Journal of Artificial General Intelligence 4(1) 1–30, 2013
DOI: 10.2478/jagi-2013-0001

Submitted 2011-06-15
Accepted 2013-07-06

Evolving Non-Dominated Parameter Sets for Computational Models from Multiple Experiments

Peter C. R. Lane PETER.LANE@BCS.ORG.UK
School of Computer Science
University of Hertfordshire
Hatfield AL10 9AB, UK

Fernand Gobet FERNAND.GOBET@BRUNEL.AC.UK
School of Social Sciences
Brunel University
Uxbridge UB8 3PH, UK

Editor: Christian Lebiere

Abstract

Creating robust, reproducible and optimal computational models is a key challenge for theorists in many sciences. Psychology and cognitive science face particular challenges as large amounts of data are collected and many models are not amenable to analytical techniques for calculating parameter sets. Particular problems are to locate the full range of acceptable model parameters for a given dataset, and to confirm the consistency of model parameters across different datasets. Resolving these problems will provide a better understanding of the behaviour of computational models, and so support the development of general and robust models. In this article, we address these problems using evolutionary algorithms to develop parameters for computational models against multiple sets of experimental data; in particular, we propose the 'speciated non-dominated sorting genetic algorithm' for evolving models in several theories. We discuss the problem of developing a model of categorisation using twenty-nine sets of data and models drawn from four different theories. We find that the evolutionary algorithms generate high quality models, adapted to provide a good fit to all available data.

Keywords: cognitive modelling, evolutionary algorithms, model comparison, optimisation

1. Introduction

As in many sciences, computational models are used in the psychological and cognitive science communities in the development of theories. A computational model crystallises some elements of a theory into an executable form. It is generally accepted that computational models offer a number of valuable features. These include: (a) clear and rigorous specification of the mechanisms and parameters underpinning a theory; (b) derivation of testable predictions; (c) possibility of simulating complex behaviour, both qualitatively and quantitatively, irrespective of the number of variables involved; (d) possibility of systematically manipulating some variables to explore a model, which enables a better understanding of the (often non-linear) dynamics of a system; (e) help in making sense of rich and dense datasets; and (f) provision of explanations that are 'sufficient', in

the sense that they can produce the behaviour under study (Gobet and Waters, 2003; Newell and Simon, 1972; Pew and Mavor, 1998; Rumelhart and McClelland, 1986; Simon and Gobet, 2000). Finally, modelling can have important implications for applied research. For example, in clinical psychology, precisely-specified theories can inform the design of treatments, and provide guidance for policy making, including prevention (Gobet and Schiller, 2011, discuss the case for problem gambling).

However, some important issues that may limit the benefits of computational modelling have been unresolved so far. Amongst the most pressing, one can mention the difficulty of comparing different models, in particular when they differ in their complexity and the number of parameters (Richman and Simon, 1989; Ritter et al., 2003), the danger of developing models that overfit the data (Roberts and Pashler, 2000), the limited scope of some models (Newell, 1990), the exact role of models in developing general theories of, e.g., cognition (Newell, 1990), and the need to clearly separate, in a given model, what constitutes the theoretical claims and what should count as just implementational convenience (Cooper et al., 1996; Cooper and Shallice, 1995). In particular, there are few, if any, standardised guidelines telling the researcher how to develop and improve a model, and how to compare against other models and against empirical data (Gluck et al., 2010; Gobet and Ritter, 2000; Grant, 1962; Lane and Gobet, 2003, 2012; Newell, 1990; Pew and Mavor, 1998; Ritter et al., 2003; Roberts and Pashler, 2000; Stewart and West, 2010), except that parameters are tuned to fit a dataset.

Some earlier suggestions include those of (Cooper, 2002, Chapter 9), who proposes a Monte Carlo simulation approach, running the same model repeatedly over the same input to obtain descriptive statistics of the model's behaviour. This repeated running of the model treats a model as a single individual, and repeated runs as generating the behaviour of a group of individuals; the repeated results can be compared with the measured performance of human participants. Cooper (2002, p.375) notes that this approach can be used to "give the modeller a better understanding of dependencies of the model's behaviour on its parameters, or it may be used to determine values of the parameters that yield the optimal fit between the model's behaviour and participant behaviour." However, apart from suggesting the Monte Carlo approach is applied to a subset of the parameter space, Cooper (2002) does not propose any detailed techniques for ensuring good models are developed and compared. Gunzelmann (2008) compares an ACT-R model with human performance using a variety of different measures of fitness, including response time and variations in the provided stimuli; although again, this process is not automated.

The ultimate goal of developing cognitive models is to produce better theories of human behaviour. However, most theories in psychology and cognitive science have a number of parameters, and this number can sometimes be considerable. In spite of some efforts in this direction (Kase, Ritter, and Schoelles, 2008; Moore Jr, 2011; Ritter, 1991; Tor and Ritter, 2004), few modellers (outside mathematical psychology) routinely use formal, automated techniques to optimise the parameters and/or the structural components of their models. The fact that most cognitive models are not optimised poses important problems for building and evaluating theories. For example, what is the meaning of comparing two theories if their parameters have not been optimised (Gobet and Ritter, 2000; Ritter, 1991)? A complicating factor in psychology is that many experiments are frequently performed in the same paradigm, leading to multiple datasets, any one of which may be modelled.

One of the challenges we take up in this article is the problem of adapting a model to multiple datasets simultaneously: we propose a solution by treating the problem as one of multi-objective

optimisation, and then use a form of genetic algorithm to locate models which are not outperformed by any other model; these models are known as the *non-dominated* set. Although it is not possible to guarantee the presence of a truly optimal model, the optimisation process develops a set of models which are superior to a significant number of competing models. Hence, our contribution in this article is a technique for generating a pool of candidate models where the models are drawn from multiple theories and evaluated against multiple datasets. This work complements that of authors proposing alternative evaluation functions (e.g., Stewart and West, 2010) and techniques for gathering empirical data (e.g., Myung and Pitt, 2010); all three aspects of gathering data sets, designing evaluation functions, and finding good candidate models are important in the methodology of cognitive modelling.

As an example we consider the problem of categorisation, which is a long-standing and important area of psychology where many of the modelling issues identified above have been found (Fisher, Pazzani, and Langley, 1991; Medin, 1989; Murphy, 2002). Categorisation is the process of placing instances of stimuli into classes. A classic categorisation experiment, described by Medin and Smith (1981), requires experimental participants to place stylised faces into one of two categories. As summarised by Smith and Minda (2000), psychologists and modellers have subjected this experiment to intense study. However, as Smith and Minda make clear, the intensity of the study has led to a large number of competing models and theories of how categorisation occurs in humans. Turning this volume of study into a coherent set of ideas is a difficult challenge. Smith and Minda's attempts have, in their turn, been critically assessed (Gluck et al., 2001; Nosofsky, 2000). The debate highlights the important methodological issue of how to best justify and support a cognitive model when it must explain data from a wide range of experiments, and be compared against multiple competing models.

In this article we propose methods that enable theorists (a) to automatically optimise theories not only against a single measure, but against an arbitrary number of measures, and (b) to automatically compare the behaviour of theories against alternative theories, using either one or several measures. For this to apply, we require that the implementation of a cognitive theory should support the following two properties. First, models derived from the theory must support *automated* behavioural tests (Lane and Gobet, 2003, 2012), where each behavioural test is a comparison of the model to experimental data. Second, we require the class of models within the theory to be parameterised, so that computational optimisation techniques can locate those models which best meet multiple fits to experimental data; we introduce a novel form of non-dominated sorting genetic algorithm (Goldberg, 1989; Schaffer, 1984) to find these models.

We begin by outlining the problem of categorisation and how measures of fitness between a model and a set of experimental data may be computed. An experimental *constraint* on a model is defined in terms of the experimental setup, the collected set of data, and the measure of fitness which the model is assessed against. We then describe our proposed methodology for representing classes of models drawn from competing cognitive theories, before introducing an optimisation technique based on evolutionary theory. We use a series of experiments to both demonstrate the efficacy of our approach, and to illustrate how automatic optimisation techniques can help propose and resolve theoretical questions about the suitability of different theories.

2. Psychological Data in Categorisation

Categorisation is the task of assigning an experimental stimulus into one of a predetermined set of categories. Typically, the experimental participant is provided with a set of stimuli along with their category label to learn from: these are the *training* examples. The participant is then asked to select a label for each stimulus from a test or *transfer set*.

In a machine learning application, it would be relatively straightforward to select an algorithm by attempting to reduce the number of mistakes made on the transfer set; if the stimuli are all sampled independently from the same source, then this is the *generalisation error*. However, in a psychological experiment, what is of interest is the behaviour of human participants when carrying out the task, and in particular how they categorise novel data. For example, having learnt the training items, what proportion of participants place each training or transfer item into category A? How long does it take a participant to obtain a response for each item? How many errors are made whilst learning the training items? These questions help uncover the appropriate data structures to use in modelling human memory, and a theory is judged on its ability to produce models which match the performance of human participants in a wide range of experiments and measures, such as response time and number of errors. Every experimental result forms a constraint on the range of candidate models provided by a theory.

We define an experimental *constraint*, encapsulating the following information:

1. The experimental setting, consisting of its stimuli, and the separation of the stimuli into training and transfer sets.
2. The data collected from the participants.
3. The measure of fit used to compare the model's performance with the participants'.

Each constraint can then be applied to a model by running the model on the experiment, collecting similar data to that obtained from the participants, and then computing the measure of fit between the performances of the model and the participants.

We focus on a specific categorisation experiment, known as the five-four structure. This experiment has yielded a rich set of experimental data and models (Medin and Smith, 1981; Smith and Minda, 2000). In this section, we introduce the five-four structure, and the canonical experimental setup. Also, we consider how the collected data are used in theory development, in particular discussing how the measure of fit between the model and the data is computed, and explain further how this measure of fit is combined with the experimental setup to form an experimental constraint.

2.1 The Five-Four Structure

Smith and Minda (2000) describe a collection of thirty previous experimental results using the five-four structure, and analyse a number of mathematical models of behaviour. Although instantiated in different ways, the basic experiment uses the structure illustrated in Table 1. There are four binary attributes: different interpretations for each stimulus are created by varying the meaning of the attributes. The four binary values provide sixteen different stimuli, labelled E1 to E16. Examples of category A are typically those closer to having all four attribute values set, whereas examples of category B are typically those closer to having all four attribute values unset. Example E2 is interesting because it is the only one with two set values which is in category A.

Example	Attribute (A)				Example	Attribute (A)			
	A0	A1	A2	A3		A0	A1	A2	A3
A examples					Transfer items				
E1	1	1	1	0	E10	1	0	0	1
E2	1	0	1	0	E11	1	0	0	0
E3	1	0	1	1	E12	1	1	1	1
E4	1	1	0	1	E13	0	0	1	0
E5	0	1	1	1	E14	0	1	0	1
B examples					E15	0	0	1	1
E6	1	1	0	0	E16	0	1	0	0
E7	0	1	1	0					
E8	0	0	0	1					
E9	0	0	0	0					

Table 1: The five-four structure used in categorisation experiments.

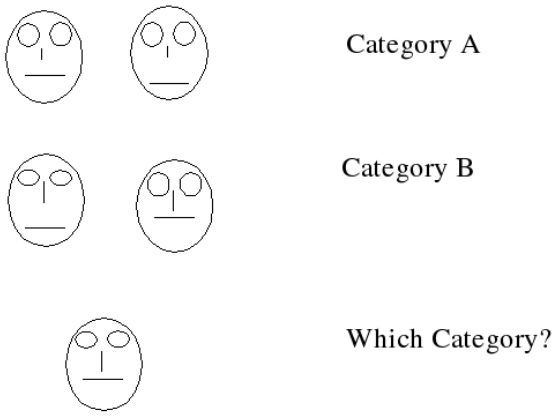

Figure 1: Example stimuli presented to humans taking a categorisation experiment.

These data can be used to create different categories of objects, depending on what interpretation is given to the attributes. For example, by making A0 eye height, A1 eye separation, A2 nose length, and A3 mouth height (as shown in Figure 1), we obtain the face experiment performed by Medin and Smith (1981). The thirty different experiments discussed by Smith and Minda (2000) used different interpretations of the attributes, and varying instructions for the participants.

2.2 Computing the Fitness

Given a model and an experiment, we can obtain the predicted behaviour of the model on the experiment. In experiments with the five-four structure, responses are collected for each of the 9 training and 7 transfer items. Across experiments, the probability of the response being A provides 16 predicted and 16 target figures for each (experiment, model) pair. Following standard practice in

	P($R_A|E_i$)							
Example	1st	2nd	3rd	4th	5th	$\cdots$	29th	Avg
E1	0.78	0.97	0.89	0.77	0.81	$\cdots$	0.90	0.83
E2	0.88	0.97	0.94	0.97	0.75	$\cdots$	0.75	0.82
E3	0.81	0.92	0.94	0.98	0.95	$\cdots$	0.97	0.89
E4	0.88	0.81	0.72	0.70	0.77	$\cdots$	0.95	0.79
E5	0.81	0.72	0.78	0.60	0.80	$\cdots$	0.90	0.74
E6	0.16	0.33	0.27	0.55	0.42	$\cdots$	0.23	0.30
E7	0.16	0.28	0.30	0.28	0.30	$\cdots$	0.20	0.28
E8	0.12	0.03	0.09	0.17	0.25	$\cdots$	0.19	0.15
E9	0.03	0.05	0.05	0.13	0.11	$\cdots$	0.04	0.11
E10	0.59	0.72	0.45	0.73	0.62	$\cdots$	0.59	0.62
E11	0.31	0.56	0.20	0.65	0.31	$\cdots$	0.23	0.40
E12	0.94	0.98	0.88	0.87	0.89	$\cdots$	0.99	0.88
E13	0.34	0.23	0.58	0.22	0.34	$\cdots$	0.33	0.34
E14	0.50	0.27	0.08	0.28	0.31	$\cdots$	0.43	0.40
E15	0.62	0.39	0.75	0.52	0.62	$\cdots$	0.60	0.55
E16	0.16	0.09	0.12	0.12	0.20	$\cdots$	0.14	0.17

Table 2: Target behaviours for the probability of responding A in different experiments using the five-four structure. This is a subset of Appendix B of Smith and Minda (2000) (and leaves out one duplicated dataset).

psychology, these 16 figures are converted into a single measure of fit between the target behaviour and the model. This measure may be computed in different ways, of which we consider two: the sum-squared error (SSE) and the average-absolute difference (AAD).

$$\text{SSE} = \sum_{i=0}^{15}(p_i - t_i)^2$$

$$\text{AAD} = \left[\sum_{i=0}^{15}|p_i - t_i|\right]/16$$

where p_i is the i^{th} predicted response, and t_i the i^{th} target behaviour.

2.3 Defining Constraints

The experimental constraints are defined by selecting the experiment to perform, a target response to model, and a measure of fitness. Results are reported by describing the constraint. For example, in Table 4, the constraint 'SSE 1st' refers to using the first piece of experimental data, and fitting using sum-squared error. (The 1st, 2nd, 3rd etc refer to the data presented in Smith and Minda (2000).

AVG is the average across all the 29 datasets considered there.[1] Target figures for a selection of datasets and AVG are shown in Table 2.)

We evaluate a model by comparing its performance against the experimental constraints. This evaluation requires the following steps:

1. Perform the defined experiment on the model: in this case, train the model on the given training data, and collect its probability of assigning every instance to category A.

2. Use the measure of fitness to compare the model's performance against the target response, given by the collected data.

For example, constraint 'SSE 1st' would require the model's performance to be evaluated using the sum-squared error measure of fitness, comparing the model's probabilities with those from the 1st experiment, listed in Table 2.

3. Models of Categorisation

The motivation behind different models of cognitive behaviour varies across cognitive science. We include examples here of three important classes of models: mathematical models, which capture the essence of a particular behaviour through mathematical equations; connectionist models, which use mechanisms loosely based on neural systems in the brain; and discrimination-network models, which explain high-level patterns of human learning and memory. We describe how each model works, the parameters which define the model, and how it is used to predict values for the target behaviour: the probability of responding with category A.

3.1 Mathematical Models

Smith and Minda (2000) consider eight mathematical models, most taken from earlier psychological literature on categorisation. We describe two of these here to illustrate the kinds of model being considered. The aim of all eight models is to compute, from the training examples, the probability of responding with category A to a given example. The *context model* (Medin and Schaffer, 1978) is an exemplar model, which defines the probability of responding with category A given exemplar E_i as:

$$\mathrm{P}(A|E_i) = \frac{\sum\limits_{j \in C_A} \eta_{ij}}{\sum\limits_{j \in C_A} \eta_{ij} + \sum\limits_{j \in C_B} \eta_{ij}}$$

where

$$d_{ij} = c \left[\sum_{k=0}^{3} w_k |x_{ik} - x_{jk}| \right]$$

1. Note that two of the sets of results in Smith and Minda (2000) appear to have been duplicated.

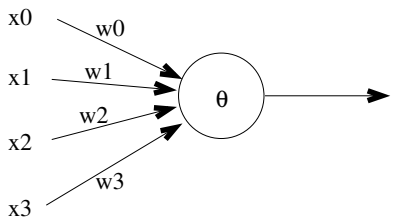

Figure 2: Structure of the perceptron.

and x_{ik} and x_{jk} are the values of the item and exemplar on attribute k, w_k is the attentional weight to attribute k, c is the sensitivity parameter, $C_{A/B}$ are the training examples in category A/B, and $\eta_{ij} = e^{-d_{ij}}$.

The model is completed with a 'guessing' parameter, used to model the fact that, infrequently, people simply guess a category, without any memory retrieval. The probability that the model will guess when providing its response is γ, with a 0.5 probability of choosing category A. The model provides a probability of responding with category A, which may be compared directly with the gathered experimental data.

The *prototype* model uses a similar set of formulae to the context model, but calculates similarity to a canonical example, not all the examples. The canonical example is found with reference to the initial dataset: category A is represented by all four features taking the value 1, and category B by all four features taking the value 0.

The two mathematical theories are parameterised by six parameters: $\{ w_0, w_1, w_2, w_3, c, \gamma \}$, where $\Sigma_{i=0}^{3} w_i = 1$.

3.2 Connectionist Models

The typical connectionist network (Rumelhart and McClelland, 1986) comprises a set of nodes interconnected by weighted links. The links pass activation between the nodes. Each node computes a simple function. Learning is a process of modifying the weights on the links between nodes. We use a simple, single-unit network (a perceptron; illustrated in Figure 2) to model the categorisation experiment. The decision function for choosing category A is:

$$\sum_{i=0}^{3} w_i \times x_i \; > \; \theta$$

where w_i is the i^{th} weight, x_i is the i^{th} input, and θ a threshold.

The model is a *learning* model, and the weights, w_i, may be trained using the rule:

$$\Delta w_i = \eta \times x_i \times (target - output)$$

where $output$ is the current output value, and $target$ is the desired output. η is a parameter governing the learning rate.

Connectionist models output a definite class, and so, to get a population of models from which to compute the probability of responding with category A, we train 20 models for 10 learning cycles to run the experiments. The model is defined by three independent parameters: θ, the threshold,

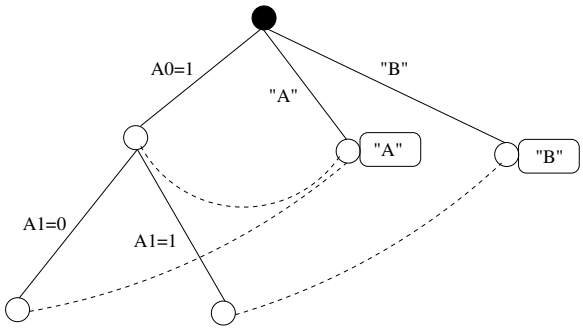

Figure 3: An example discrimination network, classifying the five-four structure

η, the learning rate, and ρ, the probability of learning an example. The connectionist theory is parameterised by three parameters: $\{ \theta, \eta, \rho \}$.

3.3 Discrimination-Network Models

Discrimination-network models, such as CHREST (Gobet et al., 2001), capture the *processes* underlying behaviour. The internal representation for learnt knowledge is a discrimination network, as illustrated in Figure 3. Information is stored as chunks at individual nodes. Tests on the links between nodes are used when sorting a pattern from the root node (the black circle at the top). The dashed links indicate *naming links*, which are used by CHREST to associate categories to perceived information. The discrimination network is built up incrementally as the model is given each input in turn.

Three kinds of learning occur. The first, *discrimination*, adds a new test link and node to the network. Discrimination occurs when a pattern is sorted to a node, but does not match the chunk stored at that node. The new test link is created based on the next available test feature in the pattern. The second learning mechanism, *familiarisation*, adds information to the image of a node. Familiarisation occurs when a pattern is sorted to a node and matches the stored chunk. New information from the sorted pattern is added to the stored chunk. The third and final learning mechanism is used to add the naming links between nodes of different modalities. Specifically, during training, when CHREST has retrieved a node based on a pattern from the five-four structure, and retrieved a node relating to the category of that pattern, CHREST can form a naming link between the two retrieved nodes. Patterns are classified by CHREST during testing by sorting them through the network, and then retrieving the node using the retrieved node's naming link.

As with the connectionist models, the discrimination-network models give a definite categorisation for each stimulus, so, to obtain probabilities of response, we need to simulate a number of models. (In the experiments discussed below, 20 models were used.) Variations in the level of performance are achieved by making learning stochastic. When a training example is given to each model, it will, with probability ρ, attempt to learn the pattern. The probability of response is then obtained from the number of trained models which produce category A in response to each example. The CHREST theory is parameterised by just one parameter: $\{ \rho \}$.

4. Evolving Non-Dominated Cognitive Models

In the two previous sections we have described the set of experimental constraints for these models, and the range of theories from which models may be drawn; the remaining problem is how to find one or more models within the theories which satisfy the constraints. The challenge of identifying an optimal cognitive model is usefully considered as an *optimisation problem*. In addition, because we require each model to perform as well as possible in multiple experiments, the problem is a *multi-objective* optimisation problem.

We begin this section by formalising the search for a cognitive model as a multi-objective optimisation problem. One of the challenges is that it is not always possible to rank two different solutions: one may be better on one objective, but the other better on a second. Therefore, we look for a set of solutions, where no member of the set can be outperformed everywhere by another solution; this set is known as the *non-dominated set*, and, if perfectly calculated, will contain the optimal models; a variety of terms is used in the literature for this set, including 'efficient' and 'pareto-optimal', but we prefer the descriptive term 'non-dominated'. In common with other applications, the nature of these models precludes the use of an analytic technique to calculate the non-dominated set. Instead, search techniques, which gradually approximate the non-dominated set, are used to find an approximation to the optimal models.

There exists a range of techniques for finding non-dominated sets in the literature. Genetic algorithms (Holland, 1975) are one of the most suitable for multi-objective optimisation, due to the parallel nature of their search, which allows even complex problems to be tackled; several different algorithms of this kind are used (Coello, 2000, 2003; Maneeratana, Boonlong, and Chaiyaratana, 2004; Srinivas and Deb, 1994). We describe a version of the non-dominated sorting genetic algorithm (NSGA), and describe a new version, specifically designed to develop optimal cognitive models for multiple theories.

4.1 Non-dominated sets of parameter values

Consider the illustration in Figure 4 of two competing experimental constraints. The x-axis represents different values for the single parameter, and the y-axis represents the quality of fit of the model to an experiment; fit is assumed to be computed so that values are positive, and the aim is to reduce the fit to a value of 0. The curve f_1 on the left of the figure represents the range of fit of various parameter settings to one experimental setting. Notice there is a clear optimum, labelled x_2, where $f_1(x)$ is a minimum. The curve f_2 to the right of the figure represents the fit to a second experimental setting. Notice there is again a clear optimum for $f_2(x)$, labelled x_4, but it is different to x_2.

The notion of a non-dominated set of parameter values is taken from multi-objective optimisation theory. A non-dominated assignment of values is defined as one which is better than all other values in at least one constraint (also see Maneeratana, Boonlong, and Chaiyaratana (2004)). Thus, x_2 and x_4 are in the non-dominated set, as they are better than all other assignments to x in f_1 or f_2 respectively. x_1 is not in the non-dominated set, because x_3 is just as good for f_1 but better for f_2, that is $f_1(x_1) = f_1(x_3)$ but $f_2(x_1) > f_2(x_3)$. Similarly x_5 is not in the non-dominated set. All assignments for x between x_2 and x_4 inclusive are non-dominated assignments, as can be checked visually.

Algorithms for multi-objective optimisation tasks may be designed to return examples of the complete non-dominated set; a genetic algorithm to achieve this is described in Section 4.3.2.

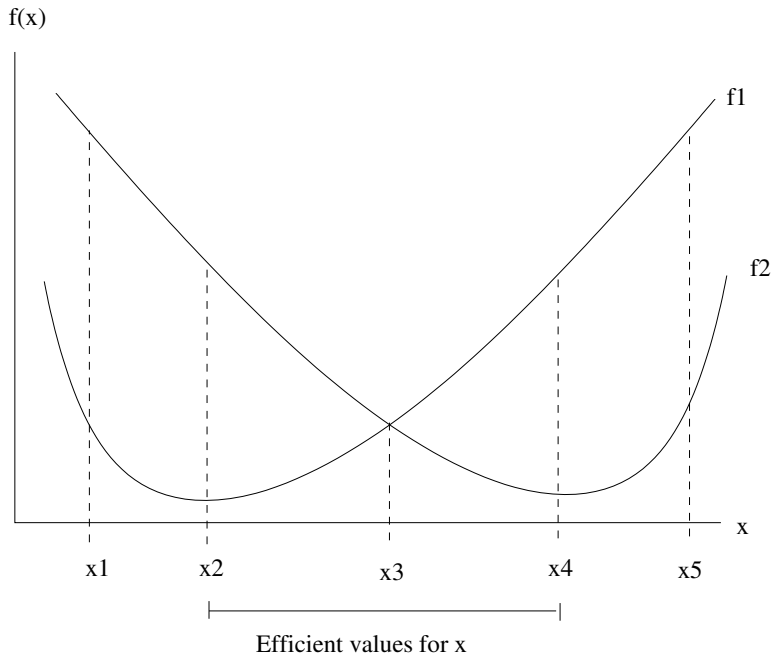

Figure 4: Illustration of the non-dominated set of parameter values for a model fit to two different constraints.

Although our main aim is to make a case for the development of non-dominated sets of parameter values as a necessary precursor to comparing classes of computational models, there are some objections to using the complete set. The main objection can be seen by considering the point x_2 in Figure 4. Although this point is optimal with respect to constraint f_1, it has a very poor performance with respect to constraint f_2; this is an example of over-fitting. Another objection arises from considering the relative importance of the constraints; in the example above, treating f_1 and f_2 equally could lead to a convincing case to be made for the point x_3 as 'the' best value. However, in an experimental setting, some experiments may be regarded as more robust than others, or perhaps some experiments fit better within a given modelling paradigm. Returning a representative sample of the complete non-dominated set allows us to observe the effect of all considerations of relative priority of the constraints, from completely equal, in the case of x_3, to ignoring f_2, in the case of x_2, or ignoring f_1 in the case of x_4. The approach advocated here is that, initially, the modeller should become aware of the complete set of candidate models, and use this set to make qualitative decisions about parameter values, to observe the fit of a particular model to given experimental settings, and also to compare competing theories.

4.2 Multi-objective optimisation

In Section 3, we discussed some cognitive theories, each of which defines models using a set of parameters. We will treat the four classes of models as defining a space, $\mathcal{M}$, of possible models. Our aim is to find one or more $m \in \mathcal{M}$ satisfying the multiple experimental constraints, where m is a specific set of values for the parameters in a given theory.

Each experimental constraint is defined as a function, mapping a given model to a real number:

$$f_i : \mathcal{M} \rightarrow \mathbb{R}$$

Without loss of generality, we minimise $f_i(m) \geq 0$. Note that i refers to the specific experiment.

Model m_1 is said to *dominate* model m_2 if:

$$\forall i \bullet f_i(m_1) \leq f_i(m_2) \ \wedge \ \exists j \bullet f_j(m_1) < f_j(m_2) \tag{1}$$

In other words, m_1 does at least as well as m_2 everywhere, but there is at least one task in which m_1 does better. For our categorisation task, this would mean that model m_1 provides at least as good a fit as m_2 in every experiment, and in at least one experiment m_1 provides a better fit than m_2.

The non-dominated solutions will be those models, selected from multiple theories, which are not outperformed on all of the multiple tasks. We now describe an algorithm for automatically locating non-dominated models.

4.3 Genetic algorithms

A *genetic algorithm* (GA) (Holland, 1975) is a form of search technique inspired by natural selection. The distinctive property of a GA is that it continually refines a *population* of individuals. The population is sorted, using a *fitness* function, to rank the individuals. The best individuals are then combined, using cross-over, to create new individuals. Mutation may also be applied, to randomly alter some of the new individuals. The search is repeated until a good individual is achieved, or the experimenter decides enough cycles have been executed.

We use three forms of GA in the experiments below:

Simple GA to evolve a model from a single theory on one experimental constraint.

NSGA to evolve a set of non-dominated models from a single theory on multiple experimental constraints.

Speciated NSGA to evolve a set of non-dominated models from multiple theories, based on multiple experimental constraints.

All three use a similar cycle of evolution, creating a population and evolving the population using cross-over and mutation. The set of numeric parameters for each theory forms the *genotype*, and these parameters are used to create a model appropriate to that theory (in an evolutionary sense, the model is the *phenotype*). Crossover between two models creates a new model using some of the parameters from the first model and some from the second. Mutation is simply the random change of one of the parameters; we have set the rate of mutation at 10%.

We use an *elitist* strategy, selecting the best individuals of the current population: for the simple GA these are the best 10% of the population, and for the non-dominated sorting GA's, these are the non-dominated members of the population. The elite members are passed unchanged to the new population, and the population size is made up through cross-over and mutation. The elitist strategy means that the best individuals of the population are always preserved between cycles.

The different forms of GA use different fitness functions, and also different compositions of their populations. However, in each case, the populations are made up from individuals, where each

individual is an assignment of values to the parameters in a given theory; each individual is a specific *model*. The fitness functions use the experimental constraints to determine how good each model is with reference to the other models in the current population(s). We now describe the different forms of GA in turn.

4.3.1 SIMPLE GA

The simple GA searches through models drawn from a single theory, ranking its population by performance on a single experimental constraint. The initial population for the simple GA consists of a random sampling of parameter values for models from a given theory. As each constraint measures the performance of a model yielding a single numeric value, it is easy to rank the population to find the best models.

4.3.2 NON-DOMINATED SORTING GA

The non-dominated sorting genetic algorithm (NSGA) is used when multiple objectives must be satisfied, and is a technique for ranking models although there is no linear measure of fitness available. NSGA modifies the standard GA by altering the manner in which the population is ranked. As there are multiple objectives, there is no simple linear measure of 'better'. Goldberg (1989) was the first to propose using domination as a means of ranking the models in the population, and later work has developed the technique (Coello, 2003; Maneeratana, Boonlong, and Chaiyaratana, 2004; Srinivas and Deb, 1994).

NSGA ranks a population by separating it into sets of non-dominated individuals. Using the definition of *dominates* (1), those individuals which are *not* dominated are first separated out of the initial population. These form set 1. Set 2 is then formed by separating out the non-dominated individuals of the remaining population. This process continues for set 3, with set 4 consisting of the remainder. The members of set 1 are now passed to the new population unchanged, and the remainder of the new population is made up using cross-over preferring those members of set 1, but also using members of the other sets, with decreasing probability from set 2 to set 4. Table 3 illustrates the NSGA algorithm used here, and step 3 describes how the probability of cross-over changes with the set in which a model is found.

4.3.3 SPECIATED NSGA

One of the problems of using a single population with models drawn from different theories is that of premature convergence, where NSGA can quickly produce a population of models drawn from a single theory (Lane and Gobet, 2005). Here, we introduce a variation of NSGA, which we call "Speciated NSGA", in which separate populations are managed for each theory. Our variation ensures that we always have a selection of candidate models from each theory, and these candidates can evolve, in competition against their own and other theories. When evolution is completed, we may then determine which models from which theories are the ones which we wish to analyse.

The GA is started off with N populations, one for each of the theories being considered (in our example, $N = 4$). In almost all stages, the algorithm proceeds as with NSGA, as shown in Table 3. The only step where the N populations interact is when computing whether a model is 'dominated'. In this step, all the populations are combined, so that a model in one theory may be dominated by a model from another theory. This combination promotes *competition* between the theories. A new

1. Create an initial population.
2. For each member of the population, compute its performance on the given fitness function, and then form four sets according to the following rules:

   set 1 the non-dominated individuals of the entire population;

   set 2 the non-dominated individuals of the entire population without set 1;

   set 3 the non-dominated individuals of the entire population without set 1 and set 2; and

   set 4 the entire population without sets 1, 2, and 3.

3. Create a new population consisting of the members of set 1, and the results of performing cross-over on individuals selected from the four sets. The probability of selecting a member of set 3 is double that of selecting from set 4; from set 2 double that of selecting from set 3, and from set 1 double that of selecting from set 2.
4. Mutation is performed on the new individuals (those not in set 1).
5. The process begins again at step 2, until the maximum number of cycles, or stopping criterion, has been reached.

Table 3: Non-dominated sorting genetic algorithm. See text for details of the population makeup, and the construction of non-dominated sets.

population is created for each theory based on the distribution of individuals from its own theory across the N sets.

The chief benefit of our algorithm may be seen by considering one of the worst-case scenarios: what would happen if every model in one of the theories was dominated by models in another? In regular NSGA, there would be no new models from this dominated theory. In our variation, a new population will be created, still preferring candidates from set 2 over set 3. Once evolution is complete, extracting the non-dominated models from all the theories may mean some theories have no models. However, we will still have evolved a set of models which do well for that theory, and can analyse their performance.

A related benefit is that some theories require comparatively longer to evolve good values for their parameters. An evolutionary algorithm such as NSGA will tend to remove poorly performing models, and so fail to evolve effective models for such theories. Our approach instead ensures models from all theories are developed, and so the slower-evolving theories can 'catch up', and produce non-dominated models at a later period.

5. Experiments

We demonstrate and evaluate our approach in three experiments developing computational models of categorisation. First, we consider the performance of models on all constraints when optimised against a single constraint. Second, we explore, within a given theory, the quality of models evolved against all available constraints, using multi-objective optimisation. Finally, we consider the impact of competition against other theories, and analyse the quality of models produced. Two parameters determined in advance are the number of models within each population, and the number of cycles of evolution used. The larger these two numbers are, the more models get explored, and the more likely it is that 'the best' model(s) will be located. It is possible to automate these values to some extent Ritter et al. (2011), however, we find that using 500 models in each population, and 300 cycles of evolution provides consistent results, so these numbers are used throughout (except in the very first experiment).

When creating a new population, we have chosen to build and rank the models using their performance on the complete set of constraints. An alternative technique is to divide the constraints into a 'training' and a 'cross-validation' set, using the training set to build the models, and the cross-validation set to rank them. We have chosen not to do this here for three reasons. First, our aim is to demonstrate how evolutionary algorithms can generate a set of non-dominated models whose parameters may then be analysed. Second, this seems to fit more closely with how mathematical models are built and ranked on single datasets, in particular using the novel instances in the transfer set for evaluation (see Nosofsky, 2000; Smith and Minda, 2000). Third, where cross-validation is used, the selection of cross-validation sets must be repeated, and some averaging procedure adopted. An area where this methodology is important is machine learning, such as the training of support-vector machines. A standard recommendation is to use cross-validation to locate parameters which provide good average performance across all the cross-validation sets, and then retrain the algorithm with *all* the data using these best parameters (Hsu, Chang, and Lin, 2003, p.8) prior to subsequent testing, where the performance of the now fixed models is assessed on the test set. We return to this point in the discussion.

	Average Absolute Difference (AAD)					Sum-Squared Error (SSE)				
	AVG	1st	2nd	3rd	4th	AVG	1st	2nd	3rd	4th
AAD										
AVG	**0.03**	0.06	0.11	0.11	0.11	0.02	0.10	0.28	0.31	0.29
1st	0.08	**0.04**	0.12	0.11	0.16	0.13	0.04	0.39	0.35	0.57
2nd	0.09	0.12	**0.03**	0.12	0.07	0.16	0.31	0.03	0.46	0.13
3rd	0.10	0.10	0.13	**0.04**	0.14	0.21	0.23	0.47	0.09	0.60
4th	0.08	0.13	0.08	0.15	**0.03**	0.19	0.45	0.16	0.65	0.02
SSE										
AVG	0.03	0.07	0.09	0.10	0.09	**0.02**	0.12	0.19	0.29	0.19
1st	0.08	0.03	0.12	0.10	0.15	0.12	**0.03**	0.34	0.32	0.51
2nd	0.09	0.13	0.03	0.14	0.08	0.20	0.35	**0.02**	0.19	0.35
3rd	0.10	0.11	0.11	0.04	0.14	0.21	0.29	0.39	**0.05**	0.50
4th	0.09	0.13	0.07	0.14	0.04	0.21	0.46	0.12	0.68	**0.03**

Table 4: Sample results from experiment 1 for the context class of models. (Numbers indicate difference between model's performance and human data. Each row gives the constraint the model was optimised against, and the column the constraint it was tested against. Highlighted numbers indicate performance on a constraint which model was optimised against.)

	Average Absolute Difference (AAD)					Sum-Squared Error (SSE)				
	AVG	1st	2nd	3rd	4th	AVG	1st	2nd	3rd	4th
AAD										
AVG	**0.05**	0.10	0.11	0.14	0.08	0.06	0.25	0.26	0.40	0.20
1st	0.07	**0.07**	0.10	0.10	0.11	0.11	0.14	0.28	0.33	0.39
2nd	0.11	0.14	**0.06**	0.14	0.07	0.23	0.49	0.10	0.58	0.10
3rd	0.13	0.13	0.15	**0.05**	0.18	0.36	0.47	0.57	0.07	0.71
4th	0.06	0.12	0.09	0.12	**0.05**	0.12	0.36	0.18	0.38	0.09
SSE										
AVG	0.05	0.08	0.10	0.10	0.10	**0.06**	0.19	0.27	0.28	0.26
1st	0.07	0.07	0.12	0.11	0.13	0.13	**0.14**	0.33	0.31	0.45
2nd	0.11	0.14	0.06	0.13	0.08	0.24	0.45	**0.09**	0.42	0.16
3rd	0.11	0.12	0.13	0.05	0.15	0.29	0.43	0.44	**0.07**	0.58
4th	0.10	0.15	0.08	0.16	0.05	0.25	0.55	0.18	0.70	**0.06**

Table 5: Sample results from experiment 1 for the prototype class of models.

	Average Absolute Difference (AAD)					Sum-Squared Error (SSE)				
	AVG	1st	2nd	3rd	4th	AVG	1st	2nd	3rd	4th
AAD										
AVG	**0.05**	0.08	0.10	0.11	0.09	0.06	0.18	0.23	0.29	0.22
1st	0.07	**0.07**	0.12	0.13	0.13	0.12	0.25	0.33	0.43	0.38
2nd	0.07	0.12	**0.09**	0.14	0.08	0.12	0.35	0.19	0.47	0.16
3rd	0.10	0.12	0.14	**0.06**	0.14	0.24	0.34	0.54	0.10	0.51
4th	0.09	0.13	0.11	0.12	**0.06**	0.22	0.47	0.26	0.42	0.12
SSE										
AVG	0.05	0.10	0.09	0.11	0.08	**0.06**	0.22	0.19	0.30	0.17
1st	0.06	0.08	0.11	0.12	0.12	0.09	**0.14**	0.31	0.41	0.38
2nd	0.09	0.13	0.08	0.13	0.08	0.16	0.41	**0.16**	0.41	0.15
3rd	0.09	0.11	0.13	0.07	0.13	0.20	0.33	0.44	**0.11**	0.43
4th	0.08	0.13	0.09	0.14	0.06	0.17	0.40	0.19	0.51	**0.10**

Table 6: Sample results from experiment 1 for the connectionist class of models.

	Average Absolute Difference (AAD)					Sum-Squared Error (SSE)				
	AVG	1st	2nd	3rd	4th	AVG	1st	2nd	3rd	4th
AAD										
AVG	**0.19**	0.22	0.19	0.26	0.13	0.89	1.32	0.81	1.57	0.49
1st	0.18	**0.23**	0.18	0.26	0.12	0.90	1.37	0.75	1.64	0.39
2nd	0.18	0.23	**0.18**	0.26	0.13	1.01	1.48	0.77	1.78	0.44
3rd	0.19	0.22	0.18	**0.26**	0.14	1.00	1.43	0.82	1.71	0.55
4th	0.18	0.23	0.18	0.26	**0.12**	0.80	1.23	0.78	1.52	0.43
SSE										
AVG	0.18	0.22	0.18	0.26	0.13	**0.79**	1.23	0.78	1.45	0.43
1st	0.18	0.23	0.18	0.26	0.12	0.74	**1.17**	0.74	1.46	0.42
2nd	0.18	0.23	0.18	0.26	0.12	0.82	1.26	**0.75**	1.56	0.42
3rd	0.18	0.23	0.19	0.27	0.13	0.73	1.17	0.79	**1.46**	0.42
4th	0.18	0.23	0.19	0.27	0.13	0.74	1.18	0.80	1.45	**0.42**

Table 7: Sample results from experiment 1 for the discrimination-network class of models.

5.1 Optimising models against a single constraint

The first question addressed is whether a model developed by optimising its parameters against data from one experiment transfers effectively to other experiments. For example, a model developed with one measure of fitness against the first set of experimental data in Table 2 should produce a good level of fit with that constraint. However, what will its level of fit be on a different constraint? And how will that compare with the performance of a model developed directly against the second constraint?

Method For each class of theory, and for each of the thirty constraints, we evolve an optimised model using a simple genetic algorithm. 300 cycles through the genetic algorithm are performed, with a population of 50 candidate models, returning the single best-performing model for each class of theory. The performance of that model is then calculated for every constraint, and the results tabulated.

Results Table 4 shows the results achieved by the context class of models on five of the constraints: the constraints based on the average of the 29 datasets, and the constraints based on the first four datasets shown in Table 2. The columns in the table label the constraint used in testing, and the rows in the table label the constraint used in building the model; hence, the leading diagonal (highlighted) shows the results on the constraint which the model was optimised on. Each constraint computes the difference between the model's performance and the target data. Hence, a model performing identically to the human data would have a level of fit of 0.

Tables 5, 6, and 7 show results in the same format, but for the prototype, connectionist and discrimination-network classes of models, respectively.

Discussion We find that models optimised on a single constraint perform less well on other constraints. Importantly, the best model on each constraint is the model optimised on that constraint in 96% of all the results. The exceptional cases are usually explained by similarity in the constraints, where the target data is the same, but the fitness function alone has changed: for example in Table 6, the result for the AAD 2nd constraint is smallest when the models are evolved against constraint SSE 2nd. Only the discrimination-network models (Table 7) fail to match this pattern, and this is because the performance due to training on any constraint is similar; a result of there being only a single parameter to specify models in this theory.

The result that models optimised on a single constraint perform less well on other constraints is not surprising. It is essentially a problem of *over-fitting*, frequently found in both machine learning and cognitive modelling (Leahy, 1994; Schaffer, 1993). Finding the best performing model for one constraint means that performance is not guaranteed to be as good for other constraints. This finding justifies the need for an optimisation approach which can take multiple constraints into account at the same time. We now seek models which perform well on *all* available constraints.

We use the results obtained in this first experiment as a baseline performance, against which the models created in later experiments are compared. When properly optimised, the model specifically evolved on each constraint should outperform a second, more general model. However, a more general model may instead provide smaller divergence across all of the constraints than is shown in the tables here. One aim of the remaining experiments is to test whether this is indeed the case.

Finally, we can compare our results with those obtained by Smith and Minda (2000), who optimised their models using a "fine grained hill-climbing algorithm", repeated five times with different seeding configurations. Smith and Minda give a result for the AAD AVG constraint of

	AAD					SSE				
Theory	AVG	1st	2nd	3rd	4th	AVG	1st	2nd	3rd	4th
C_1	0.10	0.15	0.14	0.20	0.10	0.21	0.47	0.46	0.74	0.24
C_7	0.12	0.12	0.21	0.13	0.19	0.32	0.41	0.87	0.51	0.75
C (avg)	0.14	0.18	0.20	0.22	0.14	0.42	0.73	0.77	0.93	0.46
P_{13}	0.09	0.13	0.16	0.16	0.13	0.20	0.43	0.49	0.55	0.49
P_5	0.14	0.19	0.20	0.18	0.15	0.38	0.69	0.74	0.62	0.48
P (avg)	0.13	0.18	0.19	0.21	0.14	0.41	0.71	0.74	0.88	0.49
Co_{32}	0.11	0.12	0.12	0.11	0.14	0.28	0.35	0.34	0.41	0.51
Co_{14}	0.08	0.11	0.14	0.12	0.14	0.16	0.31	0.44	0.36	0.42
Co (avg)	0.11	0.14	0.15	0.15	0.14	0.31	0.48	0.56	0.53	0.48
D_4	0.19	0.23	0.19	0.26	0.13	1.00	1.46	0.81	1.74	0.46
D_{10}	0.21	0.23	0.19	0.26	0.15	1.13	1.57	0.89	1.84	0.60
D (avg)	0.19	0.24	0.19	0.26	0.14	0.96	1.40	0.85	1.70	0.52

Table 8: Sample results from experiment 2. (C stands for Context models, P for Prototype models, Co for Connectionist models, and D for discrimination-network models.)

0.091 for the Prototype theory (their Additive prototype), where we obtain a result of 0.052, and 0.047 for the Context theory, where we obtain a result of 0.029. The 40% improvement in level of fit indicates that the genetic algorithm is performing at least as well as a hill-climbing algorithm, and the improvement suggests that the fitness landscape is not smooth enough for hill-climbing techniques to work optimally. Thus, just like Ritter et al. (2011), optimising a model with a genetic algorithm found parameters that led to a better fit than the parameters mentioned in the original publication.

5.2 Optimising models against multiple constraints

The second question addressed is whether multi-objective optimisation will evolve models of a high quality when using all available constraints. While in Experiment 1 each model for each constraint had a different set of free parameters, in Experiment 2 the same model will keep the same assignment of values to its free parameters for all constraints.

Method For each class of theory, we evolve an optimised model using the non-dominated sorting genetic algorithm. 300 cycles of the genetic algorithm are run, with 500 examples of each model type. The performance of models in the final non-dominated set is then calculated for every set of experimental data, along with the average performance of all the models, and the results tabulated.

Results Table 8 shows the results achieved by two randomly selected members of the non-dominated models evolved for each theory type. Also, the average performance of all the non-dominated models is shown. There were 43 context models, 50 prototype models, 50 connectionist models, and 50 discrimination-network models.

Discussion The performance of the models in this experiment do not match the very best results achieved from the previous experiment. This is only to be expected. In the previous experiment, a

	AAD					SSE				
Theory	AVG	1st	2nd	3rd	4th	AVG	1st	2nd	3rd	4th
C_{23}	0.12	0.17	0.18	0.19	0.11	0.33	0.70	0.56	0.69	0.29
C_{25}	0.06	0.11	0.10	0.11	0.08	0.11	0.33	0.22	0.25	0.16
C (avg)	0.13	0.18	0.19	0.21	0.13	0.37	0.67	0.69	0.87	0.41
P_{12}	0.10	0.15	0.17	0.13	0.14	0.26	0.51	0.57	0.32	0.47
P_4	0.15	0.21	0.17	0.22	0.11	0.56	0.96	0.64	1.04	0.35
P (avg)	0.13	0.18	0.19	0.21	0.14	0.41	0.71	0.74	0.87	0.49
Co_7	0.10	0.13	0.15	0.16	0.12	0.19	0.43	0.48	0.52	0.37
Co_{15}	0.08	0.10	0.14	0.14	0.12	0.15	0.33	0.42	0.43	0.37
Co (avg)	0.09	0.12	0.13	0.13	0.12	0.22	0.37	0.44	0.40	0.42
D_2	0.18	0.23	0.19	0.26	0.13	0.71	1.14	0.76	1.44	0.41
D_8	0.18	0.23	0.19	0.26	0.13	0.71	1.15	0.76	1.43	0.40
D (avg)	0.18	0.23	0.19	0.26	0.13	0.71	1.15	0.76	1.44	0.41

Table 9: Sample results from experiment 3. (C stands for Context models, P for Prototype models, Co for Connectionist models, and D for discrimination-network models.)

model's quality was evaluated purely on its performance on a single constraint; in this experiment, a model's quality is evaluated based on its *generality*. A general model will always be outperformed by a specialist on one constraint. However, the general model should, *in general*, be superior, when considering its performance across all the constraints. We consider this question below in Section 5.4.

5.3 Optimising models from multiple theories

The third question addressed is whether multi-objective optimisation with multiple theories will evolve models of a high quality when using models from all available theories, and all available constraints.

Method The speciated non-dominated sorting genetic algorithm is used to evolve models from every theory, optimising against all constraints. 300 cycles of the genetic algorithm were run, with 500 examples of each model type. The performance of models in the final non-dominated set is then calculated for every set of experimental data, and the results tabulated.

Results Table 9 shows results achieved by two of the non-dominated models taken from each class of models on five of the datasets, computed using the two measures of fitness; the average performance of all the non-dominated models from each class is also shown (the third row in each group). There were 41 context models, 50 prototype models, 50 connectionist models, and 50 discrimination-network models.

Discussion Similar considerations about the performance of the models against the first experiment apply here as in the previous experiment. Due to the requirement of each model to perform well on all constraints, it is unlikely to match the very best performance achieved by models optimised against a single constraint.

	AAD					SSE				
Expt.	AVG	1st	2nd	3rd	4th	AVG	1st	2nd	3rd	4th
Context										
1	0.031	0.035	0.034	0.035	0.028	0.098	0.125	0.115	0.144	0.073
2	0.020	0.021	0.022	0.030	0.023	0.208	0.223	0.239	0.425	0.255
3	0.015	0.015	0.017	0.021	0.018	0.114	0.111	0.139	0.224	0.157
Prototype										
1	0.025	0.027	0.030	0.032	0.021	0.080	0.089	0.096	0.129	0.049
2	0.021	0.023	0.024	0.028	0.023	0.198	0.241	0.292	0.399	0.245
3	0.020	0.022	0.024	0.025	0.021	0.185	0.226	0.269	0.301	0.210
Connectionist										
1	0.023	0.020	0.023	0.023	0.025	0.061	0.046	0.059	0.067	0.059
2	0.022	0.021	0.025	0.029	0.019	0.222	0.196	0.281	0.372	0.206
3	0.018	0.013	0.015	0.018	0.022	0.191	0.083	0.098	0.145	0.272
Discrimination-network										
1	0.007	0.007	0.010	0.007	0.017	0.007	0.005	0.011	0.005	0.017
2	0.010	0.006	0.006	0.004	0.010	0.041	0.015	0.017	0.008	0.037
3	0.004	0.002	0.003	0.002	0.003	0.0	0.0	0.0	0.0	0.0

Table 10: Standard deviation of non-dominated models from their average performance. (The numbers refer to the experiments.)

The interesting conclusion from this experiment follows from comparing the figures in Table 9 with those from the previous experiment, shown in Table 8. Considering the average performance of each population, in 28 out of the 40 results, the results in Table 9 are lower, which is true for 75% of the 60 average results. This suggests that evolving models from multiple theories produces *better* results than evolving models from single theories. The reason for this is that the additional competition between the theories 'encourages' the models to evolve to better solutions.

5.4 Further Analysis

We now provide some analysis of the models obtained in the different experiments, to determine whether information about the relative merits of alternative theories and experimental constraints can be obtained. In particular, we consider how the performance of the models in the three experiments compares, whether specific theories are better on particular constraints, and how much variation there is in the final evolved models.

5.4.1 COMPARING THE PERFORMANCES

First, we consider the quality of the models created when evolving models from multiple theories or against multiple constraints. A difficult issue here is how to measure 'quality'. One measure is whether the set of models produces a consistent set of performance measures; we measure consistency by the amount of variation in the models' performances, the standard deviation of performance from the average.

	Context	Prototype	Connectionist	Discrimination
AAD 6th	0.085	0.083	0.058	0.0 (best)
AAD 7th	0.056 (best)	0.064	0.122 (worst)	0.036
AAD 15th	0.073	0.053	0.043 (best)	0.002
AAD 19th	0.077	0.052 (best)	0.052	0.004
AAD 24th	0.226 (worst)	0.200 (worst)	0.100	0.067 (worst)
mean	0.46	0.43	0.25	0.21
s.d	0.40	0.37	0.18	0.20

Table 11: Level of fit of model performance against constraint

Method For each experiment, we consider the standard deviation in performance of a set of optimised models against the average result achieved by that set on each constraint. Table 10 gives the computed figures.

Discussion The figures in Table 10 compare the variation in performance of all the models returned by the genetic algorithm against the average results for each individual constraint. The table separates the four tested theories, and each row shows the results for a separate experiment, and an individual constraint. The table suggests two interesting results. The most important result is that the amount of deviation is lower in the third experiment than in the second. This argues for the value of comparing models from different theories, instead of treating each theory in isolation. The lesser, but still intriguing, result is that the amount of deviation is usually, though not consistently, reduced between the first and third experiments. The deviation typically reduces for the constraints using AAD as a fitness measure, whereas the constraints using SSE do not all reduce between the first and third experiments. The difference between the two fitness measures is interesting, but unexplained at this stage.

5.4.2 RELATIVE PERFORMANCE AGAINST CONSTRAINT

As the different constraints represent different kinds of experiments, it is natural to ask whether different theories do better on some experiments than others. We can explore this by comparing the level of fit of each of the theories against the individual constraints.

Method The non-dominated models generated in the third experiment, optimising models from multiple theories, are analysed. For each theory, we obtain the performance of its non-dominated models on each constraint. The average level of fit is then computed, and selected results are presented in Table 11. The bottom two rows of Table 11 give the mean and standard deviation in the scores across all constraints, for each theory.

Discussion The table gives the best and worst AAD constraint scores for each of the four theories. Notice that the AAD 7th constraint is the best performing constraint for the context theory, but the worst for the connectionist theory. As is clear, and as Nosofsky (2000) also discussed, different theories do better in different experiments. This reflects theoretical differences between the different models; some theories will be better suited to some kinds of experiments than others.

One interesting point is the low AADs for the connectionist and particularly the discrimination-network theory of models. The low AADs imply that the search process has found models which

do almost as well as possible for this class of models across all constraints. In contrast, the AADs for the context models vary considerably across the constraints, from 0.056 for 'AAD 7th' to 0.226 for 'AAD 24th'; this suggests that it is hard to find context models which perform very well on all constraints simultaneously. In general, the connectionist and discrimination-network classes of models have half the AAD of the other two types.

Also worth observing is that the low AADs are achieved for the more general classes of theories; both the connectionist and discrimination-theory theories are applicable to a wide range of problem types, whereas the mathematical theories are restricted to this one categorisation problem, and the two kinds of mathematical model are specific to particular techniques of recognised concepts. In consequence, we suggest that our experiments support the value of general theories within cognitive science, and also the validity of looking for models which perform well on multiple constraints. This empirical result provides further support for the 'N=1' methodology, proposed by Gobet and Ritter (2000), where one theory is tested against multiple kinds of experiment at once.

5.4.3 MODEL DIVERSITY

We finally ask the question, how similar are the models created for each theory? Similarity of models can be judged by looking at the distribution of the values obtained for each free parameter in the theory.

Method The non-dominated models generated in the third experiment, optimising models from multiple theories, are analysed. For each theory, we obtain the range of values which each parameter in that theory takes, and compute its average value and standard deviation, recording the results in Table 12.

Discussion All of the parameters exhibit clear preferred values, with standard deviations indicating a certain amount of spread. Interestingly, the guessing parameter, γ, used in the two mathematical models has tended to similar values (0.04 and 0.06), and the probability of learning an example, ρ, used in the connectionist and discrimination-network theories has also tended to similar values (0.70 and 0.71). In particular, given that the discrimination-network theory has the least number of parameters, it tends to produce virtually identical models, as indicated by the small standard deviation. By contrast, the standard deviations can be fairly large with the context and prototype models, which have the most parameters.

6. Discussion and Conclusion

We have demonstrated how an evolutionary approach can create sets of non-dominated cognitive models considering multiple sources of experimental data. We have formalised this problem as one of multi-objective optimisation, and created a suitable optimisation algorithm, the speciated non-dominated sorting genetic algorithm, tailored to the development of cognitive models from multiple theories. Experiments have shown that good quality models are created, and that comparing models from multiple theories against multiple sets of data improves the quality of the models created.

At one level, creating a set of optimised models is important for determining which of two or more theories is providing a better explanation for the experimental data. In addition, the optimisation process creates large amounts of information about the range of candidate models generated by a theory. For example, a non-dominated set will typically contain 40-60 models, each of which is not out-performed on all the experimental constraints. We can potentially use this

	Parameter					
	Context					
	w_0	w_1	w_2	w_3	c	γ
mean	0.39	0.16	0.21	0.23	2.73	0.04
s.d.	0.14	0.12	0.12	0.12	0.60	0.04
	Prototype					
	w_0	w_1	w_2	w_3	c	γ
mean	0.45	0.10	0.26	0.19	1.31	0.06
s.d.	0.16	0.07	0.12	0.10	0.48	0.06
	Connectionist					
	θ	η	ρ			
mean	0.26	0.69	0.70			
s.d.	0.27	0.55	0.11			
	Discrimination-network					
	ρ					
mean	0.71					
s.d.	0.01					

Table 12: Mean and standard deviation for parameter values found in a set of non-dominated models.

information to identify the best models and examine the effect of multiple constraints in comparing and optimising theories. This information also offers new opportunities to understand the behaviour of models of different theories and to examine the role of their parameters.

Our approach is based on the assumption that it is useful to compare models against multiple constraints. Nosofsky (2000) raises the concern that certain theories will be more likely to explain data in some experimental settings than in others. For example, some experimental settings will suggest an exemplar approach to categorisation, whereas others encourage participants to use a prototype approach. It is therefore useful to look at whether theories do better in some experiments than in others. Our results in Table 11 indicate that each theory does produce better measures of fit on different constraints, supporting Nosofsky's concern.

However, we also found that the more comprehensive connectionist and discrimination-network theories provided a better overall level of fit across the constraints than did the more specialised mathematical theories. Our result provides support for the 'N=1' methodology, proposed by Gobet and Ritter (2000), where one theory is simultaneously tested against multiple kinds of experiment. The requirement to perform well on multiple experiments, where different approaches are suggested by the setting, will lead to the development and use of more comprehensive theories, which automatically adjust their internal processes to take into account the experimental setting. Gobet et al. (1997) describe how this can be done for one form of discrimination-network theory, EPAM, in the five-four task.

The evolutionary approach to developing models directly addresses some concerns that have been expressed about the use of formal modelling (Roberts and Pashler, 2000). In particular, by

showing how multiple theories can be compared on multiple constraints, we show that the problem of overfitting can be addressed by forcing the same theory to account for as much data as possible. This idea is not new, and is part of the argument for unified theories of cognition put forward by Newell (1990). As shown in the first experiment, the evolutionary approach can produce better models than a hill-climbing algorithm, due to the complexity of the fitness landscape. We also note that our approach complements work on model selection and model description, such as that discussed by Pitt et al. (2006); Pitt, Myung, and Zhang (2002).

In common with most treatments of multi-objective optimisation, we have treated the parameters as real numbers, and the cognitive models used in the experiments all have numeric, real-valued parameters. However, the approach taken here is not restricted to such parameters. A 'parameter', more generally, is a component of a model which has some kind of value; in computational terms, this value may be an integer, a floating-point number, a boolean, a category label or even a more general string description, amongst others. Category or string labels may be used to identify more qualitative kinds of model behaviour: an example might be the attention strategy to adopt when scanning a list of items (e.g. Feigenbaum and Simon, 1984). In most cases, parameters can be viewed as an item drawn from a set. For example, w_0 in the mathematical models is a floating-point number drawn from the set $[0, 1]$. Other models may have parameters such as *AttentionStrategy*, drawn from the set $\{$Anchor, Middle, Top-Down$\}$. The evolutionary techniques described in this article can operate on such parameters: the two major operations on parameters, to select a random value, and mutate, are easily implemented as a selection from the set of available values.

Our presentation of the problem of finding good models from multiple datasets and multiple theories as a multi-objective optimisation problem leads to a set of non-dominated models. Each of these models is not out-performed on all experimental constraints. This approach is not the only way to generate candidate models. In simple mathematical models, such as those used for the five-four task, some form of hill-climbing algorithm may be suitable. With more complex models, perhaps the most popular technique is to determine model parameters in one set of experiments and maintain the same parameter values in future experiments. For example, early work with EPAM (Feigenbaum and Simon, 1984) led to timing parameters for the operations of familiarisation and discrimination which were then used in subsequent experiments without reconfirming them each time: the approach is akin to solving a sequence of equations, one unknown at a time, e.g. see Richman, Simon, and Feigenbaum (2002).

A variation on this theme, requiring multiple datasets of roughly equivalent quality (such as the five-four data discussed above), would be to test whether models developed against a random selection of the datasets would generalise to the remaining datasets. This train/test scenario would be similar to that used in machine learning settings, and would be an interesting test of robustness. We have not done this in our experiments, as we are interested in the method of finding information about the parameters for models.

Whether developing a model on a subset of the data and testing it on the complement set is preferable to developing it on the full set is in itself an interesting question. When developing scientific theories to explain complex phenomena, it would seem suboptimal to use only one part of the data: coming up with a reasonable theory is hard enough. Indeed, when famous discoveries were made in science (e.g. discovery of the structure of the DNA by Watson and Crick), scientists did not discard half the data to later use them to test their theory. In fact, they were often desperately trying to grab all relevant data possible. Minimising the risks of overfitting and maximising generalisation of a theory is normally ensured in science in two related ways: first, by developing a theory that

covers a wide range of phenomena, and second, by collecting new data to test the theory. While the second way is beyond the scope of our approach, the first way is catered for by making it possible to optimise a theory using a wide range of tasks.

This article has focused on describing a method for optimising models from different theories across different sets of empirical data. We acknowledge a few limitations of our research. First, we have used relatively simple versions of the tested theories; for example, a connectionist model with more weights and a CHREST model with more free parameters could have been used. We intend to test the validity of our approach with more complex models in the future. Second, we only used numeric parameters. Qualitative parameters (e.g. strategies) are also used in cognitive modelling, and an interesting extension to our approach would be to develop methods enabling non-numeric parameters as well. A possible avenue would be to enable building blocks (e.g. learning algorithms) of the models to be evolved. Frias-Martinez and Gobet (2007), using genetic programming, have shown that this is possible with a simple task. Third, our method focuses on parameter optimisation, but does not address issues such as theory parsimony when comparing models. Whether and how methods developed in the field of model selection (Burnham and Anderson, 2002) can be incorporated to our approach is an interesting issue for further research.

There are alternative approaches to evolutionary algorithms to solve this multi-objective optimisation problem. Kase, Ritter, and Schoelles (2008) and Moore Jr (2011) propose a parameter-space exploration technique which has similarities to the approach proposed in this article. Kase, Ritter, and Schoelles (2008) also use an evolutionary technique, but only apply their technique to a single ACT-R model on a single experimental constraint. Moore Jr (2011) uses a *mesh* approach to approximate the parameter space in some detail, and again only demonstrates this technique using a single model and a single experimental constraint. In common with our proposal, the search is in parallel across the whole parameter space. However, the joint demands of the ACT-R model used and the mesh approach requires significant computing power: the results are generated on a HPC cluster with over 9,000 cores. The experiment in this article used relatively simple models, and runs within an hour on a standard desktop computer; in future work, we plan to explore the demands of more complex models such as ACT-R. With increases in computer speeds, multi-core processing and improvements in algorithm design, there will be many developments in this area extending the complexity of the search achievable on more complex models whilst keeping running time within reasonable time frames.

Although our example presented the development of a cognitive model of categorisation, the techniques should be of interest to theorists developing what Newell called "unified theories of cognition" (Anderson et al., 2004; Newell, 1990), in which more general theories are developed by bringing together models of specific phenomena. Automated generation of consistent parameter sets will assist the theorist in confirming consistency between models and so develop broader theories. Approaches such as this will require the development of online repositories of quality empirical evidence, as discussed in Gobet and Lane (2005) and Myung and Pitt (2010). In further work, we aim to test and refine the evolutionary algorithms presented here on a more diverse set of models, and more complex kinds of data.

Appendix: Implementation and software

Software implementing the algorithms, models and experiments reported in this article can be found in source form at: `https://github.com/petercrlane/evolving-models`

The software has been implemented in Java. The optimisation algorithms are collected together into a library, which may be used for working with other models and datasets. A detailed example, following the presentation in this article, is included, along with a graphically-driven demonstration program. Some downloadable files are available at
`http://peterlane.info/methodology.html`

References

Anderson, J. R.; Bothell, D.; Byrne, M. D.; Douglass, S.; Lebière, C.; and Qin, Y. L. 2004. An integrated theory of the mind. *Psychological Review* 111(4):1036–1060.

Burnham, K. P., and Anderson, D. R. 2002. *Model selection and multimodel inference*. Springer.

Coello, C. A. C. 2000. An updated survey of GA-based multiobjective optimization techniques. *ACM Computing Surveys* 32:109–143.

Coello, C. A. C. 2003. Recent trends in evolutionary multiobjective optimization. In Abraham, A.; Jain, L.; and Goldberg, R., eds., *Evolutionary Multi-Objective Optimization*. London, UK: Springer-Verlag. 7–32.

Cooper, R. P., and Shallice, T. 1995. Soar and the case for unified theories of cognition. *Cognition* 55:115–49.

Cooper, R. P.; Fox, J.; Farringdon, J.; and Shallice, T. 1996. A systematic methodology for cognitive modelling. *Artificial Intelligence* 85:3–44.

Cooper, R. P. 2002. *Modelling high-level cognitive processes*. Mahwah, NJ: Erlbaum.

Feigenbaum, E. A., and Simon, H. A. 1984. EPAM-like models of recognition and learning. *Cognitive Science* 8:305–336.

Fisher, D. H.; Pazzani, M. J.; and Langley, P. 1991. *Concept formation: Knowledge and experience in unsupervised learning*. San Mateo, CA: Morgan Kaufmann Publishers.

Frias-Martinez, E., and Gobet, F. 2007. Automatic generation of cognitive theories using genetic programming. *Minds and Machines* 17:287–309.

Gluck, K. A.; Staszewski, J. J.; Richman, H.; Simon, H. A.; and Delahanty, P. 2001. The right tool for the job: Information-processing analysis in categorization. In Moore, J. D., and Stenning, K., eds., *Proceedings of the Twenty-Third Annual Conference of the Cognitive Science Society*, 330–335. Mahwah, NJ: Lawrence Erlbaum.

Gluck, K. A.; Stanley, C. T.; Moore, L. R.; Reitter, D.; and Halbrugge, M. 2010. Exploration for understanding in cognitive modeling. *Journal of Artificial General Intelligence* 2:88–107.

Gobet, F., and Lane, P. C. R. 2005. A distributed framework for semi-automatically developing architectures of brain and mind. In *Proceedings of the First International Conference on e-Social Science*.

Gobet, F., and Ritter, F. E. 2000. Individual data analysis and Unified Theories of Cognition: A methodological proposal. In Taatgen, N., and Aasman, J., eds., *Proceedings of the Third International Conference on Cognitive Modelling*, 150–57. Veenendaal, The Netherlands: Universal Press.

Gobet, F., and Schiller, M. 2011. A manifesto for cognitive models of problem gambling. In Kokinov, B.; Karmiloff-Smith, A.; and Nersessian, N. J., eds., *European Perspectives on Cognitive Sciences – Proceedings of the European Conference on Cognitive Science*. New Bulgarian University Press, Sofia.

Gobet, F., and Waters, A. J. 2003. The role of constraints in expert memory. *Journal of Experimental Psychology: Learning, Memory & Cognition* 29:1082–1094.

Gobet, F.; Richman, H.; Staszewski, J.; and Simon, H. A. 1997. Goals, representations, and strategies in a concept attainment task: The EPAM model. *The Psychology of Learning and Motivation* 37:265–290.

Gobet, F.; Lane, P. C. R.; Croker, S. J.; Cheng, P. C.-H.; Jones, G.; Oliver, I.; and Pine, J. M. 2001. Chunking mechanisms in human learning. *Trends in Cognitive Sciences* 5:236–243.

Goldberg, D. E. 1989. *Genetic Algorithms in Search Optimization and Machine Learning*. Reading, MA: Addison-Wesley.

Grant, D. A. 1962. Testing the null hypothesis and the strategy and tactics of investigating theoretical models. *Psychological Review* 69:54–61.

Gunzelmann, G. 2008. Strategy generalization across orientation tasks: Testing a computational cognitive model. *Cognitive Science* 32:835–861.

Holland, J. H. 1975. *Adaptation in natural and artificial systems*. Ann Arbor: The University of Michigan Press.

Hsu, C.-W.; Chang, C.-C.; and Lin, C.-J. 2003. *A Practical Guide to Support Vector Classification*. Available from `http://www.csie.ntu.edu.tw/~cjlin/libsvm/` (accessed February 2013).

Kase, S. E.; Ritter, F. E.; and Schoelles, M. 2008. From modeler-free individual data fitting in 3-d parametric prediction landscapes: A research expedition. In *Proceedings of the 30th Annual Conference of the Cognitive Science Society*, 1398–1403. Austin, TX: Cognitive Science Society.

Lane, P. C. R., and Gobet, F. 2003. Developing reproducible and comprehensible computational models. *Artificial Intelligence* 144:251–63.

Lane, P. C. R., and Gobet, F. 2005. Multi-task learning and transfer: The effect of algorithm representation. In Giraud-Carrier, C.; Vilalta, R.; and Brazdil, P., eds., *Proceedings of the ICML-2005 Workshop on Meta-Learning*.

Lane, P. C. R., and Gobet, F. 2012. A theory-driven testing methodology for developing scientific software. *Journal of Experimental and Theoretical Artificial Intelligence* 24:421–56.

Leahy, K. 1994. The overfitting problem in perspective. *AI Expert* 9:35–36.

Maneeratana, K.; Boonlong, K.; and Chaiyaratana, N. 2004. Multi-objective optimisation by co-operative co-evolution. In Yao, X.; Burke, E.; Lozano, J. A.; Smith, J.; Merelo-Guervs, J. J.; Bullinaria, J. A.; Rowe, J.; Tino, P.; Kabn, A.; and Schwefel, J.-P., eds., *The Eighth International Conference on Parallel Problem Solving from Nature*, volume 3242 of *Lecture Notes in Computer Science*, 772–81. Springer Verlag.

Medin, D. L., and Schaffer, M. M. 1978. Context theory of classification learning. *Psychological Review* 85:207–238.

Medin, D. L., and Smith, E. E. 1981. Strategies and classification learning. *Journal of Experimental Psychology: Human Learning and Memory* 7:241–253.

Medin, D. L. 1989. Concept and conceptual structure. *American Psychologist* 44:1469–1481.

Moore Jr, L. R. 2011. Cognitive model exploration and optimization: A new challenge for computational science. *Computational and Mathematical Organization Theory* 17:296–313.

Murphy, G. L. 2002. *The big book of concepts*. Cambridge, MA: The MIT Press.

Myung, I. J., and Pitt, M. A. 2010. Cognitive modeling repository. In *Proceedings of the Thirty-Second Annual Meeting of the Cognitive Science Society*, 556. Portland, Oregon: Lawrence Erlbaum.

Newell, A., and Simon, H. A. 1972. *Human Problem Solving*. Englewood Cliffs, NJ: Prentice-Hall.

Newell, A. 1990. *Unified Theories of Cognition*. Cambridge, MA: Harvard University Press.

Nosofsky, R. M. 2000. Exemplar representation with generalization? Comment on Smith and Minda's (2000) "Thirty Categorization Results in Search of a Model". *Journal of Experimental Psychology: Learning, Memory and Cognition* 26:1735–1743.

Pew, R. W., and Mavor, A. S., eds. 1998. *Modeling human and organizational behavior: Applications to military simulations*. Washington, D. C.: National Academy Press.

Pitt, M. A.; Kim, W.; Navarro, D. J.; and Myung, J. I. 2006. Global model analysis by parameter space partitioning. *Psychological Review* 113:57–83.

Pitt, M. A.; Myung, J. I.; and Zhang, S. 2002. Toward a method of selecting among computational models of cognition. *Psychological Review* 109:472–491.

Richman, H. B., and Simon, H. A. 1989. Context effects in letter perception: Comparison of two theories. *Psychological Review* 3:417–432.

Richman, H. B.; Simon, H. A.; and Feigenbaum, E. A. 2002. Simulations of Paired Associate Learning using EPAM VI. Complex Information Processing, Working Paper #553.

Ritter, F. E.; Shadbolt, N. R.; Elliman, D.; Young, R. M.; Gobet, F.; and Baxter, G. D. 2003. *Techniques for Modeling Human Performance in Synthetic Environments: A Supplementary Review*. Wright-Patterson Air Force Base, Ohio: Human Systems Information Analysis Center.

Ritter, F. E.; Schoelles, M. J.; Quigley, K. S.; and Klein, L. C. 2011. Determining the number of model runs: Treating cognitive models as theories by not sampling their behaviour. In Rothrock, L., and Narayanan, S., eds., *Human-in-the-loop simulations: Methods and practice*. London: Springer-Verlag. 97–116.

Ritter, F. E. 1991. Towards Fair Comparisons of Connectionist Algorithms through Automatically Optimized Parameter Sets. In Hammond, K. J., and Gentner, D., eds., *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society*, 877–881. Hillsdale, NJ: Lawrence Erlbaum.

Roberts, S., and Pashler, H. 2000. How persuasive is a good fit? A comment on theory testing. *Psychological Review* 107:358–367.

Rumelhart, D. E., and McClelland, J. L., eds. 1986. *Parallel Distributed Processing*, volume 1 and 2. Cambridge, MA: MIT Press.

Schaffer, J. D. 1984. *Some experiments in machine learning using vector evaluated genetic algorithms*. Ph.D. Dissertation, Vanderbilt University, Nashville.

Schaffer, C. 1993. Overfitting avoidance as bias. *Machine Learning* 10:153–178.

Simon, H. A., and Gobet, F. 2000. Expertise effects in memory recall: Comments on Vicente and Wang. *Psychological Review* 107(3):593–600.

Smith, J. D., and Minda, J. P. 2000. Thirty Categorization Results in Search of a Model. *Journal of Experimental Psychology: Learning, Memory and Cognition* 26:3–27.

Srinivas, N., and Deb, K. 1994. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation* 2:221–248.

Stewart, T., and West, R. 2010. Testing for equivalence: A methodology for computational cognitive modelling. *Journal of Artificial General Intelligence* 2:69–87.

Tor, K., and Ritter, F. E. 2004. Using a Genetic Algorithm to Optimize the Fit of Cognitive Models. In *Proceedings of the Sixth International Conference on Cognitive Modeling*, 308–313. Mahwah, NJ: Lawrence Erlbaum.