain the average transformation over all previous computations. The latter component reflects the important role of working memory in RPM performance (as discussed in Section 2.1.2). The integrator is responsible for storing and manipulating intermediate results, precisely the role played by working memory. Integrators have also been used in other domains to model working memory at the neural level (Eliasmith, 2005).

Calculating the pseudoinverse (A'_i) is a simple linear transformation; the difficult aspect is calculating the convolution (Equation 4.1). This appears to be a very complex operation, but it is made much simpler when we realize that the circular convolution of A and B is only the element-wise product of A and B in the frequency domain (Plate, 2003). Thus we can change the problem into performing a Discrete Fourier Transform on A and B, taking the element-wise product, and then performing the inverse DFT on the result.

The DFT is a linear transformation, defined by the matrix W where $W_{ab} = \cos(\frac{-2\pi ab}{d}) + \sin(\frac{-2\pi ab}{d})i$. Thus we can compute the DFT in neurons by setting $C = W$ in Equation 4.6 (although actually we do the computations separately for the real and imaginary components, as it makes calculating the element-wise product simpler). The output of these transformations will be new vectors in the frequency domain, which we then multiply together using the method described in Section 4.3.2. The final step is to take the resulting vector and transform it back from the frequency domain, another linear transformation whose matrix is simply the complex conjugate of W. The result will be the circular convolution of A and B, and if we first compute the pseudoinverse of A then we will have computed the transformation between A and B.

The second component is the integrator to calculate the average T over time. The basic purpose of an integrator is to maintain its current value by constantly feeding that value back into itself. In order to implement the learning rule described in Equation 4.9, we modify the inputs and recurrent information slightly. Figure 4.3 illustrates how the math maps onto the components of the integrator. When we input the calculated transformation for the current pair of examples ($A'_i \otimes B_i$) this will cause the integrator to drift toward the

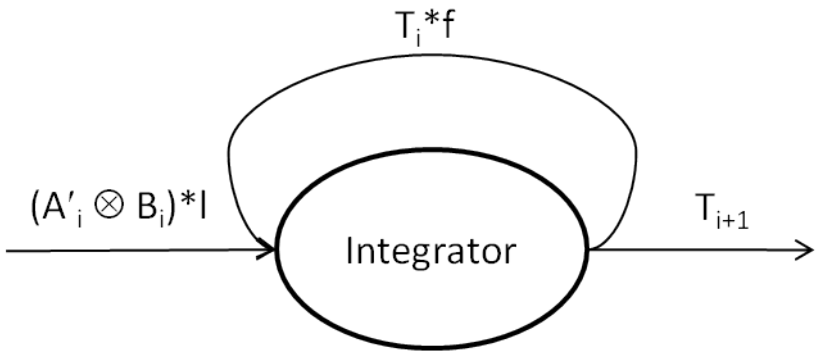

Figure 4.3: Integrator implementing the learning rule described in Equation 4.9.

inputted value. The rate of this drift is controlled by the learning rate, l. If this is all we do, then the final value stored in the integrator is the sum of all its inputs. However, we want to take the average, so we add a "forgetting rate" to the recurrent connection. This will cause the value stored in the integrator to be lost over time, so that we end up with an average rather than a sum. Ideally we would set $f = 1 - l$, as this will calculate the perfect average. However, recall that l is changing over time (it is equal to $1/(i+1)$). This is easy to accomplish on the input (we scale the value by l using a multiplicative network), but if we wanted to change f that would require constantly changing the weights on the recurrent connection. This is problematic both for computational and biological reasons, as it requires very fast and accurate changes on many neuronal connections. We strive to keep our computations as simple as possible, so instead we set f to a constant value that will give us an approximate average. Because the values being averaged have significant noise anyway, this approximation does not have a noticeable negative impact on overall performance.

Thus by the end of the run the value stored in the integrator will be the average rule for the whole matrix. We then convolve this rule with the vector representing $\text{cell}_{3,2}$ (using the same convolution network described above, but without the pseudoinverse) in order to generate a hypothesis as to the contents of the blank cell.

4.4.3 Set solver

VSA techniques

In the set solver module, the rule information we are trying to find is the items that make up the set. Our approach to this is relatively simple: we assemble the vector representations

of all three cells in the first and second row into single vectors using the superposition operator, and then average the two rows to create a general set for the matrix. If we then subtract the vector representations of $\text{cell}_{3,1}$ and $\text{cell}_{3,2}$ from this set, the remaining vector should be the last item in the set, $\text{cell}_{3,3}$.

As an example we will examine how this module solves the set matrix in Figure 4.2. Note that in this matrix there are two separate sets: one for shapes, and one for shading. It is possible to treat this as just one set with six elements, and that is the approach taken in our model, but for this example we will concern ourselves only with shape to keep things simple. The first step is to combine the representations of "circle", "square", and "triangle" into a single vector using the superposition operator. Recall that superposition creates a vector that is somewhat similar to each of its input vectors; therefore we will end up with a vector that is somewhat similar to "circle", "square", and "triangle", which we can think of as a combined (set) representation of those vectors. Note that there is no order information preserved, all we know is that those three representations are present in the combined vector. We then do the same thing for the second row, which results in the same set (since order does not matter). Averaging the sets from the first and second row will serve the same purpose as it did in the sequence module: it reinforces the information the two vectors have in common, which is the information we care about, and decreases the impact of any vector or neural based noise. We then subtract the representations of the last two cells, "square" and "triangle", leaving us with a vector that is similar to "circle", which the system returns as its prediction of what belongs in the blank cell.

Interestingly, this system will also be able to solve the sequence based problem discussed previously. We can think of "one line", "two lines", "three lines" as a set, and in the sequence based problems those sets just happen to always be arranged in the same order. This matches with the experimental data of Kirby (1983), who found that the abstract strategies represented a more powerful, general technique that could be used to solve spatial or abstract problems, while the spatial techniques could only be used to solve spatial problems. The question then is why subjects ever employ the spatial strategy (and why we would include different strategies in our model). Our theory is that spatial/sequence strategies are a more "natural" type of reasoning. It is the kind of cause and effect type reasoning that is useful in our everyday lives: a leads to b, b leads to c—I need food to eat, there is food at the grocery store, so I should go to the grocery store. This type of primitive problem solving comes so naturally to us that most of the time we do not even notice ourselves doing it. Abstract problem solving, on the other hand, is something we are taught or learn; it is a strategy we have to consciously employ. That is why people tend to use the sequence based strategy—it is our natural mode of thinking. We can also think of the differences between these two techniques in terms of their information load. In the sequence case, the rule is just a single value, the transformation vector. In the set case, the rule is a set of three different representations. This may place additional load on

components such as working memory, causing subjects to prefer the easier, sequence based strategy.

Not all sequence problems can be solved by set reasoning; for example, if row one is "one line", "two lines", "three lines", and row two is "four lines", "five lines", "six lines", then there is a common sequence between the two rows but not a common set. These types of problems tend not to occur in the RPM, but this demonstrates that, in general, subjects cannot abandon sequence based reasoning in favour of set based. With respect to our model, recall that our goal is to understand human reasoning, not simply achieve a computational solution to the RPM. This is why we include different modules that reflect the different types of reasoning observed in humans, rather than going only with the most powerful.

Neural implementation

There are no new techniques required for the neural implementation of this system. The addition and subtraction components are simple linear transformations, and the averaging of the two sets is accomplished in the same way as the averaging in the sequence solver module.

4.4.4 Figure solver

VSA techniques

Figure-type problems involve the combination of the first two cells in some way in order to form the third. The rule that needs to be found is the operation that combines the first two cells. There are many different ways in which the cells can be combined: summing features, subtracting overlapping features, subtracting non-overlapping features, and so on. However, rather than building separate reasoning systems for these different possible operations, we notice that they can all be reduced to one question: keep the features that are the same, or keep the features that are different. This encompasses all the different operations that are used to combine cells in the RPM (for example, subtraction is just an example of keeping the features that are different).

We can therefore reduce the problem to a simpler one: detect the features that are shared between the first two cells, detect the features that are different, and determine which of those feature sets shows up in the third cell. In this case the rule that needs to be learned is just the choice between these sets. To accomplish this we use a related technique: determining the contents of an unknown vector. Suppose we are given a VSA vector made up of some combination of elements. We do not know what those elements

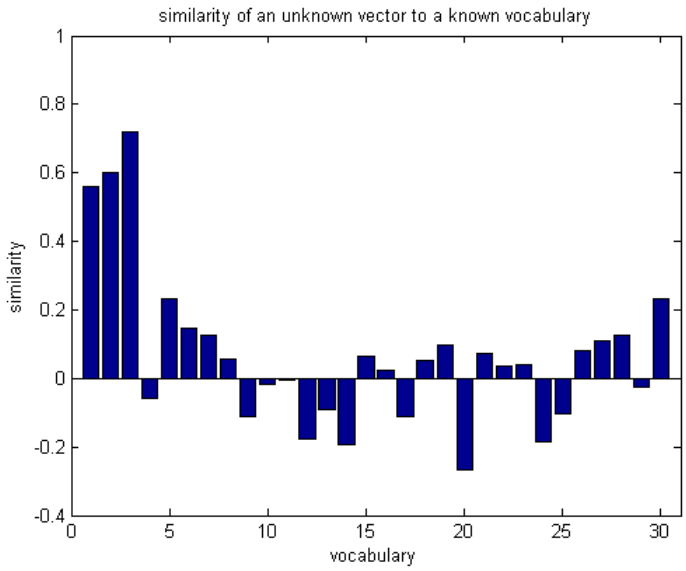

Figure 4.4: Comparison of a vector made up of a combination of unknown elements to the vocabulary representing all possible elements.

are, but we do know the vocabulary they are drawn from (i.e. we know what elements they *could* be). We can therefore take our unknown vector and determine how similar it is to each of the words in the vocabulary, which will give us a picture something like Figure 4.4. Notice that the first three words are noticeably more similar to our vector than any others; we could then guess that those three elements are what make up our vector. We use slight variations on this technique to detect same and different features.

To detect features that are the same we sum the vectors representing the first and second cell. If the two vectors had any features in common, then those features will now be represented twice in the combined vector. Therefore if we check how similar the combined vector is to the words in the vocabulary, it should be twice as similar to any features that were overlapping than it is to non-overlapping features. In other words, we should get a picture very similar to Figure 4.4, except the features that are noticeably higher will be those that are shared between the first two cells.

To detect features that are different we subtract the vectors representing the first and second cell. This will cause any features that the two vectors had in common to cancel out, and we will be left with only the features that were different. We then perform the same analysis as in Figure 4.4 to determine what elements are present in our combined vector. One complication is that any features from the second cell will be negatively represented due to the subtraction; their similarity will therefore be significantly more negative rather

than positive, but they will still be noticeably different from the non-present words.

With these two tools we can create a vector representing all the features the first two cells have in common, and another representing all the features that are distinct. We then compare each of those vectors to the third cell and check which is more similar, in order to determine which set better reflects the contents of the third cell. As usual we repeat this process for the first two rows and average the result, giving us an overall rule for the matrix (recall that rules are just a choice between the two feature sets). To pick an answer, we generate the two sets based on the cells in the third row, and then use our rule to tell us which one belongs in the blank cell.

One possible concern here is that we have returned to the reliance on preprogrammed rules. In this case the two rules would be "keep the features that are the same" and "keep the features that are different", and the system is simply recognizing which of these two rules fit. To some extent this is true—that is what the model is doing. The difference lies in the level of complexity of these rules. Although we describe the rule analytically as "keep the features that are the same", what this actually represents is the automatic, almost subconscious processing that we are constantly doing on the world around us. Noticing things that are the same or different is how we define our environment; it is so fundamental that most of the time we do not notice any thought going into it. For example, when we see two siblings, we can recognize that they look similar long before we analytically determine that they have the same nose or eyes. Any model must ultimately rely on some basic cognitive functions, and this same/different detection is what we are using in this module. These abilities—detecting same and different features—are not specific to matrix reasoning, but are general cognitive capacities. We could call this relying on predetermined information, but the difference lies in the level of abstraction of this information, and whether we could reasonably suppose it to be "built in" to human cognition.

Neural implementation

Creating the combined vectors, either by summing or subtracting the first two cells, is a linear transformation accomplished using the techniques described in Equation 4.7. The main difficulty is in detecting which features are present in those combined vectors. We do this by creating populations that respond selectively to each vocabulary word.[5] For example, if there are 30 features that could be present in the combined vectors, then we create 30 populations. We set the connection weights on each of those populations such that the input to the population will be the similarity between the input vector and the

[5]This is heavily based on the work of Stewart et al. (2009a) on implementing cleanup memory at the neural level.

vocabulary vector associated with that population.[6] Each population outputs the vector it is associated with, with a strength proportional to its input. We then sum the output of all these populations back into a combined vector. Alternatively, we could create one population and then sequentially change the input weights on that population to cycle through all the words in the vocabulary. The two approaches will perform essentially the same, they represent a trade-off between number of neurons and computation time/weight changes.

The result of only this setup would be that the output would be the same as the input. For example, suppose our vocabulary is a, b, c, d, and e, and they are associated with the A, B, C, D, and E neural populations. We form the combined vector, and suppose it is 0.3, 0.2, 0.6, -0.1, and 0.8 similar to each of those vectors, respectively. The input to the A population is the similarity between the combined vector and a, or 0.3. This will cause the A population to output a with a strength of 0.3. The same is true for B, which will output b with a strength of 0.2, and for the rest of the neural populations. When we combine all these outputs together, we will have a vector that contains $a * 0.3$, $b * 0.2$, and so on. In other words, it will be 0.3, 0.2, 0.6, -0.1, and 0.8 similar to our vocabulary—the same vector we started with.

The important modification is to get the populations to only respond to inputs over a certain threshold. In order to accomplish this we approximate the nonlinear function

$$f(x) = \begin{cases} x & : x \geq t \\ 0 & : x < t \end{cases}$$

using the linear decoders (see Section 4.3.2).[7] The main nonlinearity in this function is the step at t. It is impossible for the neurons to represent this perfectly, but again in this case we can be satisfied with an approximation. The important aspect is that inputs below the threshold will cause the population to output nothing—whether that threshold is at exactly t or $t \pm 0.05$ is not going to be critical to the system's success, given the noise inherent in neurons and VSAs.

Returning to our example, suppose we set $t = 0.5$. This means that A, B, and D will all output 0, as the input vector is not similar enough to a, b, or d to cause them to fire. Thus the vector output from the system will be a combination of vectors c and e—the features that were common to the first two cells.

[6] This means that the C matrix in Equation 4.6 will be equal to the vocabulary vector, so effectively the transformation calculates the dot product between the input and that vector.

[7] When we are trying to detect features that are different recall that we can have positive and negative similarities, so we change the function slightly to

$$f(x) = \begin{cases} |x| & : |x| \geq t \\ 0 & : -t < x < t \end{cases}$$

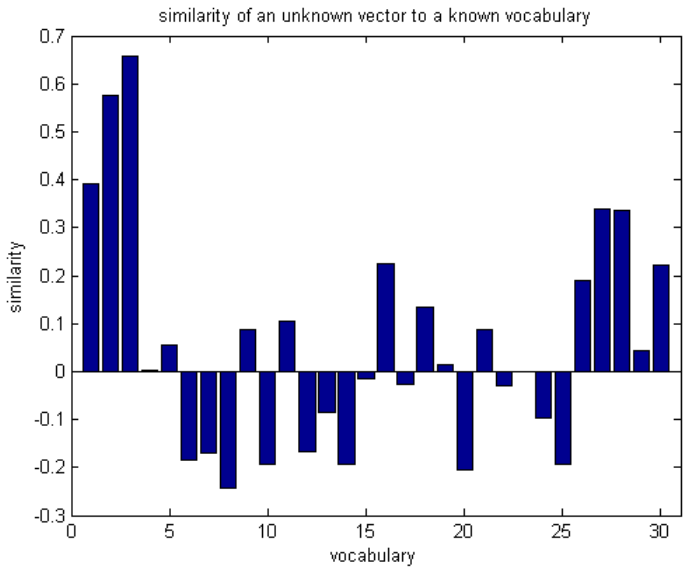

Figure 4.5: The same comparison as in Figure 4.4 but using lower dimensional vectors

This system is not working as well as we would like. The basic problem is that the system generates too many false positives or false negatives—vectors present in the output when they should not be, or not present when they should be. One possible solution is to improve the accuracy of the neural components; however, we have made significant improvements in this regard and it has not had a noticeable impact on the overall accuracy of the system. We believe that the root problem is that the VSA vectors we are using are too low dimensional to accurately differentiate below-threshold and above-threshold variables. For example, Figure 4.4 was created using 50 dimensional vectors, and there is a fairly clear distinction between present and not-present vocabulary words. If we drop the dimension down to 30, as in Figure 4.5, the picture is much less clear. With lower dimensional vectors it is more difficult to separate the random similarities from those caused by the combination of features; no matter where we set the threshold in Figure 4.5, we are likely going to get some false positives or false negatives. The obvious solution then is to use higher dimensional vectors. However, we are restricted in this respect by the computational power at our disposal; higher dimensional vectors require more neurons to represent, consequently slowing down the entire system. It currently takes about 24 hours to collect just one set of accuracy data using 30 dimensional vectors, and adding more dimensions comes at an increasingly high cost. So far we have simply not had time to explore these possibilities, but it is something we intend to investigate in the future.

4.4.5 Cleanup memory

VSA techniques

We discussed the role of cleanup memory in VSAs in Section 4.3.1. When applying this to the current problem, the vocabulary of the cleanup memory is all the rules that have been learned in the past. Then after a rule has been calculated for the matrix, it is passed through cleanup memory to see if it is an instance of a previously calculated rule. If so, then that clean rule is used to predict the blank cell rather than the messy version. The cleanup memory rule is "cleaner", even though it has been computed by the same system, because we update the cleanup memory over time so that the rules are an average of all previous computations of that rule. Thus each subsequent calculation benefits from previous work, effectively drawing from a larger pool of examples to compute a more accurate transformation. The details of this updating mechanism will be discussed in Section 4.4.6.

The cleanup memory is useful in that it improves the accuracy of the system, but it also serves an important theoretical purpose: it bridges the gap between this model of dynamic rule generation and previous theories of a library of known rules. Rather than contradicting previous theories, we are improving on them by explaining where that past knowledge comes from. We also now have an explanation as to why the use of that knowledge is a dynamic, fluid process rather than crystallized. The important aspect of a cleanup memory is that it depends upon its input. It is not abstract storage that can be queried at will, the subject needs to first generate an answer that is accurate enough to be recognized. The cleanup memory can be used to improve the accuracy of rules, but it cannot generate them out of thin air; thus subjects can still benefit from their past knowledge of rules, but the critical aspect of performance will be their fluid rule generation ability. This resolves Hunt's dilemma, and means that the reliance of previous models on a library of known rules does not render their insights useless, they can simply be reinterpreted from this new, biologically plausible perspective.

Neural implementation

Note that the underlying problem here is exactly the same as that in Section 4.4.4: we want the system to respond selectively to inputs when they are similar enough to our stored values (in this case rules rather than vocabulary words). Our neural implementation is therefore exactly the same; we create neural populations associated with each rule in cleanup memory, and those populations will output their rule only when the similarity of the input to their rule exceeds a certain threshold

This cleanup memory is less prone to error than the figure solver, as we have fewer rules than vocabulary words and a correspondingly lower chance of false positives or negatives.

In addition, only one population should be active, which means that the system does not need to combine the results of multiple populations. As a result, this cleanup memory system works quite well, not encountering the same problems as in Section 4.4.4.

4.4.6 Controller

The strategies used to solve the matrices in the previous sections have been very serial: take the input, generate rules, and pick an answer. This is sufficient for the simple matrices, but as they become more complicated we require more complex problem solving behaviour. This is the role played by the controller—coordinating the various modules so that they can work together to solve a matrix. This section of the model is not implemented at the neural level, but we still strive to motivate its operation based on observed human behaviour. In addition, there do exist realistic neural models that are able to perform the types of operations described here (Stewart et al., 2010).

One important role of the controller is quality control. The neural modules do not have a sense of how "good" their rule is, they just compute the best answer they can and return it. For example, if the sequence module is given a set type problem, it does not recognize that there is no sequence to be found; it just computes the best transformation it can and makes a prediction based on that. It is up to the controller to take that output and check it to see if it represents a valid rule or not. The basic technique for performing this check is the same for all modules: the controller checks to see if the rule is consistent with the known information in the matrix. For example, the rule returned by the set solver module is supposed to be a set of items that exist in each row. The controller therefore takes the rule that is returned and checks if the first and second row do in fact match that set. If it finds a good match then the controller uses the information returned by the module, if not it tries another module to see if it can do better. We use experimental data to guide the order in which the modules are tried. Kirby (1983) and Carpenter et al. (1990) both found that subjects tend to try simple spatial rules first, only moving on to more abstract types if the spatial techniques fail. We therefore begin with the figure solver module, and then proceed to the sequence and then set module if no good rules are found.

Another reason we require a controller is that most RPM problems involve multiple rule types. Each module is built to only understand one type of problem, so if there are mixed types then that will appear as random noise to the module and make it very difficult to find a correct rule. The controller is responsible for breaking the problem down into subproblems, the hope being that those subproblems will be governed by only one rule type and therefore solvable by the neural modules. This follows a similar iterative approach: the controller first tries to find rules at the most general level, and then if that fails continues to break the problem down into more fine-grained partitions until a successful rule is found (or the problem cannot be broken down any further).

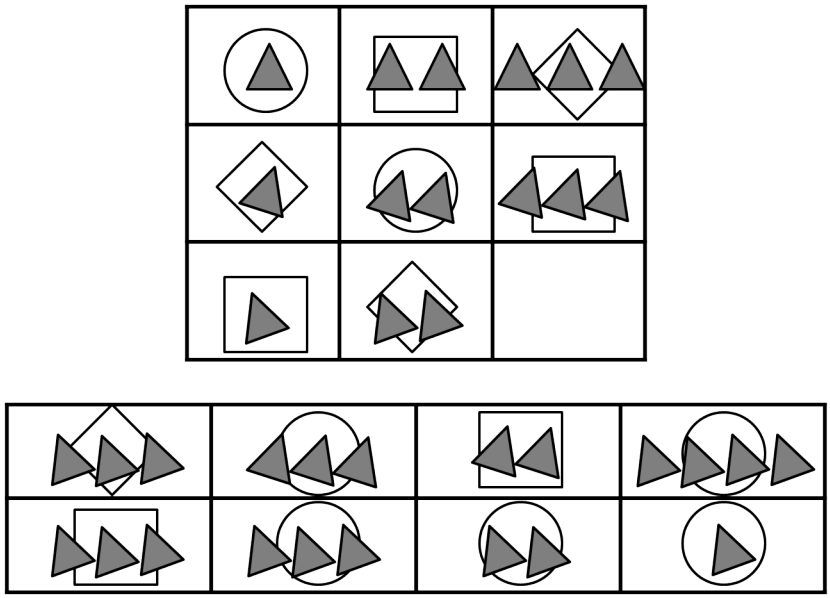

Figure 4.6: A matrix involving multiple rule types

To understand how the problems are broken down, we first need to discuss how the matrix information is encoded into VSA form. We will use Figure 4.6 as an example. First, the matrix is broken down into features. These represent the high-level components of the matrix; in the example, one feature would be the large background shapes and another feature would be the foreground triangles. This can be thought of as the correspondence finding step, identifying the broad groupings (correspondences) of components. This is the most important problem solving step that is done by hand rather than computed by the model; as discussed in Section 4.2, this is a necessary limitation that allows us to focus our attention on the rule generation modules. The next step is to describe the attributes of these features. We do this using a set of *attribute* $\otimes$ *value* pairs; for example, we might describe the foreground triangles in $\text{cell}_{1,2}$ as *shape* $\otimes$ *triangle* + *number* $\otimes$ *two* + *shading* $\otimes$ *solid* + *orientation* $\otimes$ *vertical* and so on. We can therefore define three different levels of representation for a matrix; the most general represents each cell as a combination of features, then we can represent single features as a combination of attributes, and finally we can represent single *attribute* $\otimes$ *value* pairs. For example $\text{cell}_{1,2}$, consists of two features, A and B, and $A = \textit{shape} \otimes \textit{triangle} + \textit{number} \otimes \textit{two} + \textit{shading} \otimes \textit{solid} + \textit{orientation} \otimes \textit{vertical} + \ldots$. The first level would look at the background shapes and foreground triangles together $(A+B)$, the second level we could focus only on the foreground triangles (A), and the third level we could look specifically at the orientation of those triangles $(\textit{orientation} \otimes \textit{vertical})$.

These levels correspond to the ways in which the problem can be broken down by the

controller. It first sends each module a description of the matrix at the most general level, and sees if they are able to find any good rules. If they are not, it then tries them at the feature and then attribute level. Again we use this ordering because it seems to be the way in which humans approach the problem; they first try to find quick, broad rules, only looking at a more detailed level when that proves unsuccessful (Carpenter et al., 1990).

The third role of the controller is to combine the results from these various modules together in order to select an answer. As we have seen, the controller can have results from various modules and various levels of analysis, and it needs to put them all together somehow. For example, if everything goes well the controller might get three rules for Figure 4.6: a feature-level set rule for the background shapes, and two attribute-level rules for the foreground shapes (one for number and one for orientation). Along with these rules, each module also returns its prediction of what the blank cell should look like. To combine them, the controller scores each answer based on its similarity to each of the predictions. The answer that ends up with the highest score is selected as the overall response.

One final role of the controller is to update the cleanup memory. Recall that the cleanup memory provides more accurate results by storing the rules that have been computed previously. The reason this gives more accurate results is that the controller updates the rules stored in cleanup memory whenever a new rule is calculated; the rules in memory therefore represent the results of many different calculations, and are less affected by noise than the particular calculations for one matrix. Thus there are two tasks that the controller performs whenever a new rule is generated. If that rule is not already in the cleanup memory, the controller adds it in so that it will be present in the future. If the rule is already present, then it modifies the stored rule slightly based on the current rule. Effectively, the rule stored in memory is a rolling average of all previous instances of that rule.

4.5 Neurophysiological basis of the model

There is a tendency in cognitive research to scan a subject while they perform a task, see which brain areas become active, and conclude that those brain areas are responsible for that task. This works reasonably well when investigating very low-level abilities, such as saccading to a target, but increasingly less well as we move into more complex tasks. The problem is that as tasks become more complicated they involve increasingly large and diverse areas of the brain, all interacting in unknown ways. The result is that a brain area will become active in many different tasks, and therefore any given tasks is less helpful in illuminating what that area of the brain is actually doing. Intelligence tests are some of the most complex cognitive tasks, and have proportionally ambiguous results in the scanner. We must therefore be very careful when attempting to map the components of our model

onto these results. Nevertheless, an effort to do so is valuable for three reasons. First, it helps to ensure that our model is faithful to the biology, and is not relying on functions with no known parallel in the brain. Second, it helps us map the results of the model onto human data; if we know how the model maps onto the brain, we can make predictions as to the neural and network properties of the brain regions involved. Third, it assists in our overall goal of achieving a better understanding of general intelligence; by mapping the model onto the brain, we help apply the insight gained in this model to the broad range of intelligent behaviour associated with those brain areas.

One of the most comprehensive theories of intelligence in the brain to date is that of Jung and Haier (2007), which they call the Parieto-Frontal Integration Theory (P-FIT). This theory draws on a broad range of evidence—brain volume data from MRI, white matter tract analysis using DTI, functional data from PET/fMRI, lesion studies, and genetic analysis—that has been accumulated in the modern neuroscientific study of intelligence, and attempts to unify it into an overall structure. One of the main strengths of this theory is that it does not try to be more than it should; it recognizes the danger of making specific functional claims, and instead focuses on identifying the broad regions that come up over and over in these studies and how they might interact to produce intelligent behaviour.

The core of P-FIT is, as the name suggests, a network of frontal and parietal areas; specifically, the dorsolateral prefrontal cortex (DLPFC), inferior and superior parietal lobule, and the anterior cingulate (ACC). These areas are picked out because they are consistently highlighted in many different types of studies: larger cortical volume in these areas is associated with higher intelligence, they almost always are found to be active when subjects are scanned while performing complex tasks (such as the RPM), and the high level of genetic control over these areas matches the high measured heritability of intelligence.

In addition to identifying brain regions correlated with intelligence, we must also associate functional roles with those regions if we are to map them onto the model. Jung and Haier describe four functional components. First, sensory information enters the brain in primary sensory cortices, and this information undergoes some low-level processing in the unimodal association cortices (e.g. extrastriate areas). Second, sensory information is assembled in the parietal cortex, where it is abstracted and formed into a structured symbolic representation. Third, this symbolic information is passed forward to the prefrontal regions for processing (this function is necessarily vague, as it is the least well defined by the available research). Fourth, the results of the processing are passed to the ACC, which is involved in response selection.

These roles map quite well onto the various components of our model. Visual matrix information enters through primary visual cortex. It is then passed to parietal areas, which translate the visual information in to the structured VSA form. The VSA representations are then passed to the frontal areas, where the various rule generation modules compute

rules and generate hypotheses as to what belongs in the blank cell. Finally, the results of these computations are passed to the ACC, which selects one of the eight possible answers. Note that our model was not designed with P-FIT in mind, rather these parallels were an emergent property of the modelling process.

We may also seek to move beyond P-FIT and provide a more detailed location of the model's components. First of all, the three neural modules: figure solver, sequence solver, and set solver. The most interesting result in this area is recent work by Golde et al. (2010). They divide RPM problems into the same sequence and set categories as we developed (which they call sequential versus distributive). Their hypothesis was that premotor cortex, which has been associated with learning and generating sequences in other domains, such as motor planning, would also be selectively involved in sequence rather than set based rules. They tested this hypothesis by placing subjects in a scanner as they solved sequence and set based problems, and compared the brain activation in the two cases. Their results confirmed their hypothesis: PMC was significantly more active for sequence problems, while set problems activated more anterior areas (i.e. DLPFC). These results are quite new, and we should be very careful making claims based on single imaging studies, but on this basis we might tentatively locate the sequence solver module in the PMC, and set solver further forward in DLPFC areas. It is unclear how the figure solver module fits into this picture, as these types of problems were not investigated by Golde et al.. The combinations could be thought of as types of sequences, and therefore associated with the PMC. Alternatively, depending on how low-level we want to place the same/different computations, we might even put the bulk of this module in visual processing areas. However, without specific evidence we believe it is better to locate the figure solver module in the DLPFC, as this area of general high level cognition has been much better established by the available research.[8]

We can also provide more detailed analysis of the functional components of the controller. The controller has three main roles: coordinating the input and output for the neural modules, evaluating the correctness of that output, and selecting a response. The first is what is commonly referred to as executive functions—the management of the contents of working memory in order to accomplish a goal. This has been the subject of several different investigations, and is almost always associated with the DLPFC. For example, D'Esposito et al. (1995) conducted a study where subjects were asked to perform two relatively simple tasks: spatial rotation and semantic judgements (e.g. "is a cucumber a vegetable"). By themselves, these tasks showed no significant activation in prefrontal

[8]Note that we make no reference to the left or right lateralization of these components. There are some general results showing that spatial types of problems, which we would associate with the figure and sequence solver, are more right lateralized, while more abstract reasoning tends to be focused in the left hemisphere (Haier et al., 1988; Prabhakaran et al., 1997), but in general these complex tasks show strong bilateral activation. Therefore while it is very possible that these components would be restricted to one hemisphere, we do not have sufficient evidence to make that distinction.

areas, but when asked to perform both simultaneously DLPFC became active. This is presumed to represent the activation of the executive functions in order to coordinate the simultaneous performance of the two tasks. See Kane and Engle (2002) for a review of similar studies associating executive functions with the DLPFC. In light of this consistent evidence, we feel fairly confident in locating the memory management portion of the model in DLPFC.

The second component, evaluating correctness, has been the subject of work by Christoff et al. (2003). They created a task wherein subjects had to calculate the transformation between two objects, maintain that transformation in memory over a delay period, and then evaluate whether or not that transformation applied to two new objects. The latter task corresponds to the role of the second component, which takes the rule from the neural module and determines how well it applies to the matrix data. Christoff et al. found that rostrolateral prefrontal cortex, the area immediately anterior to DLPFC, becomes selectively activated when the subject must evaluate the transformation as opposed to generating it or maintaining it in working memory. Therefore the RLPFC might be a reasonable location for the second component of the controller.

The third component is response selection. We already mentioned that this role is associated with the anterior cingulate by P-FIT. However, we believe that there is better evidence linking this role to the basal ganglia. The basal ganglia are commonly believed to support action selection in the motor domain (Redgrave et al., 1999), and Hazy et al. (2007) have developed a biologically based model that demonstrates the basal ganglia's ability to perform more abstract cognitive action selection. These areas are well-integrated with the other aspects of the model, as there are strong reciprocal connections from the DLPFC to the input areas of the basal ganglia (caudate) and back from the output areas of basal ganglia (globus pallidus internal and substantia nigra pars reticulata) through the thalamus (Alexander et al., 1986). The involvement of basal ganglia in a matrix reasoning task has been demonstrated by Melrose et al. (2007), who investigated the neural correlates of sequence rule induction and found widespread basal ganglia activation, in particular the head of the caudate (the area receiving the largest number of connections from prefrontal cortex). In addition, recent work by Stewart et al. (2010) has demonstrated that realistic spiking neural models of the basal ganglia can perform exactly the kind of selection between competing alternatives that is required by this component of the controller. These results do not necessarily contradict P-FIT's emphasis on the ACC, as it could be that the two areas work together to accomplish response selection (the ACC is immediately adjacent to the head of the caudate). However, in light of these results we believe the response selection component is primarily located in the basal ganglia.

In summary, our neural modules are divided up as follows: sequence solver in the PMC, and set and figure solver in the DLPFC (the latter more by default than through any specific evidence). The controller is divided into three components: memory management in the

DLPFC, evaluation in RLPFC, and response selection in the basal ganglia. Note that we have several different tasks all converging on the DLPFC. This reflects the fact that the DLPFC is the area most often activated in scanning studies of these types of complex tasks. However, this does not mean that the DLPFC does all these things together; the DLPFC is a complex, heterogeneous region of the brain—the variety of tasks it is implicated in is simply an indication that there are various subdivisions within the DLPFC that have yet to be thoroughly investigated. In addition, in creating this mapping we must emphasize that we have not thereby specified the function of those brain areas. The regions we have highlighted almost certainly have a much broader function than the role played in our model, we are simply specifying in which of those functional areas our model components could lie.[9]

In respect to the investigation of general intelligence, we must emphasize that the mappings we have established here are only tentative. As discussed at the beginning, the results in this area are so complicated that we must be very careful associating specific functions with specific brain regions, particularly when the connection is established by only a handful of imaging studies. That being said, we believe the picture we have established here, based on P-FIT and related studies, is plausible, and demonstrates that our model can be situated in the human brain. The most important conclusion to be drawn from this analysis is that no one brain area is responsible for the performance of the model. Intelligent behaviour, such as that involved in solving the RPM, arises out of the complex interactions between these components. This is why modelling those components and their interactions at a realistic neural level is so important to understanding general intelligence.

[9]In fact, Duncan (2001) argues convincingly that the defining characteristic of these regions may be their ability to adapt to the task at hand, with areas having very broad functional roles and then fine-tuning those roles to match the needs of the current problem.

Chapter 5

Results

There will be two main sections in this chapter. The first will demonstrate that all the components work as described in the previous chapter, and that the model can generate general rules that successfully solve Raven's matrices. The second section will highlight some of the insight that can be gained from the model—for example, how it can be used to explain the spread of individual differences observed in humans.

5.1 Demonstrations

5.1.1 Sequence solver

In Figure 5.1 we see an example of the model arriving at an answer. Each line represents the model's confidence in one of the eight possible answers, and we can see that over time one of those answers (as it happens, the correct one) is selected as significantly more likely than any other. This data is here only for display purposes, it is not actually used in the module. Recall that in the actual model the neural module returns two vectors: the rule it has calculated, and its prediction of what belongs in the blank cell. The response selection being displayed here is actually a role performed by the controller. However, it is difficult to tell by looking at high dimensional vector output whether anything meaningful has been calculated. The computed answer shown in the figure is a direct result of the calculated transformation (since the eight possible answers are constant), so by looking at Figure 5.1 we can get a better picture of the model's calculation of a correct rule over time.

Having established that the system can generate a rule for a matrix, we can also investigate the nature of that rule. Does it represent an actual, general rule, or is the system "cheating", finding a specific transformation that just gets it the correct answer? If it is

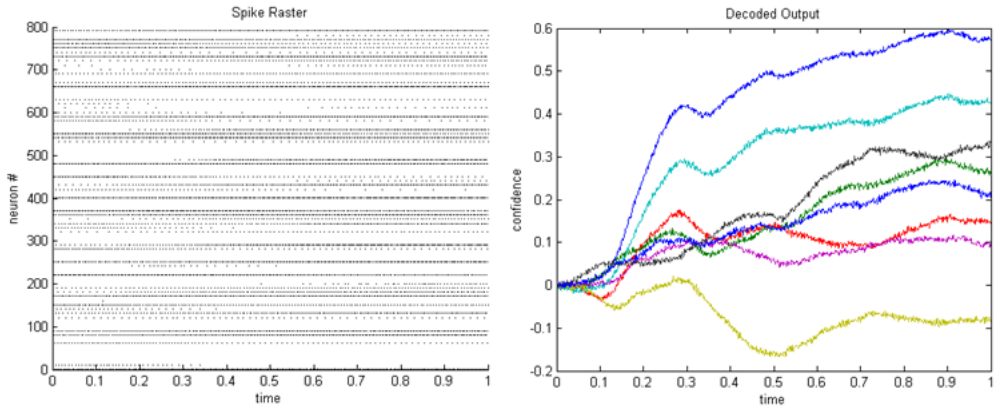

Figure 5.1: Output from the sequence solver module, expressing a confidence in each of the eight possible answers. The actual spike output is shown on the left, and the decoded information from those spikes (using the techniques described in Section 4.3.2) is shown on the right.

a general rule then it should be applicable in other cases. We will investigate this using the matrix shown in Figure 5.2. We have purposely made this matrix very simple, so that there are no complicating side-effects. At the end of the run the system's confidence in the eight answers is shown in Figure 5.3, demonstrating that it has found the correct answer. Ideally, it should have done so by calculating the rule "number of shapes increases by one". However, it could also have used the rule "display three triangles" with similar results. In order to investigate which is which, we can take the rule and apply it to different vectors. For example, if we apply it to the vector representing "four triangles", if it is a general rule then the output should be "five triangles". Note that the system has never actually encountered the vectors for "four" or "five" in the matrix—it must generate the answer based solely on its rule. This is what we have done in Figure 5.4, and as we can see it has found the correct answer. We could do the same for "four squares", to ensure that the rule is not specific to triangles; the results of this are also shown in Figure 5.4, again demonstrating the generality of the rule. Thus we see that the sequence solver system has successfully extracted the general rule that describes the relationship that governs this matrix.

5.1.2 Set solver

The results for the set solver look much the same as the sequence solver, so we will not go into them in as much detail. However, again we can demonstrate that the system is successfully computing the set that defines the matrix. We will use Figure 5.5 as an

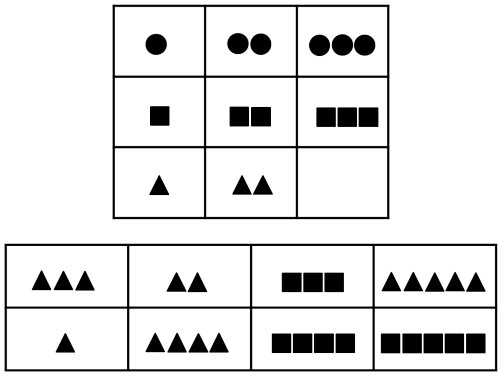

Figure 5.2: A simple sequence-based matrix.

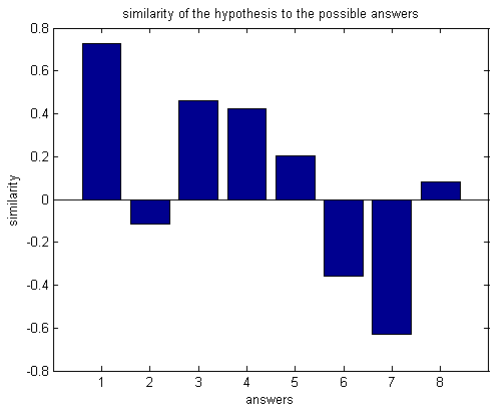

Figure 5.3: Similarity of the generated hypothesis to the eight possible answers, indicating that the model correctly believes the first answer to be correct.

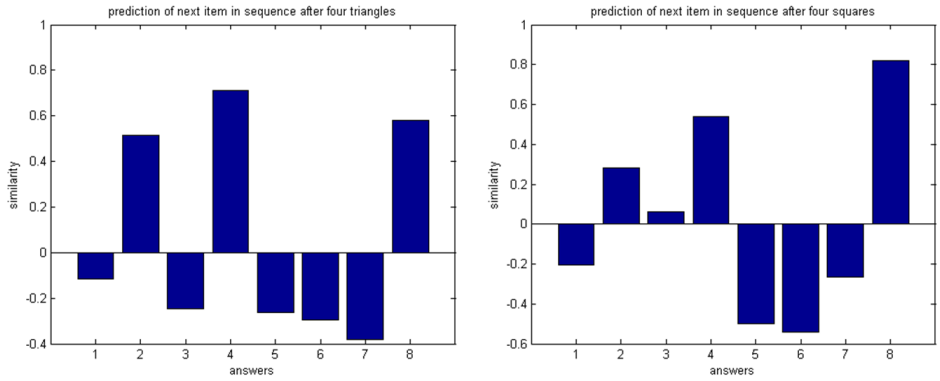

Figure 5.4: Demonstration of the generality of the calculated rules. The rule calculated based on the matrix in Figure 5.2 was applied to the novel vector representing four triangles and four squares, respectively. The model correctly selects the appropriate answer (five triangles and five squares) in both cases.

example. The results of the sequence solver module when given this matrix as input are shown in Figure 5.6. On the left we see the model's estimation of how likely each of the eight answers are, correctly picking answer number one.

The graph on the right is an interesting demonstration of how and why the model can fail. It is a comparison between the rule generated by the neural module and the eight possible answers. The rule is supposed to represent the set that defines the matrix, so it should be similar to "circle", "square", and "triangle". However, we see that while it has correctly picked out "circle" and "triangle", "square" is not as well identified and somehow "pentagon" and "cross" are present in the rule. This demonstrates the randomness inherent in using VSAs. There is no way the module could have calculated "pentagon" and "cross" being part of the rule, because they are not even present in the matrix. Their similarity is simply random; the vector generated for "pentagon" just happens to be similar to the rule calculated for this matrix. This is why increasing the dimensionality of the vectors increases the accuracy of the model—when the vectors are in a larger space, there is less chance of them coincidentally ending up near to each other. In this case the model was still able to find the correct answer, because after subtracting "square" and "triangle" it is left with a strong similarity to "circle". However, if the last row had been "triangle" and "circle", then the model would have had a problem. This demonstrates the stochastic performance of the model; even though it will find the correct answer for these simple matrices almost every time, there is still a chance that it will select an incorrect response.

In this case we do not need to worry about whether the rule is general or not. The rule that is being learned is the items that make up the set—we would not expect that

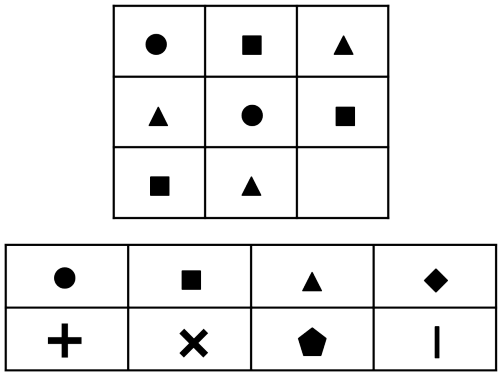

Figure 5.5: A simple sequence-based matrix.

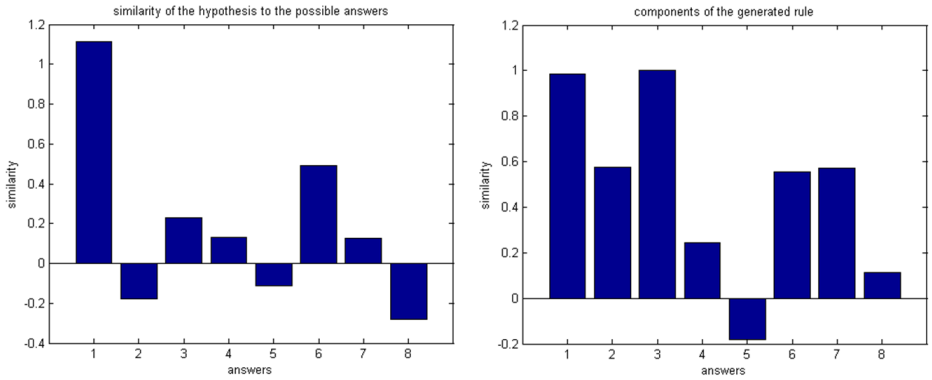

Figure 5.6: The results of the set module when applied to Figure 5.5. On the left is the similarity between the hypothesized blank cell and each of the eight possible answers, demonstrating that it has found the correct answer. On the right is a comparison between the rule generated (the set of items that define the matrix) and the eight answers, demonstrating which components make up the rule.

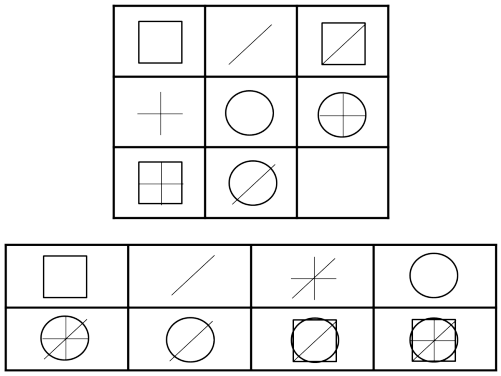

Figure 5.7: A simple figure-based matrix.

to generalize to matrices with different items. This again demonstrates the qualitative differences between the different rule types and their associated neural modules.

5.1.3 Figure solver

For the figure solver module we will use Figure 5.7. In Figure 5.8 we can see the system choosing between the features that are the same (+1) and the features that are different (-1). As we can see, the system correctly identifies that it should keep the features that are different. We can also see that it predicts the correct answer: a vector made up of a square, cross, circle, and line.

We can also see why this module works, and why it fails. In Figure 5.9 we see the output from the populations described in Section 4.4.4. Each of these populations is designed to respond to a certain word in the vocabulary; when the similarity between the input and that word exceeds a threshold, the population should become active. The purpose of these populations is to detect when a feature is present in the combined vector of the first two cells in the row (to detect features that are the same the combined vector is formed through addition, to detect features that are different the combined vector is formed through subtraction). The data from each row is presented for 0.2 seconds. We see the "circle" population performing correctly; there is no circle in the first row so it outputs 0 (0.0 to 0.2 seconds), and then there is a circle in the second and third row so it outputs approximately 1 (0.2 to 0.6 seconds). In the "square" population, however, we can see that even though there are no squares in the second row the population is still outputting a nonzero value from 0.2 to 0.4 seconds. This is a false positive—a population indicating a feature is present when it actually is not. In general the model was accurate enough on this matrix that this did not cause it to make a mistake in the final answer, but these are

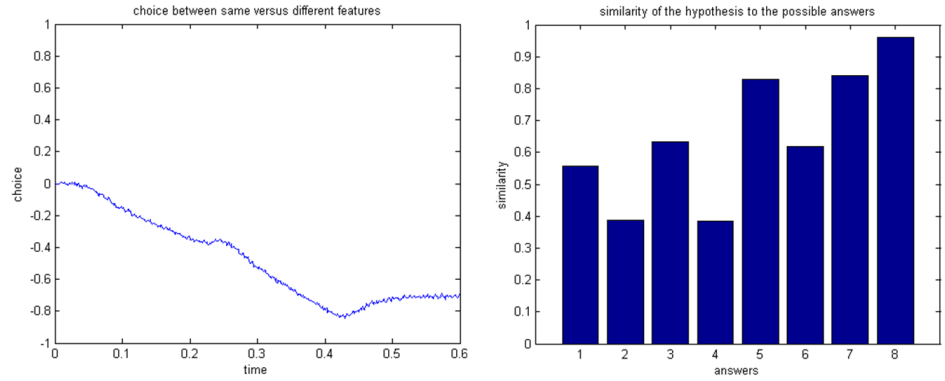

Figure 5.8: Results from the figure solver module. On the left we see it deciding between the same (+1) and different (-1) features, and on the right we see the usual comparison between the generated hypothesis and the eight possible answers, indicating that it correctly solves the matrix.

the errors that can cause the figure module to fail.

5.1.4 Learning

Recall the role of cleanup memory: it takes noisy rules as input, and outputs the clean version of the same rule stored in memory. The version of the rule stored in memory is built up over time based on previous calculations. In other words, as soon as the system realizes that it is calculating a rule that it has seen before, it can make use of all the data and calculations it has seen in the past in order to provide more accurate output.

We demonstrate the use of this cleanup memory in the sequence solver module in Figure 5.10. We created five matrices that had different surface appearances but required the same underlying rule to solve. We then ran the model on these five matrices, starting with an empty cleanup memory. In the first case learning was turned off, so the cleanup memory did not improve over time and remained empty. In the second case learning was turned on, so over time the rule stored in cleanup memory would become better and better. Notice the difference between the two results: without learning, there is no real change—in every case, the top answer has a confidence of about 0.6; with learning, the first answer is selected with a confidence of 0.6, but then we see steady improvement—0.8 for the third matrix, and 1.0 for the fifth. The matrix that the fifth run is working on is exactly the same in both cases, but the system is able to much more accurately pick out the correct answer when it can benefit from the work done in the previous five matrices.

This demonstrates two things: first, that the cleanup memory works and does in fact

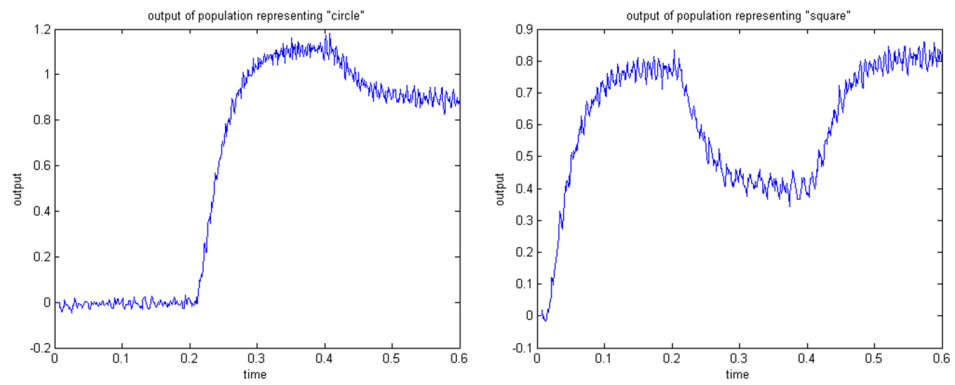

Figure 5.9: Output from populations tuned to respond to the presence of "circle" (left) and "square" (right) vectors, while the module solves the matrix shown in Figure 5.7.

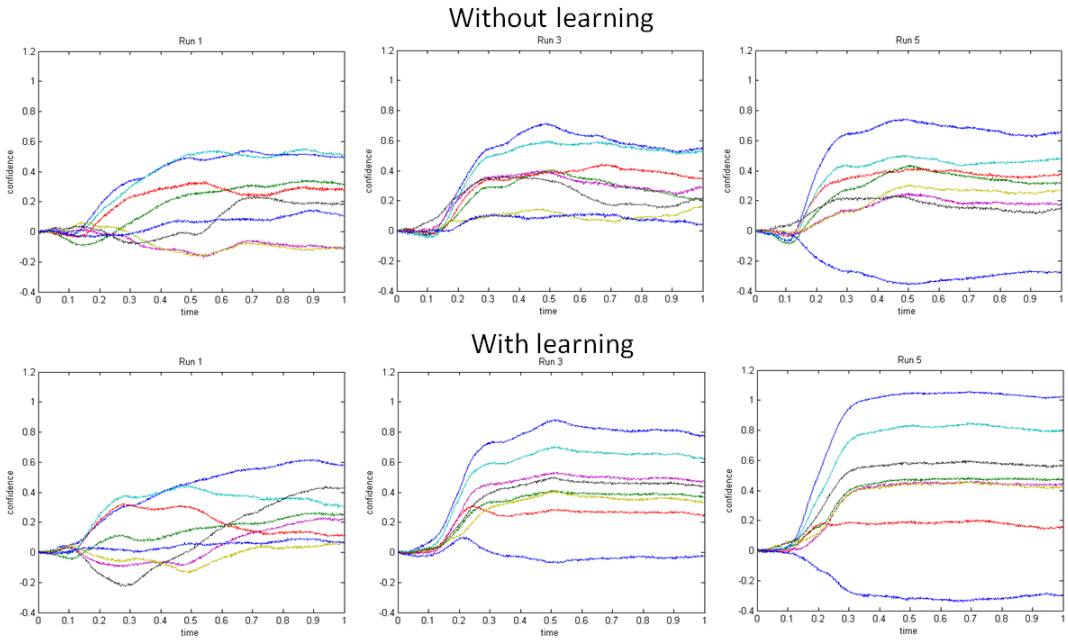

Figure 5.10: A demonstration of the effects of cleanup memory. The model was run on five matrices that appeared different but required the same underlying rules to solve. Each graph shows the system's confidence in each of the eight possible answers over time (similar to Figure 5.1). On the top are the system's results on the first, third, and fifth matrix with no cleanup memory, and on the bottom are the results on the same matrices if the system is allowed to build up a cleanup memory based on calculations in the previous matrices.

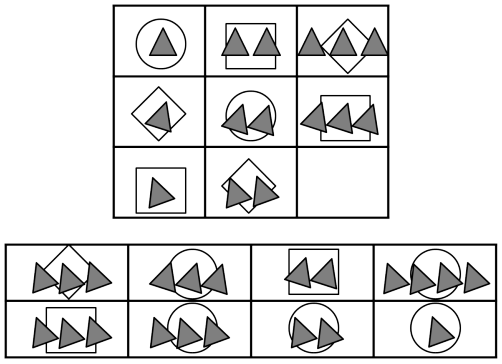

Figure 5.11: A matrix involving multiple rule types.

make the output more accurate, and second, that the learning works and is able to improve the quality of results over time based on past calculations. These results also reflect the observed impact of matrix ordering on error rates, discussed in Section 3.2; subjects are more likely to get an item correct if they have correctly solved similar items (i.e. had a chance to build up their cleanup memory) in the past. In terms of general intelligence, this demonstrates the importance of learning both in the RPM and in complex tasks in general. Being able to figure out the answer to the current problem is important, but that task is made far easier if we can take what we have learned on previous problems and apply it to the current situation.

5.1.5 Controller

The matrices used as examples in the previous sections have all been quite simple. This is because the complex matrices all require the controller, due to their use of mixed rule types. The controller does not produce much in the way of results itself, since it is primarily just assembling together the results from the neural modules, but we can demonstrate that it is able to deal with these complicated matrices. We will also analyze in more detail *how* the controller deals with these matrices, as it helps elucidate the role played by the controller.

We will run the controller on the matrix shown in Figure 5.11. The final result is shown in Figure 5.12, demonstrating that the controller is able to find the correct answer. The process that allowed the controller to arrive at this answer is seen in its output:

```
(running figuresolver)
figure score = 0.0 (0.3470201939344406)
---------feature 0-----------
(running figuresolver)
```

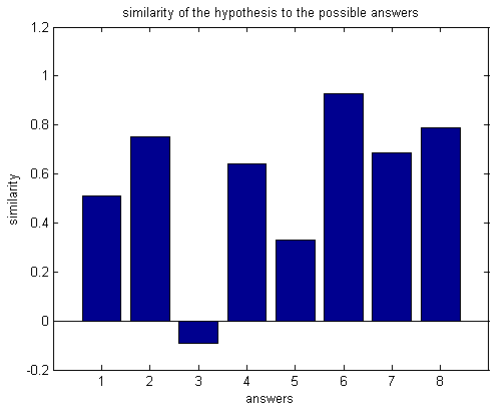

Figure 5.12: The controller's final output, demonstrating that it thinks number six is most likely to be correct.

```
figure score = 0.0 (-0.07407767176628113)
(running sequencesolver)
sequence score = 0.3786676619563173
(running setsolver)
set score = 0.9681230620643543
using setsolver answer
---------feature 1-----------
(running figuresolver)
figure score = 0.0 (0.5886807680130005)
(running sequencesolver)
sequence score = 0.7294316297795124
(running setsolver)
set score = 0.6549323878505379
--attribute shape--
(running sequencesolver)
sequence score = 0.8733839373210761
using sequencesolver answer
--attribute angle--
(running sequencesolver)
sequence score = 0.9285401068071122
using sequencesolver answer
--attribute number--
(running sequencesolver)
sequence score = 0.6866626666469495
```

```
(running setsolver)
set score = 0.9688592307841991
using setsolver answer
hypothesis: ...
similarities: [0.51, 0.75, -0.09, 0.64, 0.32, 0.92, 0.68, 0.79]
```

First we see the controller running the figure module. This means that it takes the matrix data, passes it to the figure module, and then waits until the module returns a rule and a hypothesis for the blank cell. The "figure score =" line is the controller's estimation of how well the rule generated by the module matches the known matrix data. The threshold of when we will consider the rule a success is a parameter of the model; in this case it was set to 0.8. The controller decides that the rule was not very good, and so the model has correctly failed to find a rule for the whole matrix (since there is no single rule that describes all the changes in this matrix).[1]

The next step is to break the matrix down into features (recall the different levels of representation described in Section 4.4.6). The first feature is the background shapes, which are governed by a set type rule. As we would expect then, the figure and sequence solver both fail to find a correct rule, but the set solver finds a rule that is a good match for the matrix data. The next feature is the foreground triangles. In this case none of the modules find a good rule at the feature level, so the feature is broken down into its component attributes: the shape, angle, and number of foreground objects. Shape and angle are solved by the sequence solver module (lack of change is a type of sequence, defined by the identity transformation), but we see something interesting in the number attribute.[2] Even though the number of triangles forms a sequence, in this case the sequence module failed to find that rule. However, the set module *was* able to find a correct rule. This demonstrates the superclass nature of the set based strategy, as discussed in Section 4.4.3. It also demonstrates the model's ability to respond gracefully to failure, attempting to find other routes to the solution. The "hypothesis" line is a collection of the hypotheses generated by the neural modules that were able to find a correct rule, so it contains four vectors: one hypothesis for the first feature, and one for each of the attributes in the second

[1] Note that only the figure module gets a chance to find a rule for the whole matrix. This is not due to any innate limitation of the sequence or set solver module, but is an external constraint we have imposed in order to make the model more realistic. Our hypothesis is that only this very simple figure based reasoning is able to operate on a whole-matrix, "gestalt" level—the other modules require the matrix to be broken down into more specific components.

[2] At the attribute level, only the sequence and set modules attempt to find rules. Again this is an external constraint we have imposed in order to make the model more realistic. Since the figure module is meant to represent the lower-level visual type reasoning of determining whether the same features are present in two cells, it does not make much sense to apply it at the attribute level. What would it mean to say that a 90 degree orientation is present in both cells? It only seems realistic to make those kinds of judgements on unified objects, which only exist at the feature level and above.

feature (it is a long vector and so we edited it out). The "similarities" line indicates how similar each of the eight possible answers are to that hypothesis. The controller correctly picks the most similar response, number six.

This illustrates the three main roles of the controller: managing the input and output of the neural modules, evaluating their responses, and selecting a final answer. It also demonstrates the many parameters that exist in a subject's strategy: What order should the levels be tried in (whole matrix and down, or attribute level and up)? Which modules should be employed at each level? What order should the modules be tried in? How good does a rule have to be before we consider it a success? And given multiple rules, how should they be combined in order to select an overall response? These questions only become apparent when we attempt to build a unified model, they are not inherent in the neural components themselves. In previous sections we focused on why analyzing these abilities at the neural level can provide unique insight into the questions of general intelligence, but this section reminds us that intelligence is more than the functioning of a specific cognitive ability, it is also about how we put those abilities together to accomplish the task at hand.

5.2 Empirical results

One of the great advantages of creating biologically realistic models is that we can then use those models to achieve insight into human cognition. For example, one of the most important questions in the study of general intelligence is the study of individual differences; why are some people smarter than others? We can use our model to help answer this question by examining what factors make the model "smarter"—in other words, how do manipulations of the model lead to increased or decreased performance on the RPM?

One of the main factors we discussed in Section 2.1.1 was brain size; subjects with more hardware tend to perform better. We can model this by increasing or decreasing the number of neurons used in our neural modules. Having more neurons allows us to work with higher dimensional vectors, and we have seen in the previous discussions that the dimensionality of the vectors is key to the accuracy of the VSA operations. Therefore we would expect that increasing the number of neurons (and dimensionality of the vectors proportionately) would increase the performance of the model. We test this hypothesis by running the model repeatedly on a subset of Raven's matrices while varying the dimension of the vectors used in the system. The randomly generated components of the model, such as neuron parameters and VSA vocabulary, will cause variability in any given run, so we run the model many times at each dimension and collect aggregate results. The set of matrices we use in this test (and in all future results in this section) is a sample of items from the RPM that can be solved by the sequence and set type of reasoning. We do not

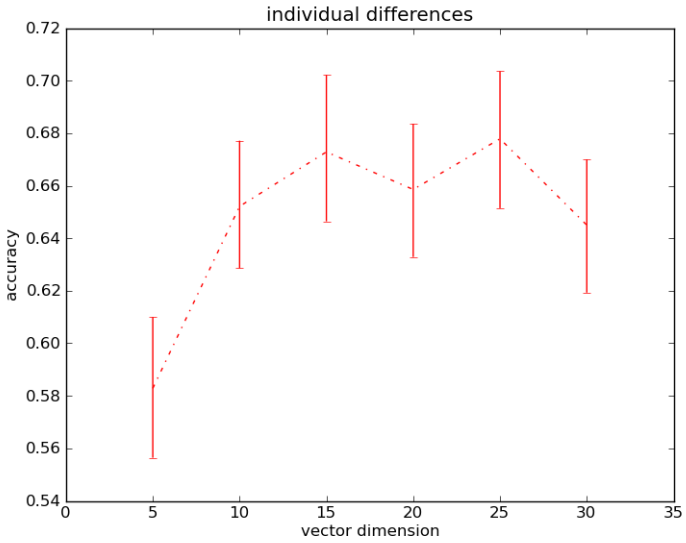

Figure 5.13: Effect of varying the dimensionality of the VSA vectors (and proportionally manipulating the number of neurons) on average accuracy on the RPM

include figure type problems, because as discussed in Section 4.4.4 that module is not yet working as intended, and so will not provide us with meaningful results.

As we can see in Figure 5.13, there is a clear difference in performance depending on the dimensionality of the vectors the system is able to use. However, we also see that there is a ceiling effect; there is a good improvement from about 5 to 15 dimensions, but after that we see no real change. This is a prediction of the model, which we could test by plotting brain volume in the areas associated with this model against RPM score.

It is worth noting at this point that what we are interested in here is the relative performance of individuals across these manipulations. It would be a mistake to try to draw conclusions based on the absolute performance, for example to say that "subjects who can handle 30 dimensional vectors can expect 65% accuracy on the RPM", or to say "the model performs at the same level as an average subject". The problem is that there are too many variables not captured by the model. For example, correspondence finding and visual parsing are important factors in problem difficulty, as discussed in Section 3.2, but are completely absent from the model; therefore they will have a significant impact on human subjects' performance, but no impact on the model's performance. That is why we cannot make direct performance comparisons. However, what we can do is speak about relative performance changes and differences. For example, we can say that human subjects with more neurons in the brain areas involved in solving the RPM should see relatively

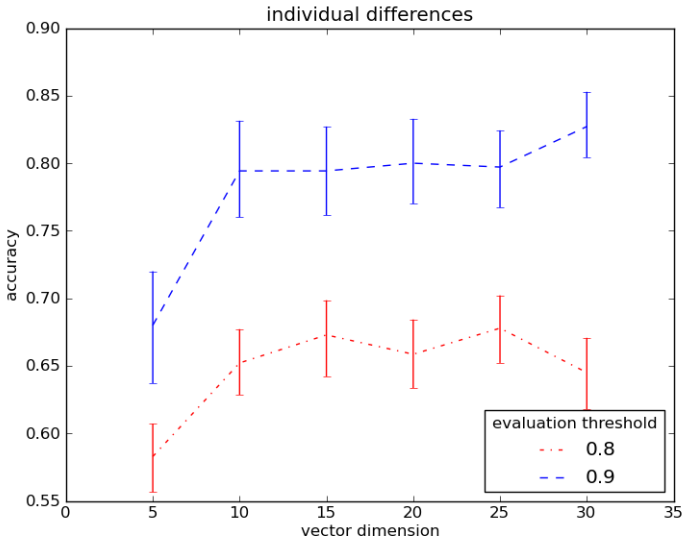

Figure 5.14: Effect of varying the quality of rule that will be accepted by the evaluation component of the controller.

enhanced performance. We can also predict a ceiling effect to this enhancement. These are true despite the fact that we have not modelled every aspect of human cognition, because we are only making claims that involve changes in the aspects that we *have* modelled.

Having examined the low-level variable of neuron number, we will now look at how manipulating the high-level performance of the model can affect performance. One of the most important parameters in the model is in the rule evaluation component, where the system must decide whether a successful rule has been returned by the neural modules. We can map this on to how conservative a subject is; some will be less cautious, just taking the first rule that seems decent and selecting an answer, while others will be more careful, making sure that their rule is a good fit before moving on. What effect will this have on their performance? We can model this by raising or lowering the threshold at which the model will accept a rule. The results of this manipulation are seen in Figure 5.14. Note that we see the same pattern of improvement due to increased number of neurons, but everything has been shifted upwards. This demonstrates the interesting interactions between low-level and high-level variables: the subject's basic pattern of ability is still determined by neural capacity, but they can change what they do with that capacity by employing different strategies to accomplish their goal.

Another high-level component is the response selection. We saw in Section 2.1.2 that the ability to choose between competing alternatives is an important aspect of intelligent

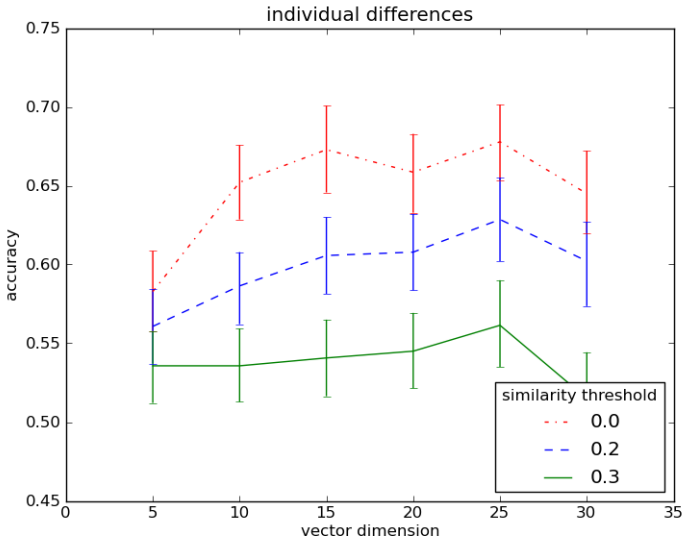

Figure 5.15: Effect of varying the accuracy with which the system is able to distinguish competing responses.

behaviour. We can model this by manipulating the threshold that the model will require between the winning response and the next best alternative. A smaller threshold indicates more accurate selection—the subject is able to find the correct response despite heavily competing alternatives. The effect of this manipulation is shown in Figure 5.15. Interestingly, in this case we see a completely different effect than with the previous manipulation. It seems that the less accurate the subject's response selection is, the less they benefit from increased vector dimension/neuron number. This makes sense when we consider that the purpose of increasing the vector dimension is to increase the accuracy of the calculations; if the subject cannot accurately distinguish the responses, then the subtle accuracy changes associated with increased vector dimension will be lost on them. This reveals the complex nature of individual differences; first, not all causes of individual differences will affect performance in the same way, and second, the causes may have complex interactions with one another. Again this emphasizes the importance of modelling; these effects cannot be understood by studying one factor in isolation, we must also study how the factors interact.

5.3 Summary of contributions

The primary contribution of this thesis is a working, realistic neural model of the cognitive processes involved in rule generation. In Section 5.1 we demonstrated that our model successfully recreates these processes; it found rules that enabled it to carry out the three different types of reasoning identified in Section 4.4.1, and we demonstrated the generality of those rules. We also saw that the model was able to solve complex matrices involving combinations of rule types, and that it was able to use cleanup memory to improve its performance over time.

The usefulness of this model was demonstrated in Section 5.2. We focused on one area of potential investigation, individual differences. First we saw that by manipulating the number of neurons used in the model we can manipulate its overall accuracy, and developed the novel prediction that there would be a ceiling effect on the benefits of increased neural capacity in humans. Second, by manipulating a parameter of the model representing subjects' rule-evaluation criteria, we were able to shift overall performance up and down while preserving the performance differences caused by neural capacity. Third, by manipulating the precision of answer selection, we were able to modify performance by modulating the benefits gained by increased number of neurons. These results demonstrate the complex interactions between low and high level variables that combine to create individual differences.

Section 5.2 addressed only some of the ways in which we might investigate individual differences, and individual differences is just one of the complex factors of human cognition to which we might apply the model. These results are intended to demonstrate the value of this model, and its ability to provide insight into the underlying aspects of general intelligence. It is only through modelling the complex interactions between components that these insights become apparent, and it is thanks to the biological constraints applied at the global and neural level that we are able to apply these insights to human cognition.

Chapter 6

Conclusion

6.1 Future work

There are many interesting directions this research could take in the future. First and foremost, we would like to get the figure module working more reliably. We will then have a system that can perform well on the entire RPM, allowing us to investigate a broader range of problem types and reasoning abilities. As discussed in Section 4.4.4, we are confident that achieving this goal is primarily a matter of increasing our available computing power rather than refinements to the theory, and so we anticipate success in this direction soon.

The second area we would like to pursue is increasing the range of our empirical results. There are a number of other interesting questions that we could answer using this model. First, an important technique used to investigate human intelligence is lesion studies; if a certain area of a person's brain is damaged, how does that affect their intelligence? Since we established a mapping between the components of our model and regions in the brain, we should be able to mimic the effects of lesions and see how that affects the model's performance. Second, we could investigate the affects of "ageing" the model. We discussed in Section 2.2.1 how throughout our lifetime we steadily lose neurons; we could model this by randomly eliminating neurons from our system, and examining how robust it is to this damage. Third, we could add more complex behaviour to the controller, so that it is able to use different high-level strategies (such as the response construction versus response elimination discussed in Section 3.3). Fourth, we would like to analyze the interaction of individual differences with learning. The results shown in Section 5.2 all had learning turned off, so that we could see the effect of the chosen manipulation in isolation. However, there should be interesting interactions between these components; for example, if subjects have increased neural capacity and are able to calculate rules more accurately, that will

affect the contents of cleanup memory and feed back into future calculations. These are all important questions in the study of general intelligence, and it will be interesting to see what insight this model can provide.

Third, we would like to incorporate more realistic visual processing into the model. Visual processing is an important aspect of performance on the RPM and almost all other complex tasks. Its absence from the current model was a necessary limitation in order to focus our attention on the rule generation system, but it is an unfortunate blank spot in its explanatory power. Hopefully now that the rule generation system is better established, it will be possible to go back and incorporate these processes into the model in a more realistic way.

Finally, we would like to validate the predictions of this model experimentally. For example, our model predicts that there is a ceiling effect on the interaction between number of neurons and RPM performance. If we compare the relationship between grey matter volume and RPM score in human subjects, will we see the same thing? In addition, in developing our model we came up with a new method of understanding the different reasoning types involved in the RPM—the figure, sequence, and set approaches. Are these different techniques seen in human subjects? We could also look in more detail at the network and neural characteristics predicted by our model, and compare them to the neurophysiology in our suggested brain regions. For example, the integrator that is central to the sequence solver module is much more accurate if it is made up of neurons with long post synaptic currents. Do we find these kinds of neurons in the premotor cortex, the region in which we suggest the sequence module lies?

The broad trend of this future work is to continue to use the model to investigate the underlying factors of general intelligence, improving the model's capabilities as necessary in order to do so.

6.2 Closing remarks

In Section 4.1 we described three main motivations for our work: to understand rule generation in the Raven's Progressive Matrices test, to use that understanding to better understand the broader principles of general intelligence, and to illustrate the contributions of neural modelling to this investigation. We are now in a position to analyze our success in these endeavours.

In respect to the first goal, we have created a biologically plausible model that can generate the rules needed to solve Raven's Progressive Matrices. This is valuable in itself, as previously this rule generation ability was largely a mystery. We have also thereby removed the reliance of previous theories on a library of known rules. This is useful both

for researchers studying the RPM and for experimentalists using the RPM, as it resolves the conflict between the observed fluid nature of RPM reasoning and the crystallized nature of relying on pre-existing rules. We have also shown that we can use this model to help explain experimental results on the RPM, such as the range of individual differences. Thus this research has made contributions to understanding the abilities involved in Raven's Progressive Matrices.

We can also extract many of the insights gained while studying the RPM to a broader understanding of intelligence in general. First of all, intelligence cannot be attributed to one component. One component can be crucial to intelligence, in that without it we cannot act intelligently, but that does not mean that by understanding that component we understand general intelligence. Take the response selection of the controller as an example. Intelligence researchers have suggested that the ability to choose between competing alternatives is key to intelligence, and we can see in the model that this is true—if the system cannot pick a response, it will not do well on the RPM. However, one could never understand the model we have developed, or the overall behaviour of the system, by analyzing that one aspect of the controller.

On the other hand, intelligence cannot be attributed to a list of components either. This is another approach that is sometimes taken to understanding intelligence, to try to enumerate all the pieces involved and hope that when the list is complete intelligence will be understood. However, it is relatively easy to provide a complete itemization of all the components of our model, but even with such a description it would be extremely difficult to understand its overall behaviour. The problem is that the pieces do not operate in isolation, and in attempting to dissect them into self-contained packages we destroy the information that defines them. Intelligence arises out of the complex interactions between these components, and it is only when placed in this context that the role of the components becomes clear. This is where modelling comes in to play; it forces us to try to put these components together, thereby adding both to our understanding of the individual components and to our overall understanding of the system's behaviour.

We hope that this is what we have accomplished with our work here. By developing a model that successfully recreates important aspects of human cognition on one of the best tests of general intelligence, we are forced to analyze not only what elements are involved but how they interact to produce such complex behaviour. These efforts have already provided interesting insights, and leave us with many new directions to pursue in the future.

References

Alexander, G., DeLong, M., and Strick, P. (1986). Parallel organization of functionally segregated circuits linking basal ganglia and cortex. *Annual Review of Neuroscience*, 9:357–381. 63

Bartels, M., Rietveld, M. J. H., Van Baal, G. C. M., and Boomsma, D. I. (2002). Genetic and environmental influences on the development of intelligence. *Behavior Genetics*, 32(4):237–49. 18

Bethell-Fox, C., Lohman, D., and Snow, R. (1984). Adaptive reasoning: Componential and eye movement analysis of geometric analogy performance. *Intelligence*, 8(3):205–238. 29

Binet, A. (1905). New Methods for the Diagnosis of the Intellectual Level of Subnormals. In Kite, E., editor, *The Development of Intelligence in Children*. Publications of the Training School at Vineland, Vineland. 4, 5

Bors, D. (2003). The effect of practice on Raven's Advanced Progressive Matrices. *Learning and Individual Differences*, 13(4):291–312. 33

Bors, D. and Stokes, T. (1998). Raven's Advanced Progressive Matrices: Norms for First-Year University Students and the Development of a Short Form. *Educational and Psychological Measurement*, 58(3):382–398. 6

Broca, P. (1861). *Sur le volume et la forme du cerveau suivant les individus et suivant les races*. Typographie Hennuyer, Paris. 10

Brouwers, S. A., Van De Vijver, F., and Van Hemert, D. A. (2009). Variation in Raven's Progressive Matrices scores across time and place. *Learning and Individual Differences*, 19(3):330–338. 6, 7

Carpenter, P., Just, M., and Shell, P. (1990). What one intelligence test measures: a theoretical account of the processing in the Raven Progressive Matrices Test. *Psychological Review*, 97(3):404–31. 7, 14, 26, 27, 32, 36, 58, 60

Cattell, R. (1940). A culture-free intelligence test. *The Journal of Educational Psychology*, 31(3):161–179. 6

Cattell, R. (1987). *Intelligence: Its structure, growth, and action*. Elsevier Science Publishing Company, New York. 2

Christoff, K., Prabhakaran, V., Dorfman, J., Zhao, Z., Kroger, J., Holyoak, K., and Gabrieli, J. D. E. (2001). Rostrolateral prefrontal cortex involvement in relational integration during reasoning. *NeuroImage*, 14(5):1136–49. 26

Christoff, K., Ream, J. M., Geddes, L. P. T., and Gabrieli, J. D. E. (2003). Evaluating self-generated information: anterior prefrontal contributions to human cognition. *Behavioral Neuroscience*, 117(6):1161–8. 63

Conway, A. R. a., Kane, M. J., and Engle, R. W. (2003). Working memory capacity and its relation to general intelligence. *Trends in Cognitive Sciences*, 7(12):547–552. 14

Dawson, M., Soulières, I., Gernsbacher, M. A., and Mottron, L. (2007). The level and nature of autistic intelligence. *Psychological Science*, 18(8):657–62. 28

Deary, I. J., Penke, L., and Johnson, W. (2010). The neuroscience of human intelligence differences. *Nature Reviews Neuroscience*, 11(3):201–11. 18

Deary, I. J., Whalley, L., Lemmon, H., Crawford, J., and Starr, J. (2000). The Stability of Individual Differences in Mental Ability from Childhood to Old Age: Follow-up of the 1932 Scottish Mental Survey. *Intelligence*, 28(1):49–55. 17

DeShon, R., Chan, D., and Weissbein, D. (1995). Verbal overshadowing effects on Ravens Advanced Progressive Matrices: evidence for multidimensional performance determinants. *Intelligence*, 21:135–155. 26, 46

D'Esposito, M., Detre, J., Alsop, D., Shin, R., Atlas, S., and Grossman, M. (1995). The neural basis of the central executive system of working memory. *Nature*, 378(6554):279–281. 62

Duncan, J. (2001). An adaptive coding model of neural function in prefrontal cortex. *Nature Reviews Neuroscience*, 2(11):820–9. 64

Eliasmith, C. (2005). A unified approach to building and controlling spiking attractor networks. *Neural Computation*, 17(6):1276–314. 49

Eliasmith, C. and Anderson, C. (2003). *Neural engineering: Computation, representation, and dynamics in neurobiological systems*. MIT Press, Cambridge. 36

Evans, T. (1968). *A program for the solution of a class of geometric-analogy intelligence-test questions*. MIT Press, Cambridge. 30, 31, 36

Flynn, J. R. (2009). Requiem for nutrition as the cause of IQ gains: Raven's gains in Britain 1938-2008. *Economics and Human Biology*, 7(1):18–27. 19, 20

Fry, A. and Hale, S. (1996). Processing speed, working memory, and fluid intelligence: Evidence for a developmental cascade. *Psychological Science*, 7:237–241. 14, 15

Galton, F. (1888). Head Growth in Students at the University of Cambridge. *Nature*, 38:14–15. 10, 11

Galton, F. (1892). *Hereditary genius*. Macmillan and Co., London. 3, 4, 6, 17, 18

Gayler, R. (2003). Vector Symbolic Architectures answer Jackendoff's challenges for cognitive neuroscience. In Slezak, P., editor, *ICCS/ASCS International Conference on Cognitive Science*, pages 133–138, Sydney. University of New South Wales. 36

Gentner, D. (1983). Structure-Mapping: A Theoretical Framework for Analogy. *Cognitive Science*, 7(2):155–170. 34

Golde, M., von Cramon, D. Y., and Schubotz, R. I. (2010). Differential role of anterior prefrontal and premotor cortex in the processing of relational information. *NeuroImage*, 49(3):2890–900. 46, 62

Gottfredson, L. (1997). Mainstream science on intelligence: An editorial with 52 signatories, history, and bibliography. *Intelligence*, 24(1):13–23. 1

Gould, E. (2007). How widespread is adult neurogenesis in mammals? *Nature Reviews Neuroscience*, 8(6):481–8. 16

Gray, J. R., Chabris, C. F., and Braver, T. S. (2003). Neural mechanisms of general fluid intelligence. *Nature Neuroscience*, 6(3):316–22. 3, 15, 22

Gray, J. R. and Thompson, P. M. (2004). Neurobiology of intelligence: science and ethics. *Nature Reviews Neuroscience*, 5(6):471–82. 18, 19

Haier, R., Siegel, B., Maclachlan, A., Soderling, E., Lottenberg, S., and Buchsbaum, M. (1992). Regional glucose metabolic changes after learning a complex visuospatial/motor task: a positron emission tomographic study. *Brain Research*, 570(1-2):134–143. 12

Haier, R., Siegel, B., Nuechterlein, K., Hazlett, E., Wu, J., Paek, J., Browning, H., and Buchsbaum, M. (1988). Cortical glucose metabolic rate correlates of abstract reasoning and attention. *Intelligence*, 12:199–217. 12, 62

Hall, R. (1989). Computational approaches to analogical reasoning: A comparative analysis. *Artificial Intelligence*, 39:39–120. 30

Haverty, L. (2000). Solving inductive reasoning problems in mathematics: not-so-trivial pursuit. *Cognitive Science*, 24(2):249–298. 29

Hazy, T. E., Frank, M. J., and O'Reilly, R. C. (2007). Towards an executive without a homunculus: computational models of the prefrontal cortex/basal ganglia system. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, 362(1485):1601–13. 63

Hunt, E. (1973). Quote the Raven? Nevermore! In Gregg, L., editor, *Knowledge and Cognition*, pages 129–157. Lawrence Erlbaum Associates, Potomac. 31, 32, 36, 45, 57

Jung, R. E. and Haier, R. (2007). The Parieto-Frontal Integration Theory (P-FIT) of intelligence: converging neuroimaging evidence. *Behavioral and Brain Sciences*, 30(2):135–54; discussion 154–87. 61

Kail, R. (2000). Speed of Information Processing: Developmental Change and Links to Intelligence. *Journal of School Psychology*, 38(1):51–61. 14

Kane, M. J. and Engle, R. W. (2002). The role of prefrontal cortex in working-memory capacity, executive attention, and general fluid intelligence: an individual-differences perspective. *Psychonomic Bulletin and Review*, 9(4):637–671. 22, 63

Kaufman, A., Reynolds, C., and McLean, J. (1989). Age and WAIS-R intelligence in a national sample of adults in the 20- to 74-year age range: A cross-sectional analysis with educational level controlled. *Intelligence*, 13(3):235–253. 17, 19

Kirby, J. (1983). Effects of strategy training on Progressive Matrices performance. *Contemporary Educational Psychology*, 8(2):127–140. 27, 45, 51, 58

Kretchmer, N., Beard, J. L., and Carlson, S. (1996). The role of nutrition in the development of normal cognition. *The American Journal of Clinical Nutrition*, 63(6):997S–1001S. 20

Kyllonen, P. C. and Christal, R. E. (1990). Reasoning ability is (little more than) working-memory capacity?! *Intelligence*, 14(4):389–433. 13, 27

Li, Y., Liu, Y., Li, J., Qin, W., Li, K., Yu, C., and Jiang, T. (2009). Brain anatomical network and intelligence. *PLoS Computational Biology*, 5(5):e1000395. 12

Lovett, A., Forbus, K., and Usher, J. (2007). Analogy with qualitative spatial representations can simulate solving Ravens Progressive Matrices. In *Proceedings from the 29th Annual Conference of the Cognitive Science Society*, number 1, pages 449–454. 33

Lovett, A., Forbus, K., and Usher, J. (2010). A structure-mapping model of Raven's Progressive Matrices. In Ohlsson, S. and Catrambone, R., editors, *Proceedings of the 32nd Annual Conference of the Cognitive Science Society*, pages 2761–2766, Austin. Cognitive Science Society. 33, 34, 36

Mackintosh, N. and Bennett, E. (2005). What do Raven's Matrices measure? An analysis in terms of sex differences. *Intelligence*, 33(6):663–674. 26

Marshalek, B., Lohman, D., and Snow, R. (1983). The complexity continuum in the radex and hierarchical models of intelligence. *Intelligence*, 7(2):107–127. ix, 8, 22, 23

Matzel, L. D. and Kolata, S. (2010). Selective attention, working memory, and animal intelligence. *Neuroscience and Biobehavioral Reviews*, 34(1):23–30. 15

Melrose, R. J., Poulin, R. M., and Stern, C. E. (2007). An fMRI investigation of the role of the basal ganglia in reasoning. *Brain Research*, 1142:146–58. 63

Meo, M., Roberts, M., and Marucci, F. (2007). Element salience as a predictor of item difficulty for Raven's Progressive Matrices. *Intelligence*, 35(4):359–368. 27

Narr, K. L., Woods, R. P., Thompson, P. M., Szeszko, P., Robinson, D., Dimtcheva, T., Gurbani, M., Toga, A. W., and Bilder, R. M. (2007). Relationships between IQ and regional cortical gray matter thickness in healthy adults. *Cerebral Cortex*, 17(9):2163–71. 11

Neisser, U., Boodoo, G., Bouchard, T. J., Boykin, A. W., Brody, N., Ceci, S. J., Halpern, D. F., Loehlin, J. C., Perloff, R., Sternberg, R. J., and Urbina, S. (1996). Intelligence: Knowns and unknowns. *American Psychologist*, 51(2):77–101. 4, 19, 20

Neubauer, A. C. and Fink, A. (2003). Fluid intelligence and neural efficiency: effects of task complexity and sex. *Personality and Individual Differences*, 35(4):811–827. 12

Neumann, J. (2001). *Holistic Processing of Hierarchical Structures in Connectionist Networks*. Unpublished doctoral thesis, University of Edinburgh, Edinburgh. 47

Pakkenberg, B. and Gundersen, H. J. (1997). Neocortical neuron number in humans: Effect of sex and age. *The Journal of Comparative Neurology*, 384(2):312–20. 16

Perfetti, B., Saggino, A., Ferretti, A., Caulo, M., Romani, G. L., and Onofrj, M. (2009). Differential patterns of cortical activation as a function of fluid reasoning complexity. *Human Brain Mapping*, 30(2):497–510. 3, 22

Pfefferbaum, A., Mathalon, D., Sullivan, E., Rawles, J., Zipursky, R., and Lim, K. (1994). A quantitative magnetic resonance imaging study of changes in brain morphology from infancy to late adulthood. *Archives of Neurology*, 51(9):874–887. 16

Plate, T. (2003). *Holographic Reduced Representations*. CLSI Publications, Stanford. 36, 37, 38, 49

Prabhakaran, V., Smith, J., Desmond, J., Glover, G., and Gabrieli, J. D. E. (1997). Neural substrates of fluid reasoning: an fMRI study of neocortical activation during performance of the Raven's Progressive Matrices Test. *Cognitive Psychology*, 33:43–63. 28, 46, 62

Primi, R. (2002). Complexity of geometric inductive reasoning tasks: Contribution to the understanding of fluid intelligence. *Intelligence*, 30(1):41–70. 27

Raven, J. (1989). The Raven Progressive Matrices: A review of national norming studies and ethnic and socioeconomic variation within the United States. *Journal of Educational Measurement*, 26(1):1–16. 6

Raven, J. C. (1936). *Mental tests used in genetic studies: The performance of related individuals on tests mainly educative and mainly reproductive*. Master's thesis, University of London. 6

Redgrave, P., Prescott, T., and Gurney, K. (1999). The basal ganglia: a vertebrate solution to the selection problem? *Neuroscience*, 89(4):1009–1024. 63

Resnick, S. M., Pham, D. L., Kraut, M. A., Zonderman, A. B., and Davatzikos, C. (2003). Longitudinal magnetic resonance imaging studies of older adults: A shrinking brain. *Journal of Neuroscience*, 23(8):3295–3301. 17

Rushton, J. P. and Ankney, C. D. (2009). Whole brain size and general mental ability: a review. *The International Journal of Neuroscience*, 119(5):691–731. 11

Smith, E. E. and Jonides, J. (1999). Storage and Executive Processes in the Frontal Lobes. *Science*, 283(5408):1657–1661. 14

Soulières, I., Dawson, M., Samson, F., Barbeau, E. B., Sahyoun, C. P., Strangman, G. E., Zeffiro, T. A., and Mottron, L. (2009). Enhanced visual processing contributes to matrix reasoning in autism. *Human Brain Mapping*, 30(12):4082–107. 28

Spearman, C. (1904). "General Intelligence," Objectively Determined and Measured. *The American Journal of Psychology*, 15(2):201. 2, 3

Sporns, O. and Zwi, J. D. (2004). The small world of the cerebral cortex. *Neuroinformatics*, 2(2):145–62. 11

Staff, R. T., Murray, A. D., Deary, I. J., and Whalley, L. (2006). Generality and specificity in cognitive aging: A Volumetric brain analysis. *NeuroImage*, 30:1433–1440. 16

Stewart, T. C., Choo, X., and Eliasmith, C. (2010). Dynamic Behaviour of a Spiking Model of Action Selection in the Basal Ganglia. In Ohlsson, S. and Catrambone, R., editors, *Proceedings of the 32nd Annual Conference of the Cognitive Science Society*, pages 235–240, Austin. Cognitive Science Society. 44, 58, 63

Stewart, T. C., Tang, Y., and Eliasmith, C. (2009a). A biologically realistic cleanup memory: Autoassociation in spiking neurons. In Howes, A., Peebles, D., and Cooper, R., editors, *9th International Conference on Cognitive Modelling*, pages 128–133, Manchester. ICCM2009. 54

Stewart, T. C., Tripp, B., and Eliasmith, C. (2009b). Python scripting in the nengo simulator. *Frontiers in Neuroinformatics*, 3(March):7. 43

Stroop, J. (1935). Studies of interference in serial verbal reactions. *Journal of Experimental Psychology*, 18(6):643–662. 15

Süß, H.-M., Oberauer, K., Wittmann, W., Wilhelm, O., and Schulze, R. (2002). Working-memory capacity explains reasoning abilityand a little bit more. *Intelligence*, 30(3):261–288. 14

Tamez, E., Myerson, J., and Hale, S. (2008). Learning, working memory, and intelligence revisited. *Behavioural Processes*, 78(2):240–5. 28

Thompson, P. M., Cannon, T., Narr, K. L., van Erp, T., Poutanen, V. P., Huttunen, M., Lönnqvist, J., Standertskjöld-Nordenstam, C. G., Kaprio, J., Khaledy, M., Dail, R., Zoumalan, C. I., and Toga, A. W. (2001). Genetic influences on brain structure. *Nature Neuroscience*, 4(12):1253–8. 18

Thomson, G. H. (1939). *The factorial analysis of human ability*. Houghton Mifflin, Oxford. 2

Turkheimer, E., Haley, A., Waldron, M., D'Onofrio, B., and Gottesman, I. I. (2003). Socioeconomic status modifies heritability of IQ in young children. *Psychological Science*, 14(6):623–8. 20

Turner, M. (1989). Is working memory capacity task dependent? *Journal of Memory and Language*, 28(2):127–154. 13

Unsworth, N. and Engle, R. W. (2005). Working memory capacity and fluid abilities: Examining the correlation between Operation Span and Raven. *Intelligence*, 33(1):67–81. 14, 28

Van De Vijver, F. (1997). Meta-Analysis of Cross-Cultural Comparisons of Cognitive Test Performance. *Journal of Cross-Cultural Psychology*, 28(6):678–709. 6, 22

Verguts, T. and De Boeck, P. (2002). The induction of solution rules in Ravens Progressive Matrices Test. *European Journal of Cognitive Psychology*, 14:521–547. 27, 28

Vigneau, F. and Bors, D. (2008). The quest for item types based on information processing: An analysis of Raven's Advanced Progressive Matrices, with a consideration of gender differences. *Intelligence*, 36(6):702–710. 46

Vigneau, F., Caissie, A., and Bors, D. (2006). Eye-movement analysis demonstrates strategic influences on intelligence. *Intelligence*, 34(3):261–272. 28, 29

Williams, B. and Pearlberg, S. (2006). Learning of three-term contingencies correlates with Raven scores, but not with measures of cognitive processing. *Intelligence*, 34(2):177–191. 28

Winship, C. and Korenman, S. (1997). Does Staying in School Make You Smarter? The Effect of Education on IQ in The Bell Curve. In Devlin, B., Fienberg, S., Resnick, D., and Roeder, K., editors, *Intelligence, Genes, & Success: Scientists Respond to The Bell Curve*, chapter 10, pages 215–234. Springer-Verlag, New York. 19

Zhao, C., Deng, W., and Gage, F. H. (2008). Mechanisms and functional implications of adult neurogenesis. *Cell*, 132(4):645–60. 16