Modelling variations in human learning in probabilistic decision-making tasks

Dominic Jacques Maurice Hunt

Goldsmiths, University of London

PhD in Psychology

Declaration of Authorship

I, Dominic Jacques Maurice Hunt, hereby declare that this thesis and the work presented in it is entirely my own. Where I have consulted the work of others, this is always clearly stated.

Signed: Date: 3rd March 2020

Ethical Declaration

The secondary data analyses and modelling of existing unpublished datasets, which are presented in this thesis, were approved by the Department of Psychology Ethics Committee, Goldsmiths, University of London.

Acknowledgements

I would like to express my gratitude to my supervisor, Prof. Alan Pickering for his guidance. His perspective, challenging questions, insights, ideas and support have shaped and tempered this PhD.

Thanks to Carlos Marques, Sevil Ince, Jessica Campbell, Alexia Lonnoy, Nicola Orton, Ian Tharp and Luke Smillie for use of their collected data, without which this thesis would not have been possible.

Thanks to my friends, especially Elizabeth, Martin, Jon and David for bearing with my obsession.

Thanks to my parents for their patience, support and perspective.

Thanks to my dearest wife, Maria Cristina, who has lightened my load and kept me grounded time and time again.

Abstract

This thesis focused on evaluating the capacity of models of human learning to encapsulate the action choices of a range of individuals performing probabilistic decision-making tasks.

To do so, an extensible evaluation framework, Tinker Taylor py (TTpy), was developed in Python allowing models to be compared like-for-like across a range of tasks. TTpy allows models, tasks and fitting methods to be added or replaced without affecting the other parts of the simulation and fitting process.

Models were drawn from the reinforcement learning literature along with a few similarly structured Bayesian learning models. The fitting assumed that the same model was used throughout a task to make all the choices.

Using TTpy, significant uncertainty was found in parameter recovery for short, simple tasks across a range of models. This was traced back to significant overlap in the action sequences plausibly produced by different combinations of parameters. Replacing softmax with epsilon greedy, as the way of calculating the action choice probabilities, was found to improve parameter recovery in simulated data.

Datasets from three existing unpublished probabilistic decision-making tasks were examined. These datasets were chosen as they contained information on extraversion for all their participants, their tasks were well established, and the tasks had a gains-only promotion focus. Only one of the three tasks provided models where most of the model participant fits had strong evidence that they were better fits than uniform random action choices.

In light of the difficulties in parameter recovery for individual participants, the unusual step was taken of averaging the recovered parameters across a subset of the best performing and most consistently recovered models within the same family. A significant correlation was found between this learning rate parameter and the participant extraversion measure when the softmax parameter variance was taken into account.

Table of Contents

TABLE OF FIGURES

TABLE OF TABLES

1 Overview

Agents learn from their environments: they update their beliefs about the world by integrating new environmentally-derived information with their prior knowledge (Piaget, 1937). How this is done will vary from agent to agent, with multiple factors coming into play, such as past experiences, risk aversiveness, sensitivity of senses and many others. The ability to describe this behaviour, and its variation across individuals would be a powerful tool for understanding not only how individuals take in information, but equally how they choose to react to their environment. To provide the clearest descriptions, providing a mathematical expression of it, allows us to represent the variations in learning across individuals by way of variations in model parameter values. By expressing the methods through which learning might occur in a mathematical form, these models can be tested to see both what their behaviour would be in different circumstances, and how they compare to real-world behaviour.

This thesis focuses on evaluating models of human learning for probabilistic decision-making tasks: tasks where participants learn from feedback or rewards resulting from specific actions and stimulus cues. The most frequent modelling approaches have focused on variations of reinforcement learning (Sutton & Barto, 1998). However, some studies comparing the performance of models have shown Bayesian models to perform better (Stankevicius, Huys, Kalra, & Seriès, 2014). In this work, simple models from both approaches are evaluated and compared, with a focus on reinforcement learning methods. The capacity of these models to explain participant responses is evaluated for tasks with a range of different stimulus and action choice complexities. During their evaluation, issues were identified in recovering accurate model parameters from participant data. Some progress was made towards resolving these issues before a final evaluation of the models was made across the available datasets.

To perform the model evaluations, a computational framework was built, written in the programming language Python (Oliphant, 2007), allowing like-for-like comparisons between the models. For this to be achieved, it was necessary to describe a common set of features for both the models and the tasks the models would be fitted against. For the models, the core of these come from our understanding of the brain.

1.1 Basis for models

Learning from events involves numerous neurological systems: sensory, memory, motor and cognitive. Learning how to respond in a probabilistic decision-making task depends upon updating the predictions of action consequences. The phasic activity of midbrain dopaminergic neurons has been shown to resemble a *Reward Prediction Error* (RPE) signal (Schultz, 2000; Schultz, Dayan, & Montague, 1997; Schultz & Dickinson, 2000). An RPE signal is positive for unexpectedly large rewards and negative for unexpectedly small rewards, which in a dopaminergic neuron equates to a brief increase or decrease in the firing rate relative to the *tonic* level. When the synaptic strengths accurately encodes the expected reward from action consequences, the RPE signal is zero (Glimcher, 2011). A detailed review of dopamine and reward can be found in Schultz (2015).

Unfortunately, to date, within an individual the encoded value of a reward as it reaches the dopaminergic pathways cannot be directly measured, only inferred from observable behavioural choices (Schultz, 2016). These values can be shown to be subjective and transitive (Lak, Stauffer, & Schultz, 2014), allowing us to consider them to be consistent across the duration of a task for an individual, but preventing us from assuming that all individuals will treat them the same way.

The RPE signal allows for more than just a simple updating of the action-consequences. For example, the medial prefrontal cortex has separate excitatory and inhibitory pathways corresponding to positive and negative RPE (Matsumoto, Matsumoto, Abe, & Tanaka, 2007), suggesting that updating may differ for positive and negative consequences. Alongside predictions of action consequences, it is likely that the RPE is used to evaluate the level of uncertainty in the state of the environment (Behrens, Woolrich, Walton, & Rushworth, 2007).

Prior to the RPE signal, midbrain dopaminergic neurons can be seen to encode a salience signal (Schultz, 2016), before reward value has been fully assessed. The models examined here assume that salient stimulus cues, available actions and rewards have been identified and encoded before reaching the model. This allows us to focus on predicted action consequences, choice of actions and updating of an expected reward given an action.

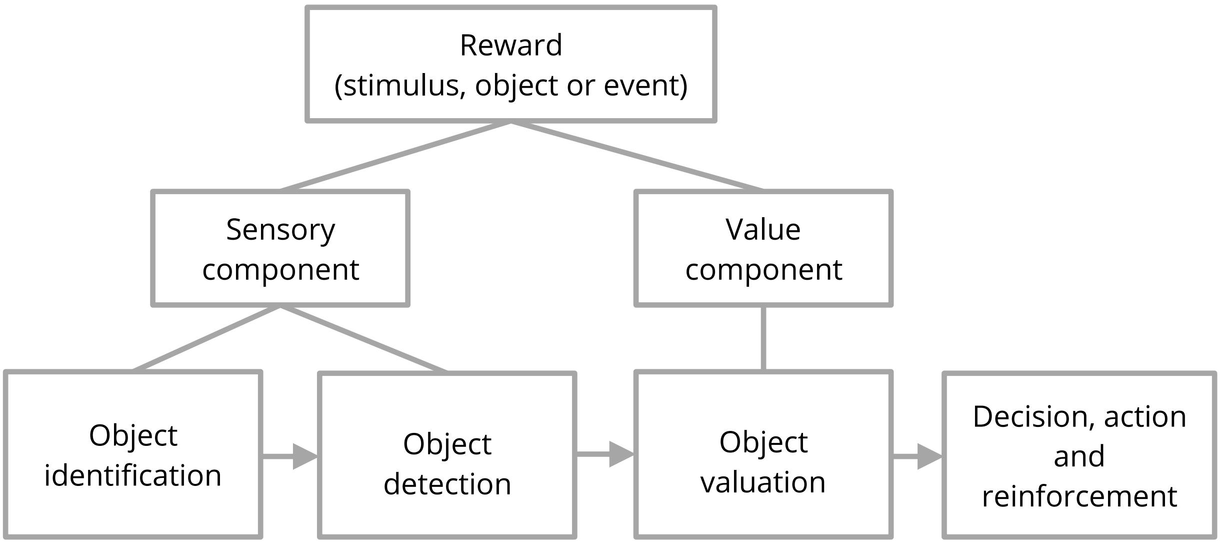


Figure 1-1 Reward component breakdown, as described in Schultz (2016). The models discussed in this thesis only examine the final box "Decision, action and reinforcement"

This updating of action consequences has been shown to vary between individuals (Smillie, Cooper, & Pickering, 2011) as well as experience changes as people age (Sojitra, Lerner, Petok, & Gluck, 2018). It is therefore preferable not to aggregate results across individuals. Furthermore, these differences can be related to *phenotypes*, a term I will use to refer to measurable properties of an individual that vary slowly, if at all (Pickering & Pesola, 2014). These may be genetic in origin, but not exclusively. Examples of phenotypes that have been examined in this context are tendency to jump to conclusions (Cafferkey, Murphy, & Shevlin, 2013; Moore & Sellen, 2006; Ziegler, Rief, Werner, Mehl, & Lincoln, 2008) and extraversion (Cooper, Duke, Pickering, & Smillie, 2014; Pickering & Pesola, 2014).

Extraversion is associated with gregariousness, assertiveness, leadership, sociability, high life satisfaction and impulsiveness (Wilt & Revelle, 2016). This has been shown to be widely generalisable across cultures (McCrae & Allik, 2002). Links between extraversion and reward learning were first proposed by Gray (1970) and later developed through several lenses, described in detail by Smillie (2013), such as incentive motivation (Depue & Collins, 1999) and reinforcement sensitivity theory (Smillie, Pickering, & Jackson, 2006). Extraversion is generally measured through the use of self-report questionnaires such as the Eysenck Personality Questionnaire, EPQ (H. J. Eysenck, 1975). Any assessment of the relationship between extraversion and reward learning are therefore harder to tease out, as extraversion is based on the outcomes rather than any possible inputs to reward learning (Smillie et al., 2006; Zuckerman, 2005). This is detailed in the review paper by Wacker & Smillie (2015).

Holroyd and Coles (2002) proposed that the RPE might modulate an electroencephalographic (EEG) signal from the medial frontal cortex ~200-300ms after a feedback event, known as feedback-related negativity (FRN). Potts et al. (2006) used a passive rewarding task to test this, where participants observed trials with a sequence of two cues followed by a reward. The first cue predicted the second cue 80% accurately and the second cue determined the reward 100% of the time. Both the first and second cues were of the same form: either a gold bar, the cue for a reward, or a lemon, the cue for no reward. Participants initially would be expected to show FRN when the reward is shown but, once the relationship between the second cue and the reward was established, the FRN would occur after the second cue is presented.

By examining the FRN during this task, Potts et al. found that its response was consistent with the phasic dopamine cell firing after a reward prediction error of midbrain dopaminergic neurons (Schultz, 1998), with a positive response to unpredicted rewards and a negative response when a reward did not occur as predicted.

The amplitude difference between the response to an unpredicted reward and the response to the absence of a predicted reward, known as a FRN difference wave or Reward Positivity, can be used as a measure of overall RPE magnitude, as decreases in the magnitude of the FRN difference waves correlate with decreases in errors in the reward predictions (Eppinger, Kray, Mock, & Mecklinger, 2008).

The link between the FRN difference waves and extraversion was examined by Smillie et al. (2011), who used the same passive task developed by Potts et al. (2006). They found that the FRN difference wave was stronger for high extraverts, more than one standard deviation above the mean score, than for low extraverts, also known as introverts, with scores more than one standard deviation below the mean.

This result was strengthened by Cooper et al. (2014) who found a positive correlation (r=.36) between extraversion scores and the size of the Reward Positivity. These findings were consolidated in a subsequent study (Smillie et al., 2019) replicating the previous study with a larger sample, 100 participants compared to 25, and once again finding a positive correlation (r=.26, p= .005), indicating that Reward Positivity may be at least partly modulated by extraversion. This in turn would suggest that extraversion could partly modulate the magnitude of the RPE.

This thesis examined unpublished datasets of probabilistic reward learning tasks of varying complexity where a standard questionnaire measure of extraversion had been collected for each participant. This allowed an exploration of which aspect of RPE-based reinforcement learning might be related to extraversion.

1.2 Considerations for modelling human learning behaviour

With the brain basis outlined above, it is possible to combine these with more computational considerations to produce our requirements for modelling the probabilistic decision-making tasks. Models of human learning can be evaluated by their capacity to reproduce the responses made by participants (Daw, 2011; Daw & Doya, 2006). In order to be able to identify learning within a participant's actions, the tasks must repeatedly present similar, simple situations, allowing both the models and participants to build up an understanding of the underlying statistical structure of rewards. The tasks should also contain many potential choice sequences so that each participant can be uniquely identified by their choices. For the situations to be simple, they must be *Markovian*, i.e. independent of each other, such that the current stimulus cues and available actions completely capture the probabilities of different consequences for each action (Haykin, 2009).

Any models that are to be considered ideally must be able to have their behaviour modified to span the range of human decision-making behaviour, such as those described in chapter 1.1. This would be achieved using parameters that can express this diversity of learning behaviour, while also modelling a given person's behaviour accurately using the same parameter values across a range of tasks. For us to be able to compare models across tasks, we assume that any participant properties represented by model parameters are stable over short durations.

The models must also be flexible in their design, allowing them to be applied to a variety of different types of tasks. Equally, models should also have the potential to be extendable, so that they can be applied to tasks of different levels of complexity; for example, tasks with a different number or type of stimuli, or where a reward is or is not provided. Any model must also be computationally feasible by brain-like systems. In this way, when the complexity of the task is increased there will be less chance of finding parts of the model parameter-space whose responses diverge from those provided by humans. This final requirement leads to the idea that the model should not only be able to represent the variation in human learning, but that the model parameters should be relatable to phenotypes. From this, we are led to look for models that would be capable of being implemented in the brain and map to identifiable structures in the brain.

Modifiable behaviour	To represent the range of human decision-making behaviour
Flexible design	Adaptable to different stimuli and decisions
Components map to brain-like structures	To maximise the chance of mapping to human behaviour across different task complexities.

Table 1-1: The core requirements for models to be considered along with its reason for inclusion.

As the models will be compared to the decision-making performance of participants, the focus of the models will be on those that can provide action choices based on stimuli. Equally, to limit the complexity of these models, it was decided to limit the models to those that are *model-free*: models where only action values are learnt and not the structure of the task (Beierholm, Anen, Quartz, & Bossaerts, 2011; Hampton, Bossaerts, & O'Doherty, 2006). Both model-based and model-free are known to coexist (O'Doherty, Lee, & McNamee, 2015), with mechanisms in place to decide which takes priority at any given time (S. W. Lee, Shimojo, & O'Doherty, 2014) and there are indications that even for simple tasks model-based components are used in decision making (Dayan & Berridge, 2014). In addition to this, while it is plausible that multiple learning models are running in parallel in different brain systems, to reduce the complexity of the fitting it was assumed that each participant used only one learning model throughout their task run, but it was not necessary to assume that all participants used the same model.

Having established these requirements, there are a wide range of models that could be considered: cognitive architectures (Sun, 2008), reinforcement learning (Sutton & Barto, 1998), Bayesian models (Jones & Love, 2011) and neural networks (J. X. Wang et al., 2018) among others. It was decided to focus on reinforcement learning models along with some Bayesian models that could be directly compared.

1.3 Outline of thesis

This project aimed to develop tools for comparing the performance of probabilistic decision-making models. The comparisons were performed with existing and to-be-collected data gathered, from a series of probabilistic reasoning and learning tasks. The primary research question was to identify the most appropriate and powerful approach to modelling task performance, and its variation across individuals.

The comparison of models across tasks and participants in an unbiased way, was achieved in two ways: by using comparison metrics that consider the different model complexities and by using consistent tools for all evaluations, modifying the setup as little as possible when moving from one evaluation to another. For this, a computational framework has been written in a way that allows a broad range of models, experiment tasks and evaluation methods to be swapped in or out without affecting the other parts. A unified way of interfacing the models to tasks was implemented and applied to a range of different models, allowing their different features to be compared more directly. This framework was implemented in accordance with the recommendations of Eglen et al. (2016). It is described in detail in chapter 2 along with the approaches used to fit the models to participant data.

Potential models found in the existing literature were evaluated for their ease of generalisation. Those that looked promising were translated into a common mathematical form described in chapter 3. The models were then implemented within the Python framework and validated, if possible, against either other implementations of the same model or results from a published simulation. The implementation also involved modifying the models such that they could be applied to other, previously examined, experiments, allowing the model to be fitted to the data from those experiments. As data from new tasks became available, previously implemented models were extended to be compatible with any new task features and then fitted to any new data. Finally, comparisons could be made between models across experiments and data sets. The models in their final form are presented in chapter 3.

During this process, simulated participant datasets were generated with a few of the models to test the fitting process. This highlighted some issues with parameter recoverability, with causes found both in fitting procedures and the inherent recoverability of some models. Most notably, it was found that the use of a SoftMax function to estimate action choice probabilities results in a significant loss of information, hampering parameter recoverability. These issues were discussed in chapter 4.

Datasets from various kinds of experiments were available for this project, both from previous student projects at Goldsmiths and student project data from our collaborators at other universities in Greenwich and Melbourne. Tasks were limited to gains only promotion focused tasks. A promotion-focus provides participants with a motivation to win points over the course of the experiment by providing potential higher payoffs. Gains only refers to participants receiving no losses as part of the task, just rewards and non-rewards. This avoids any need to disentangle possible mechanisms for reward and punishment processing (Schultz, 2007).

One task that was initially examined was the “beads tasks” (see Moore & Sellen, 2006). In this task, participants are told about two jars, that contain white and black beads, for example, Jar A is 85% black and Jar B is 85% white. In each trial, the participant sees the colour of a bead drawn with replacement from one of the two jars. Participants must then indicate their confidence that the beads are drawn from jar A. This task essentially requires participants to compute the posterior expectation of a black bead given the series of beads displayed to date, with the confidence that the beads are drawn from jar A being a direct function of this expectation. An initial exploration found that the information provided by participants during this task was insufficient to uniquely identify model parameters to participant responses. The fitting of this data was therefore abandoned.

A task with a slightly more complex reward was subsequently examined. The *Decks task* is a modified version of the one used by Worthy, Maddox, & Markman (2007), and similar to the IOWA gambling task (Bechara, Damasio, Damasio, & Anderson, 1994). Participants were presented each trial with two stimuli on a screen, one red and one blue. These were said to be the top cards of two decks of cards 80 cards long. In each trialstep participants choose a deck to take a 'card' from. They are then shown the value of the card. Each card has a predetermined reward associated with it, whose value was between one and ten. The objective was to maximise the accumulated reward values. For this task, three sets of participant data were available. The results of this dataset are discussed in chapter 6.

To evaluate the performance of models on tasks where the stimuli change, the *Weather task* was used, a task based on one described by Gluck & Bower (1988) and later adapted by Knowlton, Squire, & Gluck (1994). It asks participants to associate a series of cues with one of two outcomes. One to three cue cards, from a set of four cards, are presented to the participant in each trial. The participant must decide which one of the two possible outcomes the displayed cards are most likely associated with. Once the participant decides, they are then told if they were correct or not. The cues each have a probabilistic relationship with the two outcomes, with this version of the task having a novel probabilistic relationship, with the probability of an outcome varying depending on the combination of cues displayed. In the first phase of the task, the *learning phase*, participants are given feedback on whether their choice was correct. In the second phase, the *testing phase*, participants are not given any feedback. In total, there were 56 trials in the learning phase and 14 test phase trials, with equal numbers of each of the 14 cue combinations in each task phase. For this task, three sets of participant data were available. The results of model fitting for these datasets is discussed in chapter 8.

For one of the Weather task datasets, participants were also asked to perform a final task, known as the *Probabilistic Selection task*, based on the task as described in Frank et al. (2007). For this task, participants are asked to learn the likelihood of being rewarded for six different actions, each given its own symbol. In the first phase of the task the actions are shown in three pairs with complementary reward probabilities that differ for each pair (80:20, 70:30, 60:40). Participants are asked to pick the most rewarding action, whereupon they are provided with a reward if there is one. In the second stage of the task, the participant is again shown pairs of symbols, but as well as repeating the original training pairs, there are novel pairs made of symbols that were unpaired in the first stage. The participants are not given any rewards in this second stage.

As the participants from this dataset are the same as one of the Weather task ones, a comparison was made between the parameters recovered from the Weather task fitting and the Probabilistic Selection task fitting. This comparison tested our assumption that the model parameters are associated with stable features of the participants. This is discussed in chapter 8.4.

Both the Weather task and the Probabilistic Selection task have multiple phases, within which different models might dominate for a given individual (Frank et al., 2007). The impact that selectively fitting might have on parameter recovery, irrespective of the model chosen is discussed in chapter 5.

	Changing stimulus cues	*Static stimulus cues*
Changing actions		Probabilistic Selection task
Constant actions	Weather task	Decks task

Table 1-2 The tasks upon which the models were evaluated, classified by their use of static or varying stimulus cues and static or varying action choices.

2 Thesis methodology

To compare and evaluate the performance of learning models in identifying causal links across events, it is necessary to take the mathematical descriptions of proposed models and write them as computer code. This code will need to be written in such a way that the model implementation can be used to fit participant data and act as if it were a participant performing a task. Ideally, as the models are used to fit participant data from across a range of tasks, it would be best to implement each model only once. In this way, there are likely to be fewer mistakes in the single implementation of each model than in multiple implementations, thereby allowing us to trust our results more. However, it does require the model to be written in such a way that it can flexibly adapt to a range of task types, increasing the complexity of the model implementation. It also increases the complexity of the code surrounding the model, as it will need to act as an interface between the task and the model or the participant data being fitted and the model. This work will therefore need model's to be implemented using a common structure and communicate with the tasks using a common interface.

One solution is to write a framework into which models, tasks and participant data can be placed and interact in a consistent way (Eglen et al., 2016; Poldrack et al., 2019). This can be done by writing a modular computer program such that the only parts that change are the ones that have been explicitly asked to change. This has two added benefits. As the models and tasks can be swapped without changing the rest of the program, both must use consistent methods to communicate with the rest of the program. While this does constrain their structures, it does encourage clarity and consistency in the way they are described in code. The other benefit is that by changing only small amounts of code each time, it becomes possible to clearly identify any differences between models or between tasks. This allows certainty in the information has been passed to and from the models.

2.1 Data generation and analysis framework

The framework used in this thesis, Tinker Taylor py (TTpy), is composed of a series of components, each designed to be modified independently of the others.

- Task implementation
- Model implementation
- Participant data loading
- Fitting method
- Structure for running an experiment with a model
- Structure for fitting models to participant data
- Structure for organising and managing all outputs

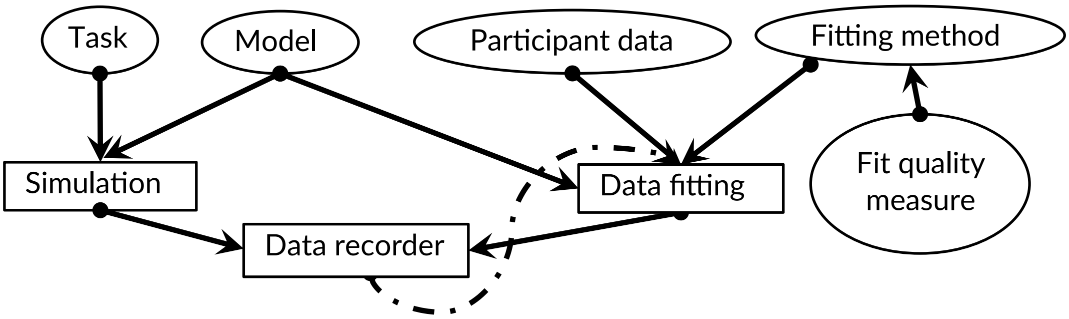


Figure 2-1 A diagram describing how the main components of the framework interact. Oval framed components denote easily swappable parts. All parts can be modified. Users can perform a simulation without fitting any data and data from a simulated task can then be fitted, as denoted by the dotted line.

These components have been created in such a way as to make it clear how to implement new versions that are compatible with all the existing components. While this makes each new component slightly more complicated to write, it allows existing components to interact with them immediately.

The framework has been written in the open source, interpreted, programming language *Python* (Millman & Aivazis, 2011). Python has been chosen for its clarity, its large number of packages, its availability on most operating systems, its already widespread use for scientific modelling, including in Psychology, as well as my pre-existing familiarity with the language. By writing the framework in Python any researchers who are interested in running it can be sure that it will be able to run on their computer. Also, when compared with other programming languages, they are likely to have an easier time understanding and modifying the code regardless of whether they are programming novices or those used to write in other programming languages.

The code has been made available on the website `bitbucket.org` (`https://bitbucket.org/djhunt/pyHPDM`) alongside documentation written both as comments in the code but also as a set of webpages. The documentation can also be found at `https://pyhpdm.readthedocs.io`.

The framework relies heavily on the scientific python, *SciPy,* library of packages (Oliphant, 2007). The choice of Python allows access to a wide variety of libraries written with scientific data analysis in mind. The libraries are largely platform independent, allowing researchers using different computer systems to collaborate and validate each other's work. They are also mature and well maintained, being regularly updated by many companies and volunteers. Documentation for the framework is incorporated into the code and is written in such a way as to be easily extracted into a set of searchable web pages using the *Sphinx* library (Brandl et al., 2018). Tests for the code, to verify that it works as expected, can be performed in two ways. Firstly, as models and experiments are implemented, the results from previous papers can be replicated. Secondly, formal tests can be implemented using the *pytest* library (Krekel, 2017). The framework has been written using Python version 2.7, as the later Python versions 3.* did not have all the necessary packages when this project was started.

Before running a simulation or data fitting, each of the necessary components is initialised and these initialised components are then passed to the simulation/data fitting routine. To aid with replication, this initialisation and passing of components is typically written in a file that can be stored with the output.

2.1.1 Keeping track of each simulation/fitting

So that anyone using the framework can understand what happened during the running of the framework and the progress of the program, a set of recorder and displayer functions are provided in what is known in Python as a *module*. They manage the saving, storing, logging and displaying of data from all parts of the framework. These functions are accessible through a recorder object (technically a *class instance*) that is initialised during the initialisation phase and then passed to the simulation or fitting modules. It is designed to provide a log of what went on during the simulation as well as record all the data and graphs that were produced. When correctly set up this provides, with very little user input, enough information to replicate the findings and enough detail to understand what went on during each task trial.

The model data is recorded in such a way that it can be treated as if it were participant data. This allows simulated data to be generated and treated as test ideal participants.

2.1.2 Task-model interactions

A participant's interaction with their environment in a repeated task can be thought of as being broken down into three components during each task trial, or *trialstep*: observation, action and consequences. This breakdown of a trialstep can equally be true for the interaction of a model designed to replicate the performance of a participant. To allow the models to be as general as possible, the interface for a model should be simple and flexible enough for a model to cope with trialsteps containing any combination of these three components, ideally without being explicitly told what to expect, e.g. without knowing if there will be consequences before an action is taken. In this way, the models will be able to learn from a range of response sets (Kirsch, Lynn, Vigorito, & Miller, 2004), such as classical and operant conditioning.

Observations can be thought of as the state of the environment, including the state of any salient cues or indications of possible actions that can be taken. Consequences can be thought of as either a representation of a reward, which can be numerically represented for the model, or feedback as to what was the correct action to take, or simply as a change of state in the environment, denoted by a change in the salient cues. For the sake of simplicity, consequences that are a change in the salient environmental cues will be considered as the observation for the subsequent trialstep. Action is the active selection of a choice from a series of explicitly signalled options. For a participant, the choices may be outlined before the start of a task if they do not change with each trialstep, at which point the only cue is one denoting when it is necessary to perform an action. We can examine all the possible combinations these elements would provide, as shown in Table 2-1. Here, the term “actionable” is used to refer to cues that signal that an action is available to be performed. Without these the model, or participant, would not know that an action was expected during the trialstep.

Observation	Action	Consequences
Stimulus		
Stimulus + actionable	•	
Stimulus + actionable	•	Reward
Stimulus		Reward
		Feedback/Reward
Actionable	•	Reward
Actionable	•	
Stimulus + actionable	•	Feedback
Actionable	•	Feedback

Table 2-1 The expanded list of all combinations of observations, actions and consequences that can occur in one trialstep. Two are greyed out as they cannot be distinguished from the others.

Two have been greyed out, as they are not useful here. Feedback/Reward on its own is identical to a stimulus on its own and actions without a stimulus or consequences cannot be learnt from, so can be ignored.

As we can consider feedback on what was the correct action to take as a form of reward, the term “reward” will be used to refer to both when no distinction needs to be made.

One difficulty that needed to be addressed when creating this interface, was *when* to update the model's expectations of action consequences. As the models and the framework are designed for repeated tasks, where the task can be split into a sequence of similar trialsteps, the final moment in the sequence of events for the previous trialstep is the first moment of the first events in the following trialstep. With that in mind, by condensing the list found in Table , as shown in Table 2-2, it becomes clear that when there is a reward within a trialstep, the model expectations can be updated when the reward is provided. In the other two cases, the model expectations can be updated when the next observation occurs.

Another issue to resolve is how to cope with observation-action pairs that stop getting rewards at the end, such as when there is a test phase in the task where no feedback is given to the participant. These need to be treated differently from tasks where the feedback is the next observation, i.e., the first or second rows in Table 2-2. To prevent this, a dummy feedback is provided, signifying that there were consequences, but that these are unknown. In this way, the model will not learn from the trialstep, but can still correctly understand the trialstep structure.

Event combination			*Point at which model knows enough to update expectations*
Observation	*Action*	*Consequences*	
Stimuli			Next observation
Stimuli + actionable	•		Next observation
Stimuli + actionable	•	Reward	Consequences
Stimuli		Reward	Consequences
Actionable	•	Reward	Consequences

Table 2-2 The event combinations for a trialstep and their respective expectation update times

2.1.3 Task descriptions

The aim of the way the tasks are structured is to be capable of simulating any repeated observational, action-response, observation-action-feedback or observation-reward tasks. To do so, each experiment task is written as its own module and relies on a task *template*. In programming terminology, the task class *inherits* from a task class template. A task can provide stimuli, indicate which actions can be taken and provide rewards. Its behaviour can be varied either through some internal sequence generation or based on actions taken by a model. Stimuli and rewards can have any number of components and can be instantaneous or have a duration, although this feature is never used in the tasks examined here. With this flexibility, it may be necessary to transform a stimulus or reward into a form that the model is expecting. To do so, task interface functions are used and are stored with each task. These are discussed further in chapter 2.1.4.

2.1.4 Model implementations

Models are implemented in a similar way to the tasks: with a module based on a class template. The models are designed to receive stimuli, rewards and participant action choices, and use these to update their reward expectations and action choice probabilities. They can also make decisions based on these evaluations.

Models have three parts split off from the core of the model: stimulus formatting, the decision making and the reward formatting. The motivation behind this is to separate the learning from the peculiarities of the task, allowing the model to be general and to be fitted into a range of different task types, even those the models were never designed for. The models looked at so far all explicitly or implicitly have these parts in a form that can be easily separated from the rest of the model. The stimulus formatting and the reward formatting are considered task-specific interface components. The decision making is much less likely to be task specific, but if a task requires an action only under certain circumstances this will need to be treated differently from those that expect an action for each trialstep.

The reward component receives the feedback from the task, as well as the model's chosen action. From this, the appropriate reward is constructed for the model. For example, as will be seen in chapter 3, some models need the rewards to be expressed in the range $[0,1]$, while others can cope with arbitrary valued rewards. Others consider task feedback from one action to be useful in updating their expectations for all the possible actions. Both requirements necessitate transformations in task feedback across different models.

The stimulus processing component is designed to take the task stimulus and transform it into the form expected by the learning modules. For example, the stimulus cues might be arranged into the same representation found in the memory of the learning modules. As there are potentially multiple different ways of representing the stimulus data for a given model, care must be taken to retain as much of the initial information as possible when transforming it for a model. For example, for a task with two possible stimulus cues that are mutually exclusive, they could be represented as a binary digit, with 0 representing one cue and 1 the other. They could equally have multiple digits, all switching from 1 to 0 as the stimulus cue changed or have some digits that kept a constant value regardless of the cue currently available. Alternatively, the difference between the cues could be stored, represented by their presence separately, with one digit for each cue, so cue 1 could be represented by 10 and cue 2 by 01. It would also be possible to assign a random sequence of digits to represent a cue, as is done with Semantic pointers (Eliasmith, 2013). This allows other cues to be identified and incorporated without modifying the structure of the learning. It also allows for identification of relationships or similarities between cues to be learnt.

Given the nature of the tasks being examined, the models implemented in the framework are passed stimuli with a distinct and binary digit for each possible stimulus cue in the task, with 1 representing the presence of the cue and 0 its absence.

	Cue 1	*Cue 2*
Binary	0	1
Distinct	10	01
Repeated	11	00
Redundant	11	10
Semantic pointers	010101011	110111011

Table 2-3 Different representations for two mutually exclusive stimulus cues.

The decision component receives the data relevant for a decision and then returns a decision in the form of the action to be taken and structured information on how likely different actions were. The information needed to make the decision can be the last chosen action or the likelihoods of each action having the largest expectation of reward. The method used to make the decisions can include choosing randomly between the possible actions, weighted by their likelihoods, choosing the currently most likely action, or choosing an action only once one of the possible actions exceeds a certain threshold of likeliness. For the tasks examined in later chapters, the decision choices will be based, unless otherwise specified, on randomly choosing between the possible actions, weighted by their likelihoods.

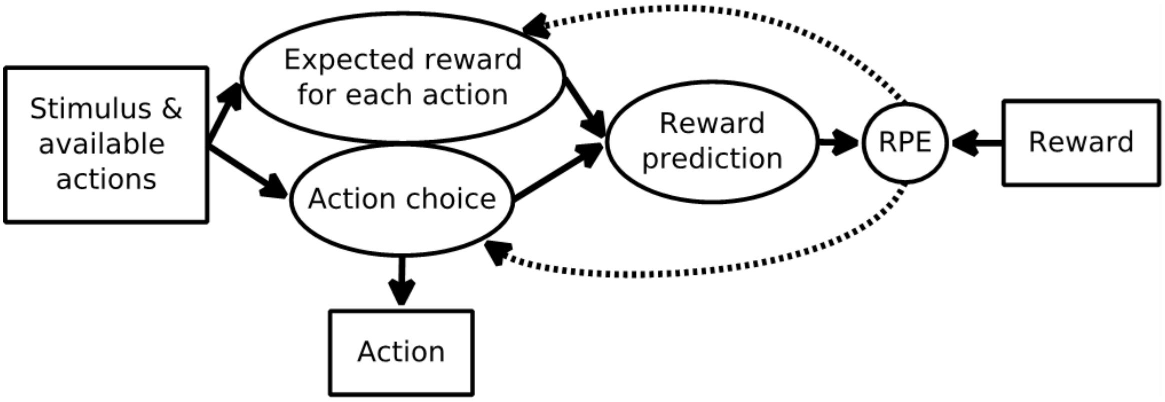
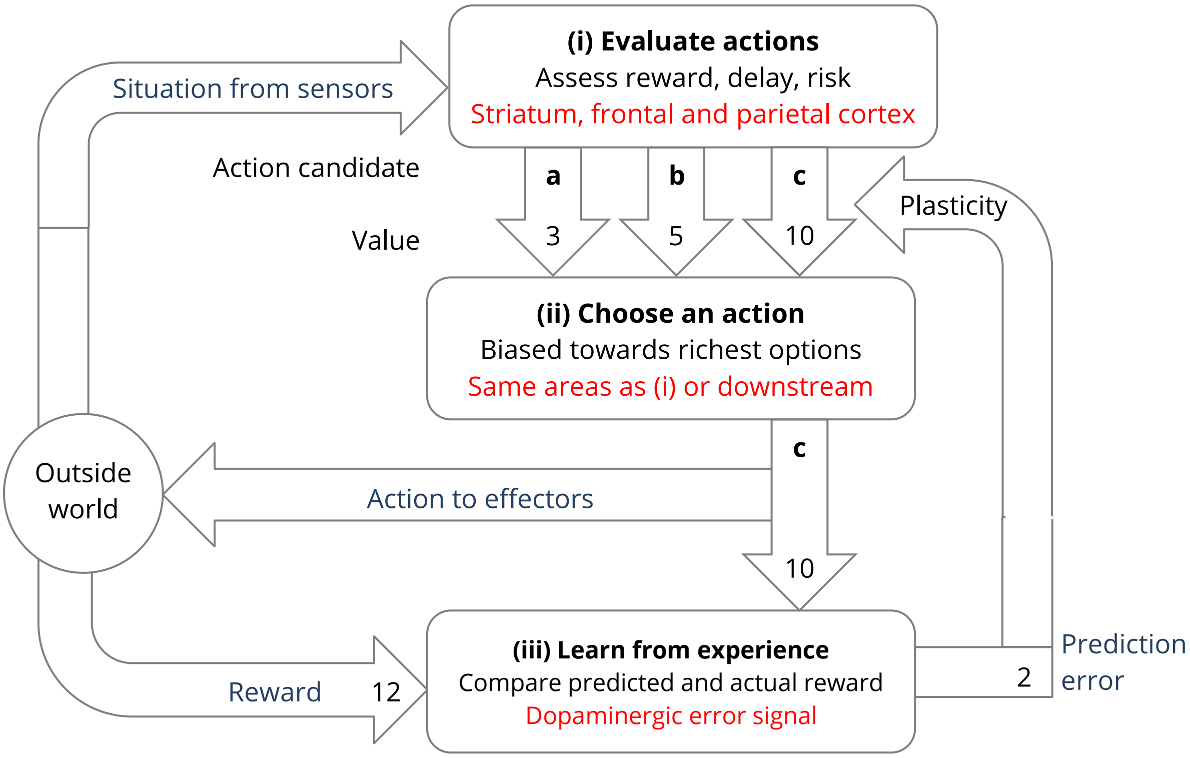


Figure 2-2 A flow chart showing the general structure of the models. Here the stimulus transformation, reward transformation and decision components are ignored. Rectangular boxes are used to denote interactions with the model's environment and ovals internal components. Dotted lines denote the integration of the Reward Prediction Error (RPE) from the current trialstep into the model's decision and prediction processes.

The core of the model has also been broken into a series of components, using the notion of a reward prediction error and inspired by similar breakdowns such as described by Schultz & Dickinson (2000) or Daw & Doya (2006) as shown in Figure 2-3. The general breakdown used within the framework and in subsequent chapters can be seen in Figure 2-2. The stimulus affects the choice of next action as well as the expected reward for each action. As the choice of next action may be dependent on the expected reward for each action, the two sets of calculations may overlap. Once an action is chosen and its expected reward has been calculated, the feedback from this, equated as a reward, is compared to the predicted reward. The result of this comparison, known as a *reward prediction error* (RPE) and often called a delta or δ, can then be used to update the values used to calculate the expected reward and the chosen action. These elements can be thought of as belonging to two broad categories: the actor and the critic. The actor chooses what action to take, based on the information it has. The critic evaluates how well reward predictions are matching up to actual rewards. In some models, an action is chosen based on the reward predictions, so there is no clear distinction between the actor and critic.

One final part that has been standardised across the models is the way in which the expected rewards are stored within each model. This is rarely explicitly discussed when presenting a model, so standardising this avoids adding another potential 'feature' to each model that could affect the performance. For most tasks, an expected reward will be stored for each action-stimulus cue pair, but this will


(i) Evaluate actions
Assess reward, delay, risk
Striatum, frontal and parietal cortex
Situation from sensors
Action candidate
Value
a
b
c
3
5
10
Plasticity
(ii) Choose an action
Biased towards richest options
Same areas as (i) or downstream
c
Outside world
Action to effectors
10
(iii) Learn from experience
Compare predicted and actual reward
Dopaminergic error signal
Reward
12
2
Prediction error

The three basic stages of many reinforcement learning accounts of learned decision-making. **(i)** Predict the rewards expected for candidate actions (here a, b, c) in the current situation. **(ii)** Choose and execute one by comparing the predicted rewards. **(iii)** Finally, learn from the reward prediction error to improve future decisions. Numbers indicate the predicted action values, the obtained reward, and the resulting prediction error.

Figure 2-3 An example of model breakdown, adapted from (Daw & Doya, 2006)

simplify for tasks where there is no variation in the stimuli, or no variation in the actions available for each trialstep, as shown in Table 2-4. This has been chosen as it is the simplest, memory efficient approach.

Event combination			What each stored expectation relates to
Observation	*Action*	*Consequences*	
Stimuli			Stimulus
Stimuli + actionable	•		Stimulus cue, action pair
Stimuli + actionable	•	Reward	Stimulus cue, action pair
Stimuli		Reward	Stimulus
Actionable	•	Reward	Action

Table 2-4 The event combinations for a trialstep and their respective expectation element meanings

2.1.5 Data for fitting

Data from past experiments can be imported and transformed into a common data format. Python has libraries to read most common data formats, including MATLAB .m files, XLSX, XLS and CSV. Tools were written using these libraries to transform the recorded data into a list of records, one for each participant.

Each participant's record would be stored as a *dictionary*, which is a collection of labelled bits of data. The data stored in these collections can be things as simple as the participant ID to a list of all of responses for the task. Currently, all the data is imported before the fitting of any participants begins.

Data from simulated participants can be read in using the same methods as those of real participants.

2.1.6 Fitting models to data

Fitting takes the sequence of events experienced by the participant and drives the model through them with a range of different parameter values. For each parameter combination a *fit quality measure* is used to transform this model experience into an assessment of how well the model, with the specific parameter values, would have mimicked the same reactions to the task as the participant. The lower the value the function returns, the better the fit and the closer the current model and its parameters are to describing the participant data (Akaike, 1974).

To provide the events experienced by the participant, the fitting process needs all the variables used to make the varying state experienced by the participant, as well as the responses of the participant. To evaluate how well a model fits these actions, we will also need to specify which variables processed by the model we want to use for its evaluation. The varying state experienced by the participant is composed of any stimuli, possible valid actions for the trialstep, the participant actions and any subsequent feedback. We can extract the data necessary for this from the recorded participant data. In certain tasks some of these will not change, such as the stimuli. These can be marked as being unchanging.

	Changing stimulus cues	*Static stimulus cues*
Changing actions		Probabilistic Selection
Constant possible actions	Biased coins Weather	Decks

Table 2-5 Examples of the tasks examined in this thesis and if they have varying stimuli and varying possible actions. This table ignores any counterbalancing that may occur with the presentation of the actions and cues to the participant. The tasks are described in detail in chapter 4.2 for the Biased coins task, chapter 6 for the Decks task, chapter 7 for the Probabilistic Selection task and chapter 8 for the Weather task.

If some trials are not considered representative, then these can be excluded from the fitting process. For example, the initial trialsteps in a task may be considered to not be representative of how the participant reacts to a task, as the participant may need some time to get used to the task.

As participant data for probabilistic decision-making tasks is inherently noisy, we would like any fit quality measure to be able to provide similar fit qualities for similarly likely action sequences and the same model parameters. This will allow us to minimise one source of error in identifying model parameters associated with participants from one single sequence of actions in a task. To what degree this is possible will be addressed in chapter 4.

2.1.6.1 *Making the model "walk in the participant's shoes"*

During the fitting, we wish to identify the model parameters that maximise the likelihood that the model would have taken the same action as the participant at each trialstep. To do so, the model performs the task with a numerical representation of the salient environmental information that the participant experiences: stimulus cues and possible actions it can take, followed by any feedback. The model is also constrained such that when it needs to make a choice, it makes its own choice, and then this choice is overruled such that it continues using the same action choice that the participant took in that trialstep. The performance of the model is then evaluated using a fit quality function based on the likelihood of the participant's actual choices for the model, with the specified parameters.

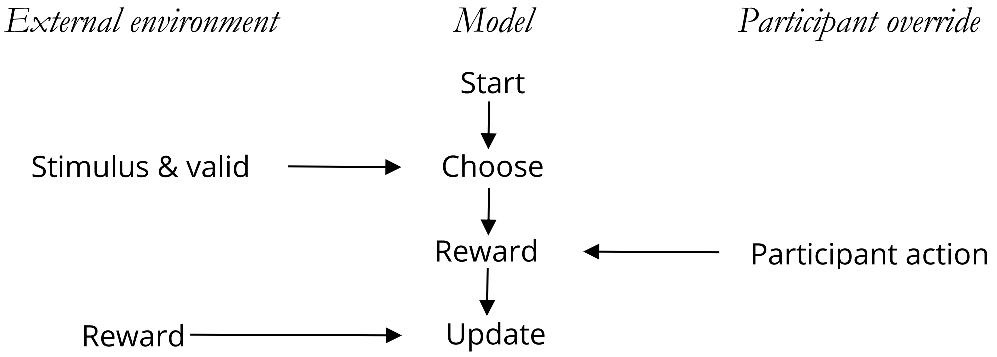


Figure 2-4 An overview of one trialstep in a task simulation during model fitting. The model is fed the external environment, using the same trial components as when the participant performed the task. Once the model has chosen an action it has its action overwritten with that of the participant.

2.1.6.2 *Fitting method*

When choosing the fitting method to use, only those that were implemented in well tested, maintained and documented codebases were considered. This was done to minimise the chance of there being any mistakes in their implementation, but also to increase the chance that they had been properly optimised to run as fast as possible. As the framework used for comparing the models is written in

Python, the SciPy Python libraries were used to provide implementations of the fitting algorithms. From these, two were investigated further; gradient descent and evolutionary fitting.

Traditionally, gradient descent methods have been used for fitting participant data to models (Sutton & Barto, 1998). These rely on calculating the direction of maximum gradient and following it until reaching a minimal point. SciPy provides suitable constrained fitters such as L-BFGS-B (Byrd, Lu, Nocedal, & Zhu, 1994), truncated Newton algorithm (Nash, 1984) and Sequential Least SQuares Programming (SLSQP) (Kraft, 1988). These provide similar results but tend to get into difficulties with different fits. The default fitting method has therefore been to try each fit using all the appropriate fitting algorithms provided. The best-fit parameters are then returned.

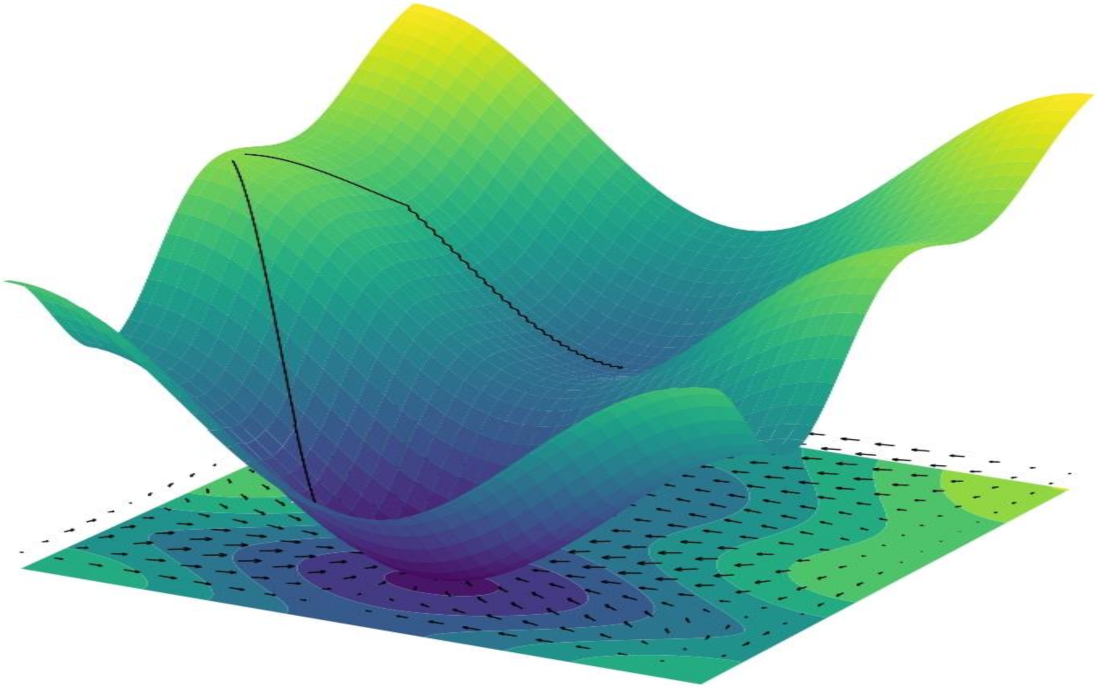


Figure 2-5 An example of a two-parameter space where a gradient descent search will not always find the global minimum. Here a function has its result shown as a position in the vertical axis. The function value is also shown as a colour scale, with dark purple being the lowest values and bright yellow the highest. There are two minima in the bounded region shown here with the one closer to the viewer being the lower of the two. By starting in some locations, the higher of the two minima will be found, but not the lower, global minimum. The two example trajectories, marked in black, show potential trajectories from two close starting points resulting in two different solutions.

Gradient decent methods have an inherent difficulty as they only follow one path through the parameter space. This makes their view of a complex parameter space narrow and may mean they miss a global minimum, as they identified a local minimum. One solution to this is to run the fit multiple times from different starting points. To increase the chance of finding the correct fit, a grid of starting parameters is used (Daw, 2011). Another issue is that gradient descent methods inherently require the fit quality to vary across the parameter space in a continuous way as well as requiring the gradient to also be continuous. This limits the tasks they can be used for.

An alternative approach to fitting is to use evolutionary algorithms (Salomon, 1998). They have the advantage that they make very few assumptions about the problem being optimised and can be used for fitting functions that are not locally smooth or, as often in our case, there are many local minima. The underlying idea is to iteratively sample a pre-chosen section of the parameter space, homing in on the best minima found. In each iteration, a set of points is randomly chosen from the parameter space, with the choice of locations weighted by the fit measure values of all previously selected places. In so doing, future points are more likely to be chosen clustered around previously identified areas with a good fit quality. New sets of points are generated until the variance between all points in the last round is below a specified threshold. As multiple potential solutions are looked at simultaneously, they share their information, and consequently it takes less time to perform the search than with gradient descent.

SciPy Python libraries have an implementation of an evolutionary algorithm based on differential evolution (Storn & Price, 1997). The implementation does not allow you to specify the initial fitting parameters, only the limits of the parameter space. You can also specify that a grid of initial parameters is used, covering the parameter space.

Unless specified, the fits in this study are performed with the evolutionary algorithm.

2.2 Fit quality measure

The aim for the fit quality is to capture in one value how well a model with specific parameters can characterise the behaviour of a participant performing a specific task. In so doing, we can draw from methods that have been designed to select a model from a range of models, as well as methods for representing how well a model represents a dataset (Burnham & Anderson, 2004).

To assess the model response, we use as our basis *Maximum Likelihood Estimation* of the probability that the model, for a set of parameters, would provide the same response as a participant. To calculate this, for each action of the participant takes, c_t, a likelihood can be calculated that the model would have taken the same action, $\mathrm{p}(c_t)$. For a sequence of T actions taken by the participant, $c_1, c_2 \cdots c_T = \mathrm{C}$, the combined likelihood of such a sequence for a given model is:

$$\mathrm{p}(c_1) * p(c_2) * \cdots * p(c_T) = \prod_{t=1}^{T} p(c_t)$$

2.1

By taking the log of this, the product of these probabilities can be transformed into a sum:

$$\mathcal{L} = \log_2\left(\prod_{t=1}^{T} p(c_t)\right) = \sum_{t=1}^{T} \log_2\big(p(c_t)\big)$$

2.2

Here, $\log_2$ is the base two logarithm, chosen to allow us to interpret the value more easily. The conventional method of representing these equations is to present them in the base of e. Changing to a representation in base 2 does not change the overall results. Only the magnitude of the fit values is changed, not their relative sizes. The effect of using base 2 is that when the model only had a 0.5 probability of choosing the action that the participant chose, $\mathrm{p}(c_t) = 0.5$, then $\log_2(0.5) = -1$. For a $\mathrm{p}(c_t) = 1$, then $\log_2(1) = 0$. For a $\mathrm{p}(c_t) = 0$, then $\log_2(0) = -\infty$. The more likely the model would be to take the same actions as the participant, the closer to zero the overall values are.

A more common form for this, and the one that will be used from now on multiplies this by a factor of -2, which provides some benefits in later calculations and makes all the values positive:

$$\mathrm{f} = -2\sum_{t=1}^{T}\log_2\big(p(c_t)\big) = -2\mathcal{L} \tag{2.3}$$

For this, more likely the model would be to take the same actions as the participant, the closer to zero the overall values are and the better fitting the model is. An example of what these sequences might provide is shown in Table 2-6. In this case, model 2, with an $\mathrm{f} = 11.39$, is the better model.

	t								f
Participant actions c_t	0	1	1	1	0	1	1	1	
Random $\mathrm{p}(c_t)$	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	16.00
Model 1 $\mathrm{p}(c_t)$	0.50	0.56	0.60	0.63	0.35	0.70	0.71	0.72	12.48
Model 2 $\mathrm{p}(c_t)$	0.50	0.91	0.95	0.95	0.05	1.00	1.00	1.00	11.39

Table 2-6 An example of how the fit quality values can vary across different models for the same sequence of actions. Here, model 2 matches the participant's actions best.

It is important to understand if the parameters providing the best fit found by the fitting process are significantly better than random. Using the structure above, we can describe the likelihood estimate for the pure random model by assuming that for a trialstep, t , each action available $d_t \in \mathcal{D}_t$, has equal probability of being chosen:

$$\mathrm{p}(d_t) = \frac{1}{\|\mathcal{D}_t\|}$$

with $\|\mathcal{D}_t\|$ being the number of different actions available at time t. When this is used in equation 2.3:

$$f_{\mathrm{rand}} = -2\sum_{t=1}^{T}\log_2\left(\frac{1}{\|\mathcal{D}_t\|}\right)$$

As T and $\mathcal{D}_t$ do not depend on the model, this will be constant when fitting a model to a set of data.

To compare these two we begin by calculating the likelihood of the sequence of actions C being created by the model we are testing, H_{mod}, rather than being a random sequence, H_{rand} (Kass & Raftery, 1995).

For H_{mod} the probability of H_{mod} given that the participant has performed the sequence of actions C, is called the *posterior probability.* From Bayes' theorem, this is defined as:

$$\mathrm{p}(H_{\text{mod}}|C) = \frac{p(C|H_{\text{mod}})p(H_{\text{mod}})}{p(C)}$$

Where $\mathrm{p}(C|H_{\text{mod}})$ is the model's *marginal likelihood*, the likelihood that for a given model, the sequence of actions C would be taken. $\mathrm{p}(H_{\text{mod}})$ is the model's prior probability. As we are only considering these two options as being the only options:

$$\mathrm{p}(H_{\text{rand}}) + \mathrm{p}(H_{\text{mod}}) = 1$$

From which we can rewrite the above equation's denominator as:

$$\mathrm{p}(H_{\text{mod}}|C) = \frac{p(C|H_{\text{mod}})p(H_{\text{mod}})}{p(C|H_{\text{mod}})p(H_{\text{mod}}) + p(C|H_{\text{rand}})p(H_{\text{rand}})}$$

By structuring $\mathrm{p}(H_{\text{rand}}|C)$ in the same way, we can now compare the two probabilities:

$$\frac{p(H_{\text{mod}}|C)}{p(H_{\text{rand}}|C)} = \frac{p(C|H_{\text{mod}})p(H_{\text{mod}})}{p(C|H_{\text{rand}})p(H_{\text{rand}})}$$

From which we can define the Bayes factor as a likelihood ratio of the prior and posterior odds (Kass & Raftery, 1995):

$$\mathcal{B} = \frac{p(C|H_{\text{mod}})}{p(C|H_{\text{rand}})} = \frac{p(H_{\text{mod}}|C)p(H_{\text{rand}})}{p(H_{\text{rand}}|C)p(H_{\text{mod}})} \quad 2.4$$

As the model has parameters, we can treat the probability of the data given to the model as a function of those parameters, θ:

$$p(C|H_{\text{mod}}) = \int p(C|\theta, H_{\text{mod}})p(\theta|H_{\text{mod}})d\theta \quad 2.5$$

Which is also the probability of that sequence of actions prior to any data being collected. We can also break down this probability as the product of probabilities of actions for each trialstep:

$$\mathrm{p}(C|H_{\mathrm{mod}}) = \prod_{t=0}^{T} p(c_t|c_{t-1}, \cdots c_1, H_{\mathrm{mod}})$$

This is similar to equation 2.1, but in this case the probability that an action is taken in a given trialstep is explicitly shown to have a dependence on the previously chosen actions. By taking the log, we can rephrase this in the form of the likelihood estimate for both hypotheses:

$$-2\log_2\big(p(C|H)\big) = -2\sum_{t=0}^{T} \log_2 p(c_t|c_{t-1}, \cdots c_1, H_{\mathrm{mod}}) = f \qquad 2.6$$

From this, we can also define a fit quality difference (Raftery, 1995):

$$\Delta f = f_{\mathrm{rand}} - f_{\mathrm{mod}} \qquad 2.7$$

We use an approximation of the probability that the data was produced by a given model, $\mathrm{p}(C|H_{\mathrm{mod}})$, known as the Schwarz Bayesian information criterion, but more commonly called the BIC (Raftery, 1995; Schwarz, 1978), to evaluate the fit quality of the model. This uses a Taylor series expansion to approximate the Bayes factor. If we assume that the model parameters are independent, the result is that the model's maximum likelihood estimation is corrected by the number of parameters in the model, $\Theta = \|\theta\|$, and the log of the number of trials, T.

$$f_{\mathrm{mod}} \cong BIC_{mod} = \Theta\log_2(T) - 2\sum_{t=1}^{T} \log_2\big(p(c_t)\big)$$

From this, an equivalent to the fit quality difference defined in equation 2.7 can be expressed as:

$$BIC_{diff} = BIC_{rand} - BIC_{mod}$$

Where $BIC_{rand} = f_{\mathrm{rand}}$ as the number of parameters in the random model, $\Theta = 0$.

The model with the highest posterior probability is the one that minimizes BIC_{mod}. As Θ is constant for the same model and $\Theta\log_2(T)$ is constant for the same task, when fitting a model to a task $\Theta \log_2(T)$will be a constant. By taking the $\log$ of the Bayes factor in equation 2.4:

$$-2\log_2(\mathcal{B}) = -2\log_2\left(\frac{p(C|H_{\text{mod}})}{p(C|H_{\text{rand}})}\right)$$

Expanding out the log:

$$-2\log_2(\mathcal{B}) = -2\log_2\big(p(C|H_{\text{mod}})\big) + 2\log_2\big(p(C|H_{\text{rand}})\big)$$

Where the right-hand side can be recognised as being in the form of the likelihood estimates in equation 2.6, combined to form the fit quality difference defined in equation 2.7:

$$2\log_2(\mathcal{B}) = \Delta f$$

This can also be expressed as:

$$\mathcal{B} = 2^{\frac{\Delta f}{2}}$$

2.8

When examining the response of more than one participant's performance, a Group Bayes Factor (GBF) can be used to provide a crude measure of the relative explanatory performance of two models for the $\mathcal{N}$ participants action sequences (Klaas E. Stephan, Marshall, Penny, Friston, & Fink, 2007). The GBF is the product of the Bayes factors for each participant:

$$\text{GBF} = \prod_{n=1}^{\mathcal{N}} \mathcal{B}_n$$

2.9

The probability associated with this Bayes factor can be calculated as an odds ratio. Using equation 2.4 and considering the prior probabilities for the two models to be equal:

$$\mathcal{B} = \frac{p(H_{\text{mod}}|C)}{p(H_{\text{rand}}|C)}$$

As these are the only two models being considered, $\text{p}(H_{\text{mod}}|C) = 1 - \text{p}(H_{\text{rand}}|C)$ (Kass & Raftery, 1995), so:

$$\mathcal{B} = \frac{p(H_{\text{mod}}|C)}{1 - p(H_{\text{mod}}|C)}$$

This can be rearranged to show that:

$$\mathrm{p}(H_{\mathrm{mod}}|C) = \frac{1}{1+\mathcal{B}^{-1}}$$

While a Bayes factor is more informative than a BIC value, a Bayes factor increases as the evidence for a model grows. For us to use a minimisation fitting technique, such as those described in chapter 2.1.6.2, a modified version will need to be used.

The simplest is to invert equation 2.8:

$$\mathcal{B}^{-1} = 2^{\frac{-\Delta f}{2}}$$

As we will be using the same Bayes factor criteria across all comparisons, the $\mathcal{B}$ we will use as a threshold will be a constant, $\mathcal{B}_{min}$. We can therefore express this as an inequality, describing a parameter fit that is sufficiently different from random.

$$1 > \mathcal{B}_{min} 2^{\frac{-\Delta f}{2}}$$

To allow us to evaluate easily across tasks, we wish to transform the inverted Bayes factor into a form that is independent of the number of trials being evaluated. To do so, we used a variation of the pseudo-R^2 described by Frank, Moustafa, Haughey, Curran, & Hutchison (2007). They evaluated their models using a pseudo-R^2 of the form:

$$R^2 = \frac{-\Delta f}{f_{\mathrm{rand}}} = \frac{f_{\mathrm{mod}} - f_{\mathrm{rand}}}{f_{\mathrm{rand}}} = \frac{f_{\mathrm{mod}}}{f_{\mathrm{rand}}} - 1$$

By transforming the Bayes factor in equation 2.8 to use this ratio we find:

$$\mathcal{B}^{\frac{-2}{f_{\mathrm{rand}}}} = 2^{\frac{-\Delta f}{f_{\mathrm{rand}}}} = 2^{\left(\frac{f_{\mathrm{mod}}}{f_{\mathrm{rand}}}-1\right)}$$

2.10

As before, this can be expressed this as an inequality, describing a parameter fit that is sufficiently different from random.

$$1 > \mathcal{B}_{min}{}^{\frac{2}{f_{\mathrm{rand}}}} 2^{\left(\frac{f_{\mathrm{mod}}}{f_{\mathrm{rand}}}-1\right)}$$

When BIC approximations are used, this becomes:

$$1 > \mathcal{B}_{min}{}^{\frac{2}{BIC_{rand}}} 2^{\left(\frac{BIC_{mod}}{BIC_{rand}}-1\right)}$$

Raftery (1995), considers a Bayes factor greater than[1] 20 as suggesting strong evidence for a model, and roughly equivalent to a probability of 0.95, although R. Wetzels et al. (2011) suggest that a Bayes factor of 20 is much stricter than this. Substituting the value of $\mathcal{B}_{min} = 20$ for the minimum Bayes factor that we will accept, we can now minimize the data using a normalised Bayes factor of:

$$f_{\mathcal{B}} = \mathcal{B}_{min}{}^{\frac{2}{BIC_{rand}}} 2^{\left(\frac{BIC_{mod}}{BIC_{rand}} - 1\right)} \tag{2.11}$$

In this form, a fit quality of 1 or below is equivalent to a $\mathcal{B}$ of $\mathcal{B}_{min}$ or higher. This can be rearranged by substituting equation 2.10 in to provide the $\mathcal{B}_{min}$ value corresponding to the $f_{\mathcal{B}}$ value:

$$\mathcal{B} = \mathcal{B}_{min} f_{\mathcal{B}}{}^{\frac{-BIC_{rand}}{2}}$$

As BIC_{rand} is the same across participants for the experiments examined here, the value of $\mathcal{B}$ can be calculated once the fitting has been completed.

One issue that we are not addressing here is that for the BIC to work, the statistical model must be regular, which is defined as a model whose mapping from model parameters to a probability distribution is one-to-one and whose Fisher information matrix is positive definite. Models that violate one or both conditions are called singular. Singular models cannot be approximated by a normal distribution, forcing us to look elsewhere for our assessment of model fit (Friel, McKeone, Oates, & Pettitt, 2017). One suggested alternative is the Widely Applicable Bayesian Information Criterion, WBIC (Watanabe, 2012), which is a generalisation of the BIC for all singular statistical models. However, it is common for modellers to use the BIC without considering this aspect.

[1] Due to the way in which they are defined, Bayes factors are the same irrespective of the base used for the exponents and logarithms. Therefore, e can be used in the place of 2 and $\log_e = \ln$ in the place of $\log_2$ without affecting the choice of threshold Bayes factor.

2.3 Participant data

The participant data discussed in chapters 6, 7 and 8 were collected as part of other research projects and repurposed for this thesis. The ethical approval for collecting the participant data analysed in this thesis was deemed as sufficient and approved by the Goldsmiths Psychology department Board of Ethics.

2.4 Data analysis

The participant behavioural data is initially assessed using task-specific measures to test that the participants have responded to the task manipulations and, if possible, these are comparable to prior published examples where the task was used.

Once participant data has been fitted, the models will be compared using several criteria. The simplest of these is the number of successful fits. Any fits that reach the parameter boundaries are considered failed fits, as a boundary fit removes a parameter from the model, transforming it into a simpler model, with the exception of the upper bounds of β, σ_α and σ_λ, which have been arbitrarily set sufficiently high that if a model has a best fit on these bounds, it is unlikely that the recovered model parameters accurately represent the learning method of the participant. To allow for numerical uncertainty from fitting, a boundary fit is considered to have occurred if a recovered parameter is within the smallest or largest 0.1% of a parameter's support, the range of values over which it spans.

Another criterion for evaluating the models is the quality of the fits, as described by the fit quality measures. This may take many forms, some of which are described in chapter 2.2 such as the log likelihood, f, in equation 2.3 or the normalised Bayes factor, $f_{\mathcal{B}}$, in equation 2.11. These can be compared, along with a participant group level evaluation using the Group Bayes factor, defined in equation 2.9. However, a fitting measure becomes a cruder model evaluation criterion when it is used to recover parameters (Daw, 2011; Strathern, 1997).

A more Bayesian approach to model comparison is discussed by Stephan, Penny, Daunizeau, Moran, & Friston (2009), who introduce a hierarchical Bayesian approach that calculates an approximate probability distribution of likely model frequency across participants. This is modelled using a Dirichlet distribution $\mathfrak{D}$, also used in the Bayesian models in chapter 3.11. The model likelihoods, ω_k, for each model, k, are converged upon using the log model evidence, $\mathcal{L}_{n,k}$, for each participant, n, as discussed in chapter 2.2. In our case this will be approximated using the BIC. As the log model evidence is used here as part of a larger formulation, the BIC must be constructed using the natural, or base e, logarithm. By using this and a starting assumption that the model likelihoods are all initially equal, $\omega_0 = [1, \cdots, 1]$, a stable ω can be calculated by iteratively recalculating until convergence:

$$u_{n,k} = \exp\left(\mathcal{L}_{n,k} + \Psi(\omega_k) - \Psi\left(\sum_k \omega_k\right)\right)$$

$$\varpi_k = \sum_{n}^{\mathcal{N}} \frac{u_{n,k}}{\sum_k u_{n,k}}$$

$$\omega = \omega_0 + \varpi$$

where Ψ is the digamma function. From this the posterior expectation of the model frequencies can be calculated:

$$EF_k = \mathbb{E}_k[\mathfrak{D}(\omega)] = \frac{\omega_k}{\sum_{i \in \mathcal{R}} \omega_i}$$

Implementations of this are found in the MATLAB VBA toolbox (Daunizeau, Adam, & Rigoux, 2014; Daunizeau, Friston, & Kiebel, 2009).

Parameters that should be similar across models will be assessed for the strength of their correlations across participants. An overall measure of correlation for a group of parameters can be calculated using Kendall's W, otherwise known as Kendall's coefficient of coefficient of concordance (Legendre, 2010). This is a rank-based correlation measure that compares sets of values and returns a measure of their ranked agreement between 0 and 1, with 0 indicating no agreement.

The fitted parameter values are also compared to other participant data collected, such as scores from the Eysenck Personality Questionnaire Revised, EPQ-R (S. B. G. Eysenck, H. J. Eysenck, & P. Barrett, 1985).

3 Models examined

One starting point for these models is to consider learning from the perspective of reinforcements of beliefs. The simplest and most computationally efficient models for reinforcement learning (RL) are based on reward prediction error (RPE) (Rosenblatt, 1958, 1961; Sutton & Barto, 1998). These rely on updating the expected outcome, based on the discrepancy between the expected reward value for an action, given the presence of a particular stimulus, and the actual reward of that action. More formally, at trial t the expected outcome for the next trial, E_{t+1} is calculated by updating the expectation from the current trialstep using the current reward, E_t

$$E_{t+1} = E_t + \alpha(r_t - E_t)$$

with α as the learning rate, set between 0 and 1 inclusive. Therefore, a constant expected reward $E_{t+1} = E_t$ is equivalent to an $\alpha = 0$ and $\alpha = 1$ results in an expected reward that matches the reward from the previous timestep, $E_{t+1} = r_t$ The simplicity of RL models is appealing, allowing for easy neuronal implementation (Rescorla & Wagner, 1972; Rosenblatt, 1961). However, when placed in undirected, delayed, real-world situations it can fail to identify causal links (Glimcher, 2011; Littman, 1994). Attempts to use RL models for some tasks can result in overly complex and rigid learning systems which negate the original advantages of RL (Sutton & Barto, 1998). Nonetheless, it is useful as the basis for many more detailed models, or in simple task contexts, such as those examined here.

One significant limitation with these RL models is that they do not take into account the uncertainty surrounding an expectation. An expectation can be thought of as the average of all possible rewards, weighted by the likelihood of those rewards. By not describing the uncertainty in the likelihood of rewards for a given action, an RL model is in effect using a point function, or Dirac-delta function, to describe the distribution of likely rewards for the given action. In other words, RL models assume that there is no uncertainty surrounding an expectation.

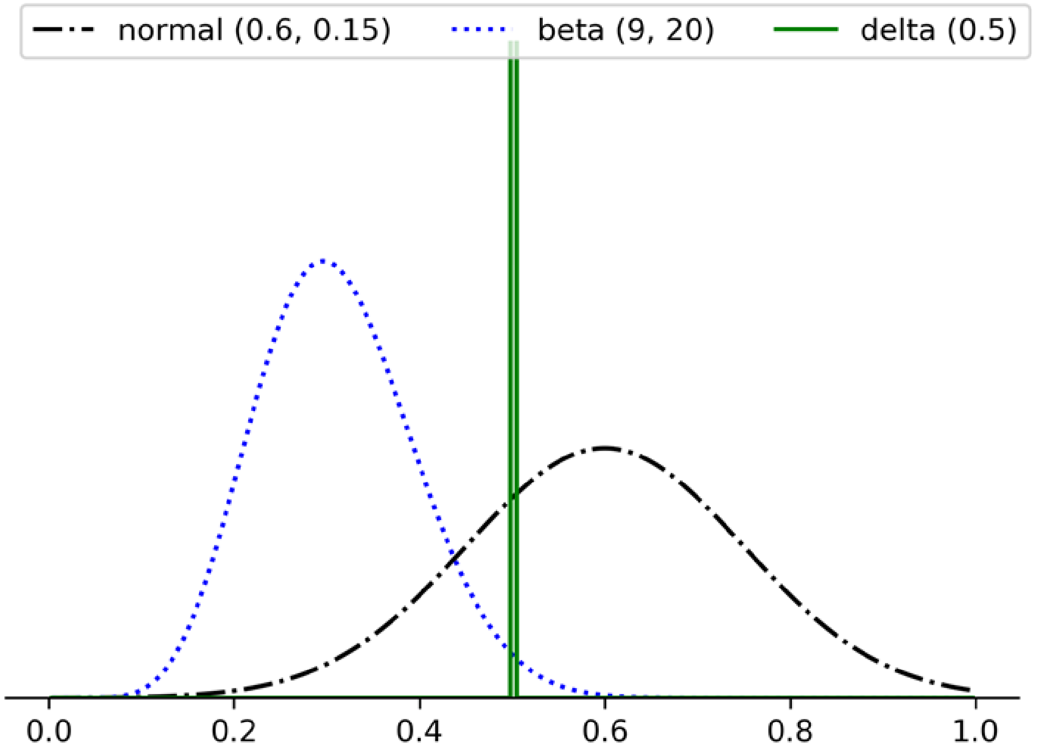


Figure 3-1 Three different representations of expectation based on different information. The Dirac delta function uses only one value and consequently has no uncertainty or tolerance for other possibilities. A normal distribution uses both the main value and a measure of uncertainty. The beta distribution uses the frequencies of each event to estimate a distribution of the event likelihoods

Another class of models addresses this limitation by updating the likelihood of rewards using Bayesian inference (e.g., Knill & Pouget, 2004). These, *Bayesian* learning models have been shown to predict human actions, but frequently involve evaluations of high-dimensional integrals that are computationally demanding and ill-suited to implementation in neuronal architectures. Furthermore, the simpler ones tend to be prescriptive, not allowing for individual variability (Jones & Love, 2011; Mathys, Daunizeau, Friston, & Stephan, 2011).

This chapter will introduce most of the models that are examined, using a common mathematical structure and notation. After introducing a model's features, a table will summarise the complete model. The models will be discussed and compared from a computational perspective. Further variations on these models were implemented as a response to the results of chapter 4. These are described in chapter 4.8.

3.1 Model notation

The expression of the models has been normalised in such a way that the same symbols are used for comparable concepts. All the models have been updated so that they can cope with arbitrary numbers of possible actions, arbitrary numbers of stimulus cues and arbitrarily large positive real rewards. These are all features necessary for one or more of the tasks whose participant data will be fitted, as described in chapter 2.1.5. For a full list of the symbols used in this thesis, along with their uses, see Appendix I.

The models are structured for tasks where participants are asked to learn causal links within repeated similar trials. Trials contain a description of the state of the pertinent environment, including the state of any stimulus cues and a description of which actions can be taken, as well as any reward from an action the participant may take during the trial. As these trials are considered to be self-contained, the models we will be examining will be model-free (Sutton & Barto, 1998). The term model-free is somewhat unclear, as it refers to whether the reward learning model builds a model of the task. That is to say, a model-free model will not attempt to identify causal relationships between sequences of trials, only between stimuli, actions and rewards within each trial. However, there is evidence that people identify causal links between trials, even when explicitly told that there are none (Plonsky, Teodorescu, & Erev, 2015).

To aid the comprehension of these models, the display of models themselves has been broken into the sections described in chapter 2.1.4. The *Reward expectation* calculates the expected reward for each action. The A*ction choice* calculates the probabilities of choosing each action, given the stimulus cues, and the chooses the action based on these probabilities. As the method of choosing of the action based on the action probabilities, P, has been separated from the models in the framework, as described in chapter 2.1.4, this is denoted in the model descriptions by the function $\mathcal{C}(P)$. The *Reward Prediction Error*, or *RPE*, calculates the discrepancy between the actual reward and the expected reward. The *Critic update* calculates a new expected reward for each action-stimulus cue pair. The *Actor update*, when there is one, calculates the new values used during the next trialstep to calculate the probabilities in the Action choice.

3.2 Q-LEARNING

One of the simplest of the reinforcement learning models is Q-learning (Watkins, 1989). This uses the discrepancy between the expected reward and the actual reward to update the expected reward for the chosen action and the active stimulus cues. The impact of the update is controlled through a learning rate parameter α, ranging between 0, no impact, and 1, which effectively replaces the expected reward with whatever the last reward was. At both of these extremes we can consider that we have another, simpler, model which contains no learning. The update is also split between the active stimulus cues, such that the change across all cues is equal to the update if there were only one cue. At time t, for each action d from the set of possible actions, $\mathcal{D}_t$, the expected reward for each action, V, is calculated by combining the expected rewards, E, for each of the active stimulus cues, s.

$$V_{d,\,t} = \sum\nolimits_{s \in S_t} s_t E_{s,d,t}$$

Here, for completeness, we have allowed the cues to have not just a state, but a magnitude, although this will not be needed in any of the tasks looked at in later chapters. The probability of choosing a given action, $P_{d,t}$, is calculated using the Softmax function, a generalisation of the logistic function and sometimes called a Boltzmann distribution.

$$P_{d,t} = \frac{e^{\beta V_{d,t}}}{\sum_{i \in \mathcal{D}_t} e^{\beta V_{i,t}}}$$

This uses an exploration-exploitation parameter, β, also commonly called the inverse temperature parameter or stochasticity parameter, to modulate the sensitivity to differences between expected reward values. If $\beta = 0$, all possible actions are equally likely to be chosen, regardless of any differences in expected rewards. In effect, there is no learning in this case. If β is very large, the action with the largest expected reward will be by far the most likely, however small the advantage it has over the other actions.

The updating of the expectation has been modified from that shown at the beginning of the chapter to allow learning to occur independently for different stimulus-cues. The learning rate is weighted by the magnitude of the stimulus cue, s_t, compared to the total magnitude of all the stimulus cues, $\|S_t\| = \sum_{s_t \in S_t} s_t$ resulting in an update expectation function of the form:

$$E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha s_t}{\|S_t\|} \delta_t$$

Where δ_t is defined as the difference between the reward received, r_t, and the expected reward for the chosen action, $V_{c_t,}$

$$\delta_t = r_t - V_{c_t,t}$$

The version proposed by Watkins (1989) adds to the reward prediction error a discounted maximum expected future reward given the stimulus cues of the following trialstep.

$$\delta_t = r_t - V_{c_t,t} + \gamma \max_d (V_{d,t+1})$$

This updates the expectation for the action chosen in the trialstep with the maximum expected reward for the following trialstep, weighted by a discount factor γ, ranging from between 0 and 1 inclusive. To do so, this is calculated as soon as the stimulus cues for the next trialstep are known and the necessary expected action rewards of the subsequent trial, $V_{d,t+1}$, have been calculated. The only modification to the Q-learning model is therefore to include a second expectation update equation immediately following the calculation of $V_{d,t+1}$.

$$E_{s_{t-1},c_{t-1},t+1} = E_{s_{t-1},c_{t-1},t} + \frac{\alpha \gamma s_{t-1}}{\|S_{t-1}\|} \max_d (V_{d,t})$$

This does not impact the choice of action for the trialstep that has just started, as the action probabilities are calculated based on the expected rewards used in this update equation.

Stages at t	Q-learning (qLearn)
Reward expectation	$V_{d,t} = \sum_{s \in S_t} s_t E_{s,d,t}$
Action choice	$P_{d,t} = \frac{e^{\beta V_{d,t}}}{\sum_{i \in \mathcal{D}_t} e^{\beta V_{i,t}}}$ $c_t = \mathcal{C}(P_t)$
RPE	$\delta_t = r_t - V_{c_t,}$
Critic update	$E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha s_t}{\lVert S_t \rVert}\delta_t : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	--

Table 3-1 The description of the Q-learning model, broken into the components used in the implementation.

Stages at t	Q-learning future (qLearnF)
Reward expectation	$V_{d,t} = \sum_{s \in S_t} s_t E_{s,d,t}$ $E_{s_{t-1},d,\,t+1} = E_{s_{t-1},d,t} + \frac{\alpha\gamma s_{t-1}}{\lVert S_{t-1} \rVert}\max_d\left(V_{d,t}\right) : d = c_{t-1}$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_{t-1}$
Action choice	$P_{d,t} = \frac{e^{\beta V_{d,t}}}{\sum_{i \in \mathcal{D}_t} e^{\beta V_{i,t}}}$ $c_t = \mathcal{C}(P_t)$
RPE	$\delta_t = r_t - V_{c_t,}$
Critic update	$E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha s_t}{\lVert S_t \rVert}\delta_t : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	--

Table 3-2 The description of the Q-learning future model, broken into the components used in the implementation.

3.3 Q-LEARNING WITH 2 LEARNING RATE PARAMETERS

RPE models can also be built to have separate excitatory and inhibitory pathways, in line with results described in chapter 1.1. Q-learning models can be adapted for this by using two learning rate parameters, α^+ and α^-, depending on if the RPE is positive or negative. When these are the same this simplifies to the Q-Learning model.

Stages at t	Q-learning with 2 learning rate parameters (qLearn2)
Reward expectation	$V_{d,t} = \sum_{s \in S_t} s_t E_{s,d,t}$
Action choice	$P_{d,t} = \frac{e^{\beta V_{d,t}}}{\sum_{i \in \mathcal{D}_t} e^{\beta V_{i,t}}}$ $c_t = \mathcal{C}(P_t)$
RPE	$\delta_t = r_t - V_{c_{t,}}$
Critic update	$E_{s,d,t+1} = E_{s,d,t} + \begin{cases} \frac{\alpha^+ s_t}{\Vert S_t \Vert}\delta_t & \delta_t > 0 \\ \frac{\alpha^- s_t}{\Vert S_t \Vert}\delta_t & \delta_t < 0 \end{cases} : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	--

Table 3-3 The description of the Q-learning model with two learning rate parameters, broken into the components used in the implementation.

3.4 OPAL

Collins & Frank (2014), proposed a way of modelling the ACC, by building on the idea of having separate excitatory and inhibitory pathways using RPE learning. This Opponent Actor Learning model (OpAL), shown in Table 3-5, uses simple reinforcement learning as a critic to calculate the RPE, with a learning rate of α_{Crit}. The actor is separated into two components, an excitatory (Go) and an inhibitory (Nogo) components, denoted G and N respectively in the equations. Both the excitatory and inhibitory components have the same structure and are both updated with each feedback using the RPE calculated in the critic. However, they respond differently to the feedback, with the excitatory path increasing for positive reward differences and the inhibitory component decreasing for the same difference. They also use different learning rates, α_G for the excitatory learning and α_N for the inhibitory learning. Both also use their current strength as a weighting for their own update, a form of update known as three-factor Hebbian update.

A version without the Hebbian element in the excitatory and inhibitory pathways was also created by Collins & Frank to demonstrate how the model would not work without it. They argue that without it their model cannot provide the same flexibility, nor account for the tendency for the excitatory and inhibitory components to discriminate between different action choice values over time. Having updated these excitatory and inhibitory components, they are then used to provide the likelihoods of actions.

Stages at t	Opponent Actor Learning without Hebbian update (OpAL_H)
Reward expectation	$V_{d,t} = \sum_{s \in S_t} s_t E_{s,d,t}$
Action choice	$A^*_{d,t} = \sum_{s \in S_t} s_t A_{s,d,t}$ $P_{d,t} = \frac{e^{\beta A^*_{d,t}}}{\sum_{i \in \mathcal{D}_t} e^{\beta A^*_{i,t}}}$ $c_t = \mathcal{C}(P_t)$
RPE	$\delta_t = r_t - V_{c_t,}$
Critic update	$E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha_C s_t}{\lVert S_t \rVert} \delta_t : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	$\left.\begin{array}{l} G_{s,d,t+1} = G_{s,d,t} + \frac{\alpha_G s_t}{\lVert S_t \rVert} \delta_t \\ N_{s,d,t+1} = N_{s,d,t} - \frac{\alpha_N s_t}{\lVert S_t \rVert} \delta_t \end{array}\right. : d = c_t$ $\left.\begin{array}{l} G_{d,t+1} = G_{d,t} \\ N_{d,t+1} = N_{d,t} \end{array}\right. : \mathrm{d} \neq c_t$ $A_{d,t+1} = (1 + \rho) G_{d,t+1} - (1 - \rho) N_{d,t+1}$

Table 3-4 The description of the OpAL model without Hebbian update, broken into the components used in the implementation.

Stages at t	Opponent Actor Learning (OpAL)
Reward expectation	$V_{d,t} = \sum_{s \in S_t} s_t E_{s,d,t}$
Action choice	$A^*_{d,t} = \sum_{s \in S_t} s_t A_{s,d,t}$ $P_{d,t} = \frac{e^{\beta A^*_{d,t}}}{\sum_{i \in \mathcal{D}_t} e^{\beta A^*_{i,t}}}$ $c_t = \mathcal{C}(P_t)$
RPE	$\delta_t = r_t - V_{c_t,}$
Critic update	$E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha_C s_t}{\lVert S_t \rVert} \delta_t : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	$\left.\begin{array}{l} G_{s,d,t+1} = G_{s,d,t} + \frac{\alpha_G s_t}{\lVert S_t \rVert} G_{s,d,t} \delta_t \\ N_{s,d,t+1} = N_{s,d,t} - \frac{\alpha_N s_t}{\lVert S_t \rVert} N_{s,d,t} \delta_t \end{array}\right. : d = c_t$ $\left.\begin{array}{l} G_{d,t+1} = G_{d,t} \\ N_{d,t+1} = N_{d,t} \end{array}\right. : \mathrm{d} \neq c_t$ $A_{t+1} = (1 + \rho) G_{d,t+1} - (1 - \rho) N_{d,t+1}$

Table 3-5 The description of the OpAL model proposed by Collins & Frank (2014), broken into the components used in the implementation.

This Hebbian update of OpAL can lead to instability, a point discussed in the appendix of (Collins & Frank, 2014). There they present a derivation demonstrating that this growth is bounded under stationary conditions, such that for G:

$$\log(G_{t+1}) < \log(G_{t=0}) + \frac{\alpha_G}{\alpha_C}(E_{t+1} - E_{t=0}) \qquad 3.1$$

However, when fitting this model this bounding was found to be insufficient, as the growth rate in the actor learning is still sufficient for many parameter combinations for OpAL to result in overflow or underflow errors when fitting participant data with 80 trialsteps. To illustrate the speed of the growth, in Figure 3-2, we can see the values of G resulting from growth from ten trialsteps in stationary conditions. Here, the same action is taken each time and the same reward, 0.5 is given. $G_{t=0}$ is set as 1. In the left graph we see how varying α_G / α_C can affect G, as result that could be expected from looking at equation 3.1. As under static conditions increases in G are marked by N tending towards zero, we can consider that:

$$A_{t+1} \approx \ (1+\rho)G_{t+1}$$

For a $\rho = 1$ and a $\beta = 1$ this would result, when estimating P in us calculating values of the order of $2 * 10^{17}$ after only ten trialsteps. While this does require the fitting to be examining high values of α_G, low values of α_C and $r_t/E_t = 10$, this is not uncommon when fitting tasks where the reward distribution varies across the task. In the right graph of Figure 3-2, we can see in more detail how the discrepancy between the reward and the expectation of the reward can lead to rapid changes in the values of G and N under stationary conditions.

It is also worth noting that because of the structure of this model, the performance assumptions made by Collins & Frank only work consistently with low values of r. As can be seen in Figure 3-3, for large reward values the growth of G is more chaotic and unstable. For this reason, rewards when fitting the OpAL model will be scaled to the range $[0,1]$.

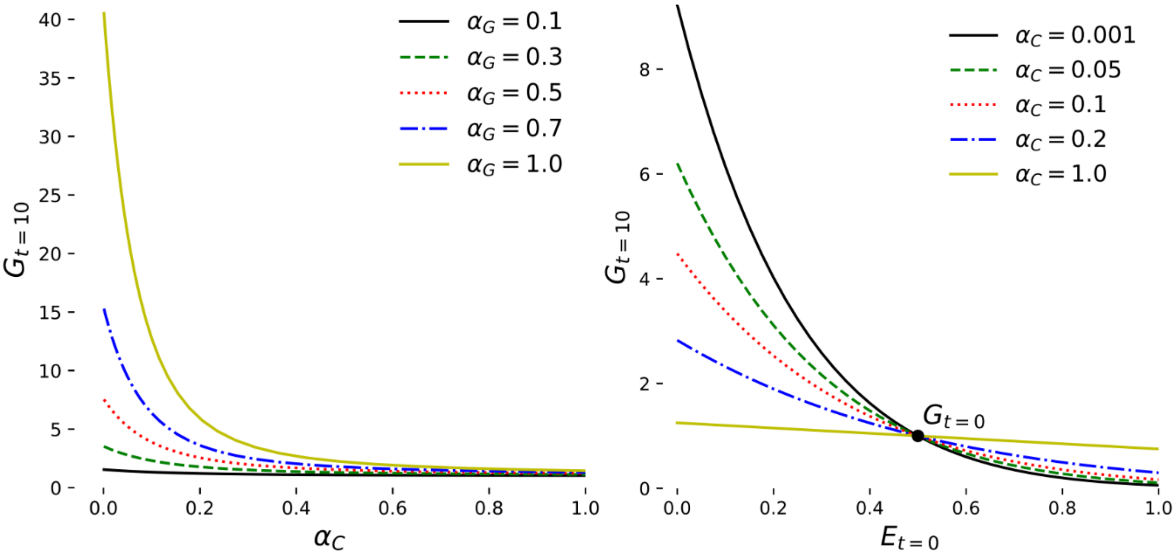


Figure 3-2 The OpAL values for G after the tenth successive trialstep with the same reward of 0.5. $G_{t=0}$ is set as 1. ***Left****: How the relationship between α_G and α_C affects the growth of G. $E_{t=0} = 0.05$. Right: How the value of $E_{t=0}$ affects the growth of G across a range of α_C with $\alpha_G = 0.5$.*

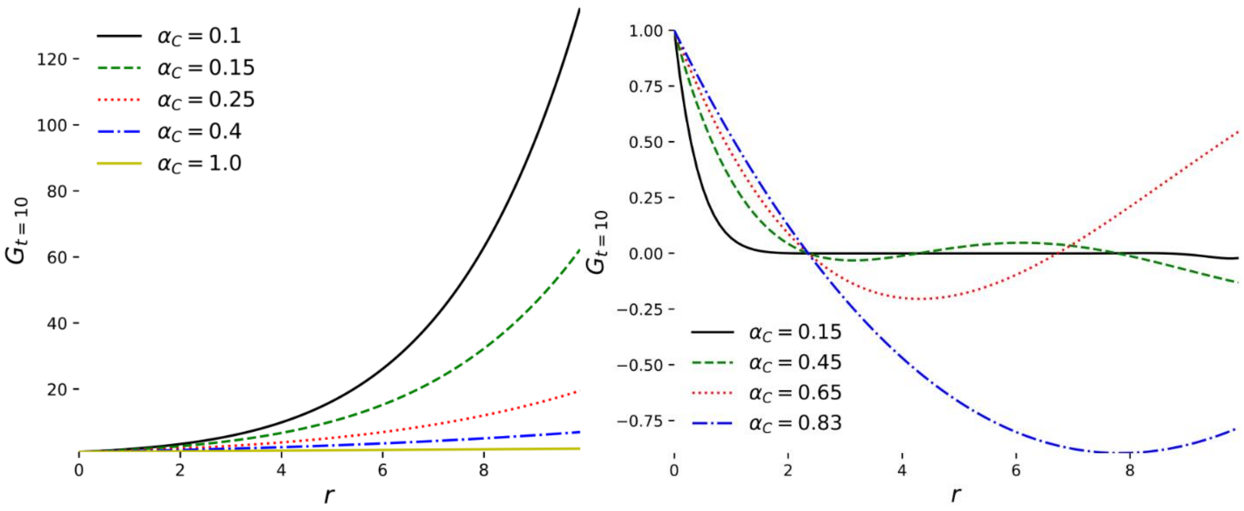


Figure 3-3 The OpAL values for G after the tenth successive trialstep with the same reward. $G_{t=0} = 1$ and $\alpha_G = 0.5$. ***Left***: $E_{t=0} = 0.8r$ ***Right***: $E_{t=0} = 1.85r$

To minimise the issues in fitting OpAL, while keeping its features, the model was modified to include an extra saturation term in the update of G and N:

$$\left(1 - \frac{G}{M}\right)$$

This model, OpALS, contains a new parameter in the saturation term, M, which acts as the largest value G and N can have, akin to including a maximal receptor occupancy. If this saturation term is to have a minimal impact on the model, then it must be as large as possible. This will therefore be a fixed value, dependent only on the implementation hardware and will not vary across participants. This results in it having a value around 50 in the Python framework. By taking the stationary model simulations shown for OpAL in Figure 3-2 and Figure 3-3 and reproducing them for OpALS with an $M = 10$, we can see in Figure 3-4 and Figure 3-5 that the saturation term does have the desired effect while not changing the dynamics of the model. A value for M of ten was chosen so that there would be some visible difference from the OpAL figures.

As OpAL-H does not have the Hebbian term, it does not have the instability of OpAL, and so does not need to be adapted.

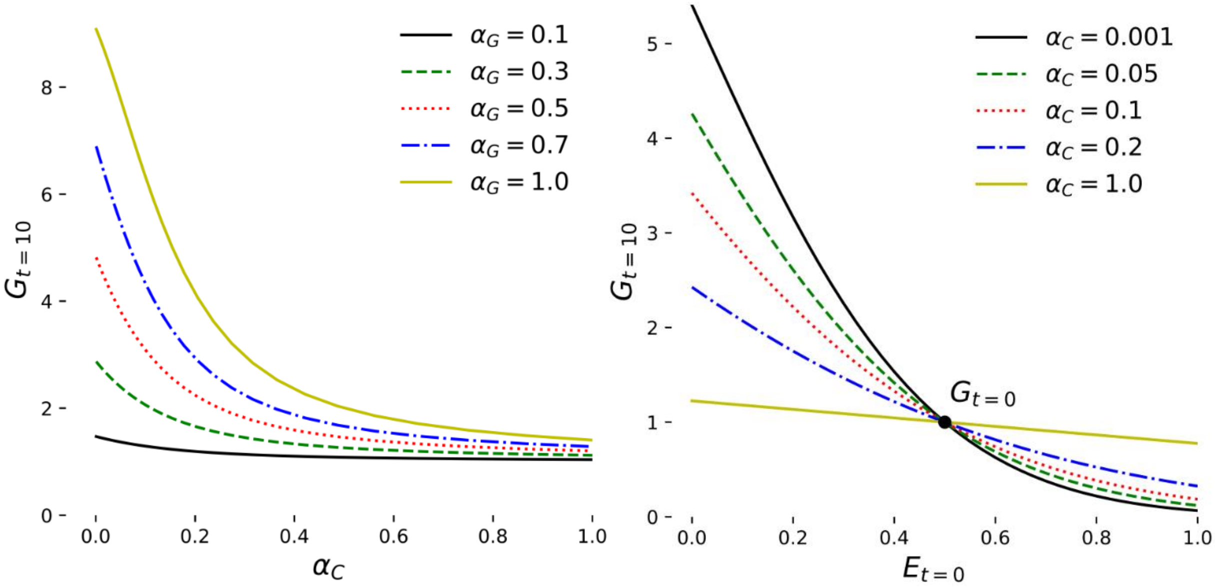


*Figure 3-4 The OpALS values for G after the tenth successive trialstep with the same reward of 0.5. $G_{t=0}$ is set as 1 and $M = 10$. **Left**: How the relationship between α_G and α_C affects the growth of G. $E_{t=0} = 0.05$. Right: How the value of $E_{t=0}$ affects the growth of G across a range of α_C, with $\alpha_G = 0.5$.*

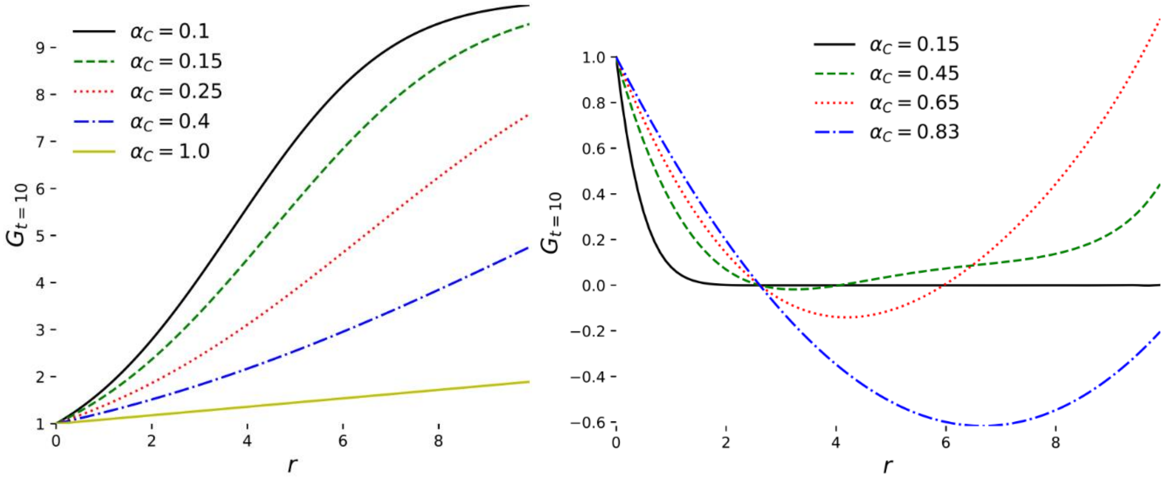


*Figure 3-5 The OpALS values for G after the tenth successive trialstep with the same reward. $G_{t=0} = 1$, $\alpha_G = 0.5$ and $M = 10$. **Left**: $E_{t=0} = 0.8r$ **Right**: $E_{t=0} = 1.85r$*

Stages at t	Opponent Actor Learning Saturated (OpALS)
Reward expectation	$V_{d,t} = \sum_{s \in S_t} s_t E_{s,d,t}$
Action choice	$A^*_{d,t} = \sum_{s \in S_t} s_t A_{s,d,t}$ $P_{d,t} = \frac{e^{\beta A^*_{d,t}}}{\sum_{i \in \mathcal{D}_t} e^{\beta A^*_{i,t}}}$ $c_t = \mathcal{C}(P_t)$
RPE	$\delta_t = r_t - V_{c_t,}$
Critic update	$E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha_C s_t}{\lVert S_t \rVert} \delta_t : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	$\begin{array}{l} G_{s,d,t+1} = G_{s,d,t} + \frac{\alpha_G s_t}{\lVert S_t \rVert} G_{s,d,t} \delta_t \left(1 - \frac{G_{s,d,t}}{M}\right) \\ N_{s,d,t+1} = N_{s,d,t} - \frac{\alpha_N s_t}{\lVert S_t \rVert} N_{s,d,t} \delta_t \left(1 - \frac{N_{s,d,t}}{M}\right) \end{array} : d = c_t$ $\begin{array}{l} G_{d,t+1} = G_{d,t} \\ N_{d,t+1} = N_{d,t} \end{array} : \mathrm{d} \neq c_t$ $A_{t+1} = (1 + \rho) G_{d,t+1} - (1 - \rho) N_{d,t+1}$

Table 3-6 The description of the OpAL model with a saturation component, broken into the components used in the implementation.

3.5 Temporal difference learning

The temporal difference model can be thought of as an extension of Q-learning that rewards actions that provide the best future rewards (Sutton, 1988; Sutton & Barto, 1998). This can be seen as an extension of the reward prediction error with a weighted extra component based on future rewards. The weightings, or discount factor, γ, are such that rewards that are further in the future are given less importance. This diminishing weighting, or discounting, changes by a factor of γ for each further trialstep. The resulting reward prediction error is:

$$\delta_t = r_t - V_{c_t,t} + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots = r_t - V_{c_t,t} + \sum_{i=1}^{i=\infty} \gamma^i r_{t+i}$$

This transforms the nature of the expected reward, $V_{d,t}$ from being a prediction of the reward at time t to being a prediction of the future discounted rewards:

$$r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots = \sum_{i=0}^{i=\infty} \gamma^i r_{t+i}$$

We can therefore consider that the reward prediction for the subsequent trialstep is an approximation of the future rewards, transforming the reward prediction error to:

$$\delta_t = r_t - V_{c_t,t} + \gamma V_{c_{t+1},\, t+1}$$

Here, $V_{c_{t+1},\, +1}$ is calculated once the action has been chosen for the following trialstep. As the following action choice depends on the actions available in the next trialstep, as well as the stimulus cue values for the new trialstep, S_t, this part of the reward prediction error is deferred to the following trialstep.

Stages at t	Temporal difference learning (TD0)
Reward expectation	$V_{d,t} = \sum_{s \in S_t} s_t E_{s,d,t}$
Action choice	$P_{d,t} = \frac{e^{\beta V_{d,t}}}{\sum_{i \in \mathcal{D}_t} e^{\beta V_{i,t}}}$ $c_t = \mathcal{C}(P_t)$ $E_{s_{t-1},d,\, t+1} = E_{s_{t-1},d,t} + \frac{\alpha\gamma s_{t-1}}{\lVert S_{t-1} \rVert} V_{c_t,t} : d = c_{t-1}$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_{t-1}$
RPE	$\delta_t = r_t - V_{c_t,t}$
Critic update	$E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha s_t}{\lVert S_t \rVert} \delta_t : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	--

Table 3-7 The description of the simplest version of the Temporal difference learning model, broken into the components used in the implementation.

An alternative way of integrating temporal discounting is discussed by Daw & Touretzky (2002). Here, they choose to separate the expected reward into two parts: an average reward and a relative reward. The average reward, $\Delta_{d,t}$, is calculated in the same way as the expectation, with a learning rate parameter τ.

$$\Delta_{d,t+1} = \Delta_{d,t} + \tau(r_t - \Delta_{d,t})$$

The RPE calculation only uses the relative value of the expected future reward. This is done by removing the average reward from the RPE:

$$\delta_t = r_t - V_{c_t,} + V_{c_{t+1},+1} - \Delta_{d,t}$$

This relative difference calculation removes the need for the discount factor γ in other temporal difference learning models, while providing quite similar results (Tsitsiklis & Van Roy, 2002).

Stages at t	Temporal relative difference learning (TDR)
Reward expectation	$V_{d,t} = \sum_{s \in S_t} s_t E_{s,d,t}$
Action choice	$P_{d,t} = \frac{e^{\beta V_{d,t}}}{\sum_{i \in \mathcal{D}_t} e^{\beta V_{i,t}}}$ $c_t = \mathcal{C}(P_t)$ $E_{s_{t-1},d,t+1} = E_{s_{t-1},d,t} + \frac{\alpha s_{t-1}}{\lVert S_{t-1} \rVert} V_{c_t,t} : d = c_{t-1}$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_{t-1}$
RPE	$\delta_t = r_t - V_{c_t,t} - \Delta_{d,t}$
Critic update	$\Delta_{d,t+1} = \Delta_{d,t} + \tau(r_t - \Delta_{d,t})$ $E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha s_t}{\lVert S_t \rVert} \delta_t : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	--

Table 3-8 The description of the Temporal difference learning with relative update, broken into the components used in the implementation.

3.6 Q-LEARNING AUTOCORRELATION

One aspect that is not examined in many reinforcement learning models is the influence that past choices have on the current choice (Lau & Glimcher, 2005). One model that attempts to address this is the Q-Learn autocorrelation model, as described by Daw (2011). Here, an extra component, κ, has been added to the action-choice probability calculation. The value of the parameter κ is zero unless the action-choice currently being calculated is the same as the one that was chosen in the previous trialstep, in which case the value can be anything in the range $[-1, 1]$, with -1 signifying a strong anti-correlation and 1 a strong correlation. By multiplying the correlation factor by β, its significance is maintained independently of the value of β.

Stages at t	Q-learning autocorrelation (qLearnCorr)
Reward expectation	$V_{d,t} = \sum_{s \in S_t} s_t E_{s,d,t}$
Action choice	$P_{d,t} = \frac{e^{\beta\left(V_{d,t} + \kappa(d = c_{t-1})\right)}}{\sum_{i \in D} e^{\beta\left(V_{i,t} + \kappa(i = c_{t-1})\right)}}$ $c_t = \mathcal{C}(P_t)$
RPE	$\delta_t = r_t - V_{c_t,}$
Critic update	$E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha s_t}{\|S_t\|}\delta_t : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	--

Table 3-9 The description of the Q-learning model with autocorrelation broken into the components used in the implementation.

3.7 Q-LEARNING-Ε

Another approach to calculating the probabilities of a given action being the best action is using the ε-greedy function. Here, the actions with the highest expected reward are identified. The probabilities for those that do not have the maximum reward being chosen is

$$\mathrm{P}\left(d \middle| V_d < \max_d V_d\right) = \frac{\varepsilon}{\|\mathcal{D}_t\|}$$

Where $\|\mathcal{D}_t\|$ is the number of valid actions at time t. The resulting odds for one of those with the maximum expected reward being chosen is

$$\mathrm{P}\left(d \middle| V_d = \max_d V_d\right) = \frac{1-\varepsilon}{\|B_t\|} + \frac{\varepsilon}{\|\mathcal{D}_t\|}$$

Where $\|B_t\|$ is the number of valid actions that have the maximum expected reward at time t.

This model was modified to include a form of autocorrelation, similar to that found in the Q-learning autocorrelation model. This is achieved using the same correlation parameter, κ, as used in the Q-learning autocorrelation model described in chapter 0. In that model, κ was a weight modifying the likelihood of a model being chosen depending on the if it was the same action that was chosen in the previous trialstep, or not. Positive values of κ would encourage a positive correlation and negative values of κ would encourage a negative correlation. The encouragement was made independent of the exploration-exploitation scaling parameter, β. Here, the same result is created using a different formulation. The impact on the likelihoods is split into two types, with correlation of the same action as before and anti-correlation of different actions from before treated the same way, and the opposites treated another way, as described in Table 3-10.

	$L_d = 1$	$L_d = -1$
$\kappa > 0$	$P_d^* = P_d + \lvert\kappa\rvert(1 - P_d)$	$P_d^* = P_d - \lvert\kappa\rvert P_d$
$\kappa < 0$	$P_d^* = P_d - \lvert\kappa\rvert P_d$	$P_d^* = P_d + \lvert\kappa\rvert(1 - P_d)$

Table 3-10 An enumeration of the different correlation modifications to the likelihoods. Here, to make explicit the modifications, the magnitude of κ is used, $|\kappa|$.

Stages at t	Q-learning-ε (qLearnE)
Reward expectation	$V_{d,t} = \sum_{s\in S_t} s_t E_{s,d,t}$
Action choice	$B_{d,t} = \begin{cases} 1 & V_{d,t} = \max_d V_{d,t} \\ 0 & V_{d,t} < \max_d V_{d,t} \end{cases}$ $P_t = \left(\frac{1-\varepsilon}{\lVert B_t \rVert}\right) B_t + \frac{\varepsilon}{\lVert \mathcal{D}_t \rVert}$ $c_t = \mathcal{C}(P_t)$
RPE	$\delta_t = r_t - V_{c_t,}$
Critic update	$E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha s_t}{\lVert S_t \rVert}\delta_t : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	--

Table 3-11 The description of the Q-learning model using epsilon greedy, broken into the components used in the implementation.

Stages at t	Q-learning-ε autocorrelation (qLearnECorr)
Reward expectation	$V_{d,t} = \sum_{s\in S_t} s_t E_{s,d,t}$
Action choice	$B_{d,t} = \begin{cases} 1 & V_{d,t} = \max_d V_{d,t} \\ 0 & V_{d,t} < \max_d V_{d,t} \end{cases}$ $P_t = \left(\frac{1-\varepsilon}{\lVert B_t \rVert}\right) B_t + \frac{\varepsilon}{\lVert \mathcal{D}_t \rVert}$ $L_d = \begin{cases} 1 & d = c_{t-1} \\ -1 & d \neq c_{t-1} \end{cases}$ $P^*_{d,t} = P_{d,t} + \begin{cases} (1 - P_{d,t})\kappa L_d & \kappa L_d > 0 \\ P_{d,t}\kappa L_d & \kappa L_d < 0 \end{cases}$ $c_t = \mathcal{C}(P^*_t)$
RPE	$\delta_t = r_t - V_{c_t,}$
Critic update	$E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha s_t}{\lVert S_t \rVert}\delta_t : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	--

Table 3-12 The description of the Q-learning model epsilon greedy and with autocorrelation, broken into the components used in the implementation.

3.8 Actor-Critic

To determine the benefit of separating the actor and the critic, a simple Q-learning model was proposed. This used the same reinforcement learning rule for both the actor and the critic, but allowed for different learning rates for each, α_C and α_A respectively. This can also be thought of as a simplification of the OpAL without Hebbian learning, as it does not have separate excitatory and inhibitory components to the actor. A basic version was created with the common Softmax function for calculating the action choice probabilities, ACBasic, along with a version using the ε-greedy function used in the Q-learning ε model, ACE.

Stages at t	Actor-critic (ACBasic)	Actor-critic-ε (ACE)
Reward expectation	$V_{d,t} = \sum_{s \in S_t} s_t E_{s,d,t}$	$V_{d,t} = \sum_{s \in S_t} s_t E_{s,d,t}$
Action choice	$A^*_{d,t} = \sum_{s \in S_t} s_t A_{s,d,t}$ $P_{d,t} = \frac{e^{\beta A^*_{d,t}}}{\sum_{i \in \mathcal{D}_t} e^{\beta A^*_{i,t}}}$ $c_t = \mathcal{C}(P_t)$	$A^*_{d,t} = \sum_{s \in S_t} s_t A_{s,d,t}$ $B_d = \begin{cases} 1 & A^*_{d,t} = \max_d A^*_{d,t} \\ 0 & A^*_{d,t} < \max_d A^*_{d,t} \end{cases}$ $P_t = \left(\frac{1-\varepsilon}{\lVert B_t \rVert} + \frac{\varepsilon}{\lVert \mathcal{D}_t \rVert}\right) B + \frac{\varepsilon}{\lVert \mathcal{D}_t \rVert}(1-B)$ $c_t = \mathcal{C}(P_t)$
RPE	$\delta_t = r_t - V_{c_t,t}$	$\delta_t = r_t - V_{c_t,t}$
Critic update	$E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha_C s_t}{\lVert S_t \rVert}\delta_t : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$	$E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha_C s_t}{\lVert S_t \rVert}\delta_t : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	$A_{s,d,t+1} = A_{s,d,t} + \frac{\alpha_A s_t}{\lVert S_t \rVert}\delta_t : d = c_t$ $A_{d,t+1} = A_{d,t} : \mathrm{d} \neq c_t$	$A_{s,d,t+1} = A_{s,d,t} + \frac{\alpha_A s_t}{\lVert S_t \rVert}\delta_t : d = c_t$ $A_{d,t+1} = A_{d,t} : \mathrm{d} \neq c_t$

Table 3-13 The description of the Actor-critic models using softmax and epsilon greedy, broken into the components used in the implementation.

A cruder version of the ACE model was created with a less discerning critic. Here, the critic assesses whether the reward is higher than the average across actions, irrespective of the stimuli. Effectively, the critic is comparing the reward with a moving average of the reward. The actor is therefore learning not how the predictions compare to its expectations of reward for that action, but, indirectly, how the actions' reward compares to those of all possible actions.

Stages at t	Actor-critic-ε-simplified (ACES)
Reward expectation	$V_t = E_t$
Action choice	$A^*_{d,t} = \sum_{s \in S_t} s_t A_{s,d,t}$ $Bd = \begin{cases} 1 & A^*_{d,t} = \max_d A^*_{d,t} \\ 0 & A^*_{d,t} < \max_d A^*_{d,t} \end{cases}$ $P_t = \left(\frac{1-\varepsilon}{\lVert B_t \rVert}\right) B + \frac{\varepsilon}{\lVert \mathcal{D}_t \rVert}$ $c_t = \mathcal{C}(P_t)$
RPE	$\delta_t = r_t - V_t$
Critic update	$E_{t+1} = E_t + \alpha\delta_t$
Actor update	$A_{s,d,t+1} = A_{s,d,t} + \frac{s_t}{\lVert S_t \rVert}\delta_t : d = c_t$ $A_{d,t+1} = A_{d,t} : \mathrm{d} \neq c_t$

Table 3-14 The description of the Actor-critic model with epsilon greedy simplified, broken into the components used in the implementation.

3.9 Meta Q-learning

Schweighofer & Doya (2003) proposed accounting for the uncertainty in the rewards by having an adaptive exploration-exploitation parameter, β, calculated based on the rate of change of average reward. An adapted version of the model proposed by is examined here. The model calculates a moving average for the reward with a learning rate parameter of τ:

$$\Delta_{d,t+1} = \Delta_{d,t} + \tau\left(r_t - \Delta_{d,t}\right)$$

A moving average is also calculated for the moving average, with the same learning rate parameter:

$$\Delta^*_{d,t+1} = \Delta^*_{d,t} + \tau\left(\Delta_{d,t} - \Delta^*_{d,t}\right)$$

The estimate of the appropriate β is based on the difference between these two moving averages, Δ and Δ*.

$$\beta_{t+1} = e^{\left(\Delta_{t+1} - \Delta^*_{t+1}\right)}$$

Stages at t	Meta Q-learning (qLearnMeta)
Reward expectation	$V_{d,t} = \sum_{s \in S_t} s_t E_{s,d,t}$
Action choice	$P_{d,t} = \frac{e^{\beta_t V_{d,t}}}{\sum_{i \in \mathcal{D}_t} e^{\beta_t V_{i,t}}}$ $c_t = C(P_t)$
RPE	$\delta_t = r_t - V_{c_t,}$
Critic update	$\Delta_{d,t+1} = \Delta_{d,t} + \tau\left(r_t - \Delta_{d,t}\right)$ $\Delta^*_{d,t+1} = \Delta^*_{d,t} + \tau\left(\Delta_{d,t} - \Delta^*_{d,t}\right)$ $\beta_{t+1} = e^{\left(\Delta_{t+1} - \Delta^*_{t+1}\right)}$ $E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha s_t}{\lVert S_t \rVert} \delta_t : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	--

Table 3-15 The description of the Meta Q-learning model, broken into the components used in the implementation.

3.10 Kalman filter

The Kalman filter, as presented by Daw, O'Doherty, Dayan, Dolan, & Seymour (2006), attempts to estimate the uncertainty in the expected reward[2]. It uses this prediction uncertainty measure, $\sigma^2_{d,t}$, to define a learning rate, $\alpha_{d,t}$

$$\alpha_{d,t} = \frac{\sigma^2_{d,t}}{\sigma^2_{d,t} + \sigma^2_{\alpha}}$$

Where σ^2_{α} is the measurement uncertainty, which is considered a constant over the duration of a task and identical for all actions. This learning rate is used in a similar way to that of the Q-learning models to calculate an updated expectation, $\hat{E}_{s,d,t}$:

$$\hat{E}_{s,d,t} = E_{s,d,t} + \frac{\alpha_{d,t} s_t}{\|S_t\|} \delta_t$$

Now it is also necessary to update the uncertainty. For the chosen action, the updated uncertainty, $\hat{\sigma}^2_{d,t}$, decreases at a rate proportional to the learning rate:

$$\hat{\sigma}^2_{d,t} = (1 - \alpha_{d,t})\sigma^2_{d,t}$$

Having now incorporated the new knowledge from the events at time t, the model now tries to include information about the unknown and unmeasurable factors affecting the task events. In practice for this model, this takes the form of a drift towards a baseline value for the expected reward and a growth in the prediction uncertainty. The drift rate, λ, sets the drift towards the baseline expected reward, E_{λ}, as well as the uncertainty growth rate from a base uncertainty of σ^2_{λ}.

$$E_{s,d,t+1} = \lambda \hat{E}_{s,d,t} + (1 - \lambda) E_{\lambda}$$

$$\sigma^2_{d,t+1} = \lambda^2 \hat{\sigma}^2_{d,t} + \sigma^2_{\lambda}$$

As the baseline expectation is the one that will be used for the initial expectation, $E_{s,d,t=0}$, we can update the equation to the form:

$$E_{s,d,t+1} = \lambda \hat{E}_{s,d,t} + (1 - \lambda) E_{s,d,t=0}$$

[2] For a clear description of the derivation of the Kalman filter, a good starting point is (Faragher, 2012)

Stages at t	Q-learning Kalman (qLearnK)
Reward expectation	$V_{d,t} = \sum_{s \in S_t} s_t E_{s,d,t}$
Action choice	$P_{d,t} = \frac{e^{\beta V_{d,t}}}{\sum_{i \in \mathcal{D}_t} e^{\beta V_{i,t}}}$ $c_t = \mathcal{C}(P_t)$
RPE	$\delta_t = r_t - V_{c_t,}$
Critic update	$\alpha_{d,t} = \frac{\sigma^2_{d,t}}{\sigma^2_{d,t} + \sigma^2_{\alpha}}$ $\hat{E}_{s,d,t} = E_{s,d,t} + \frac{\alpha_{d,t} s_t}{\lVert S_t \rVert} \delta_t$ $: d = c_t$ $\hat{\sigma}^2_{d,t} = (1 - \alpha_{d,t}) \sigma^2_{d,t}$ $\hat{E}_{s,d,t} = E_{d,t}$ $: d \neq c_t$ $\hat{\sigma}^2_{d,t} = \sigma^2_{d,t}$ $E_{s,d,t+1} = \lambda \hat{E}_{s,d,t} + (1 - \lambda) E_{s,d,t=0}$ $\sigma^2_{d,t+1} = \lambda^2 \hat{\sigma}^2_{d,t} + \sigma^2_{\lambda}$
Actor update	-

Table 3-16 The description of the Q-learning Kalman model, broken into the components used in the implementation.

3.11 Bayesian

The Kalman and meta Q-learning models described above attempt to represent the uncertainty of the expected reward by estimating a form of variance. In the cases where the reward can be thought of as feedback on which was the correct action, as described in chapter 2.1.2, this uncertainty can be thought of as the level of uncertainty in the model's prediction. While the addition of an uncertainty estimate is an improvement, it is limited by the assumption that the underlying likelihood distribution is gaussian. One of the simplest ways of extending this to a more varied range of likelihood distributions is using the Beta distribution.

The beta distribution can be used to express a family of different likelihood distributions using two parameters, often thought of as number of successes and number of failures, as shown in Figure 3-6.

The result is that the distribution can be updated for each trial by updating the number of successes and failures.

As tasks may have a more diverse range of consequences than success and failures, a Dirichlet distribution is used as a way of representing the distribution of

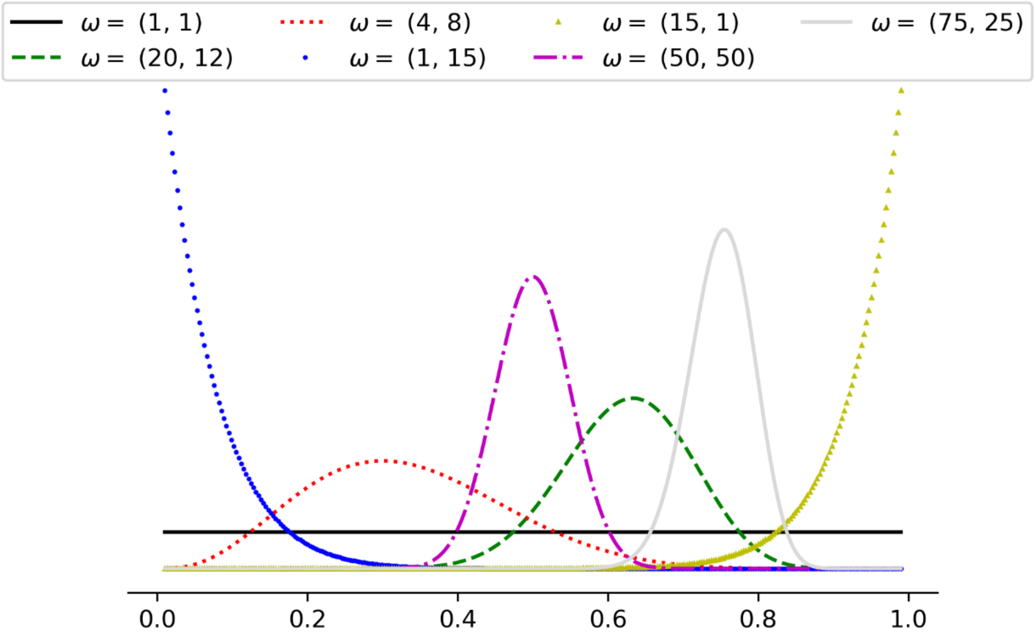


Figure 3-6 A series of example distributions that can be produced using the Beta distribution. The legend shows the values of the two Beta distribution parameters necessary to produce the given shape of uncertainty distribution.

probabilities for each possible outcome. The Dirichlet is a form of the Beta distribution generalised for a larger number of categorical rewards than the two found with the Beta distribution. More formally, for a reward r, in the set of known possible rewards $\mathcal{R}$, the associated count parameter, $\omega_{r,t}$, will be updated by one for each occurrence:

$$\omega_{r,t+1} = \omega_{r,t} + 1$$

For this model, r will be restricted to a set of positive integer values. The likelihood of a reward of r for the Dirichlet distribution $\mathfrak{D}(\omega)$ can be calculated as:

$$\mathbb{E}_r[\mathfrak{D}(\omega)] = \frac{\omega_r}{\sum_{i\in\mathcal{R}}\omega_i} = \frac{\omega_r}{\omega_0}$$

With ω_0 defined as:

$$\omega_0 = \sum_{i\in\mathcal{R}}\omega_i$$

From this, a Dirichlet distribution is constructed for each possible action. For action d, the expected reward, $V_{d,t}$, is the weighted sum of these likelihoods:

$$V_{d,t} = \mathbb{E}[\mathfrak{D}(\omega_t)] = \sum_{r\in\mathcal{R}}\frac{\omega_{r,d,t}}{\omega_{0,t}}r$$

This can be extended for use with multiple stimulus-cues. To calculate the appropriate expected reward values, the count parameters can be stored separately for each stimulus cue, s, and then combined across rewards, weighted by the activation of each cue, s_t:

$$V_{d,t} = \mathbb{E}\left[\mathfrak{D}\left(\sum\nolimits_{s\in S_t} s_t\omega_{r,s,d,t}\right)\right] = \sum_{r\in\mathcal{R}}\frac{\sum_{s\in S_t} s_t\omega_{r,s,d,t}}{\sum_{i\in\mathcal{R}}\sum_{s\in S_t} s_t\omega_{i,s,d,t}}r$$

To allow the learning rate to change between participants, the updating of the count parameters can be modified to use the same form of learning rate parameter as used in the Q-learning model, by modifying the increment of 1 to be an increment of α:

$$\omega_{r,s,d,t+1} = \omega_{r,s,d,t} + \alpha$$

However, the effect on the expected reward is less direct than with the Q-learning model, as the impact of an incremental update on the expected reward will vary depending on the size of $\omega_{0,t}$ and the distribution of each $\omega_{r,t}$. Figure 3-7 gives an example of how a distribution $\mathfrak{D}(\omega_t = [\omega_{1,t}, \omega_{2,t}] = [2,5])$ can be updated to $\mathfrak{D}(\omega_{t+1})$ with different increment sizes for $r = 2$.

As for the Q-learning model, this update function can be modified for use with multiple stimulus cues:

$$\omega_{r,s,d,t+1} = \omega_{r,s,d,t} + \frac{\alpha s_t}{\|S_t\|}$$

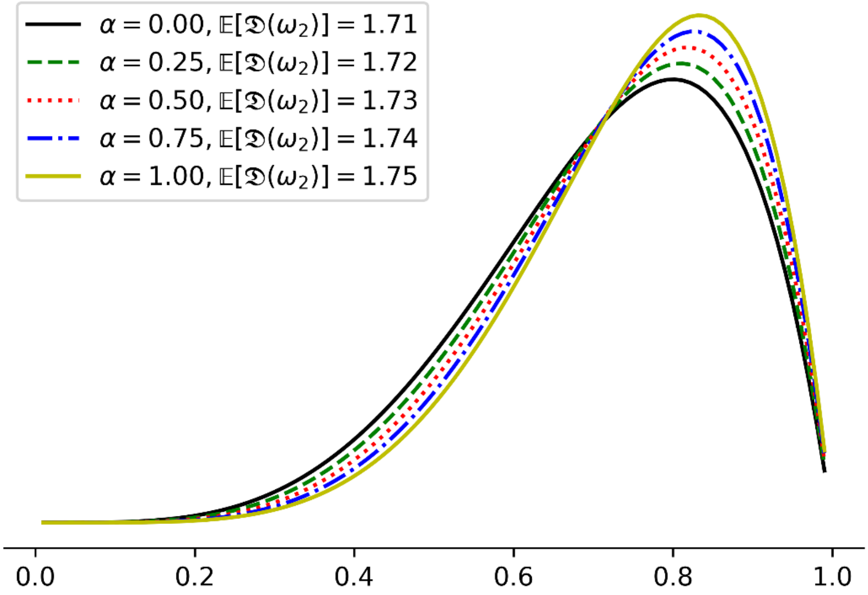


Figure 3-7 Examples of how different α increments can affect the change in expected reward in a Dirichlet distribution. All the distributions began as $\omega = [\omega_1, \omega_2] = [2,5]$ and the α updated for reward 1, whose probability distributions are shown.

Stages at t	Bayesian Probabilistic (BP)
Reward expectation	$V_{d,t} = \mathbb{E}\left[\mathfrak{D}\left(\sum_{s\in S_t} s_t \omega_{r,s,d,t}\right)\right]$
Action choice	$P_{d,t} = \frac{e^{\beta V_{d,t}}}{\sum_{i\in\mathcal{D}_t} e^{\beta V_{i,t}}}$ $c_t = \mathcal{C}(P_t)$
RPE	-
Critic update	$\omega_{r,s,d,t+1} = \omega_{r,s,d,t} + \frac{\alpha s_t}{\|S_t\|} : \begin{cases} d = c_t \\ r = r_t \end{cases}$
Actor update	-

Table 3-17 The description of the Bayesian probabilistic model, broken into the components used in the implementation.

The variance of the distributions can be calculated in a similar way to the expectation:

$$\mathrm{Var}_r[\mathfrak{D}(\omega)] = \frac{\omega_r(\omega_0 - \omega_r)}{\omega_0^2(\omega_0 + 1)}$$

This uncertainty can be used as an estimate of the need for exploration: it is more valuable to explore when there is high uncertainty than when the uncertainty is low. An estimate of the overall uncertainty can be calculated by summing all the variances:

$$\mathrm{Var}[\mathfrak{D}(\omega)] = \sum_{r \in \mathcal{R}} \frac{\omega_r(\omega_0 - \omega_r)}{\omega_0^2(\omega_0 + 1)}$$

This can be extended as before for use with multiple stimulus-cues:

$$\sigma_{d,t}^2 = \mathrm{Var}\left[\mathfrak{D}\left(\sum\nolimits_{s \in S_t} s_t \omega_{r,s,d,t}\right)\right]$$

We can draw parallels to the use of β in the Q-learning model in chapter 3.2 by inverting $\sigma_{d,t}^2$:

$$\beta_{d,t} = \frac{1}{\sigma_{d,t}^2}$$

To turn this into a form approximating that of β, a correction is necessary. The first step is to recognise that the largest uncertainty will be when the model has no information, at $t = 0$. β_d can therefore be normalised as:

$$\beta_{d,t} = \frac{\sigma_{d,*}^2}{\sigma_{d,t}^2}$$

where $\sigma_{d,*}^2$ is a normalising term that can be defined as the uncertainty from the initial prior values of reward occurrences, $\omega_{r,s,d,t=0}$, weighted by the current stimulus-cue weightings:

$$\sigma_{d,*}^2 = \mathrm{Var}\left[\mathfrak{D}\left(\sum\nolimits_{s \in S_t} s_t \omega_{r,s,d,t=0}\right)\right]$$

At $t = 0$ this would leave a $\beta_{d,0} = 1$. This is not quite what is needed for the tasks considered here, as there should be no initial preference for one action or another, which would be reflected by $\beta_{d,0} = 0$, i.e. equal likelihood for all actions. This can be achieved by modifying the calculation of β_d to be:

$$\beta_{d,t} = \frac{\sigma^2_{d,*}}{\sigma^2_{d,t} - 1}$$

Stages at t	Bayesian probabilistic volatility (BPV)
Reward expectation	$V_{d,t} = \mathbb{E}\left[\mathfrak{D}\left(\sum_{s \in S_t} s_t \omega_{r,s,d,t}\right)\right]$ $\sigma^2_{d,t} = \mathrm{Var}\left[\mathfrak{D}\left(\sum_{s \in S_t} s_t \omega_{r,s,d,t}\right)\right]$
Action choice	$\sigma^2_{d,*} = \mathrm{Var}\left[\mathfrak{D}\left(\sum_{s \in S_t} s_t \omega_{r,s,d,t=0}\right)\right]$ $\beta_d = \frac{\sigma^2_{d,*}}{\sigma^2_{d,t}} - 1$ $P_{d,t} = \frac{e^{\beta_d V_{d,t}}}{\sum_{i \in \mathcal{D}_t} e^{\beta_i V_{i,t}}}$ $c_t = \mathcal{C}(P_t)$
RPE	-
Critic update	$\omega_{r,s,d,t+1} = \omega_{r,s,d,t} + \frac{\alpha s_t}{\lVert S_t \rVert} : \begin{cases} d = c_t \\ r = r_t \end{cases}$
Actor update	-

Table 3-18 The description of the Bayesian probabilistic volatility model, broken into the components used in the implementation.

3.12 RANDOM

To evaluate the performance of the models and calculate some of the fit quality measures described in chapter 2.2 a pure random model will be used for comparison (M. D. Lee et al., 2019). For each trial, the action choice probability, $P_{d,t}$ will be the same for all available actions:

$$P_{d,t} = \frac{1}{\|\mathcal{D}_t\|}$$

This can be extended by allowing a constant bias in the action choice probabilities. A series of $\mathcal{D}$ parameters, denoted o_d, would each have values in the range [0, 1] and sum to 1. This mean there are $\mathcal{D} - 1$ free parameters, which can be represented on a $\mathcal{D} - 1$ unit simplex.

Stages at t	Random (random)	Random biased (randomBias)
Reward expectation	--	--
Action choice	$P_{d,t} = \frac{1}{\|\mathcal{D}_t\|}$ $c_t = \mathcal{C}(P_t)$	$\sum_{d\in\mathcal{D}} o_d = 1$ $P_{d,t} = \frac{o_d}{\sum_{i\in\mathcal{D}_t} o_i}$ $c_t = \mathcal{C}(P_t)$
RPE	--	--
Critic update	--	--
Actor update	--	--

Table 3-19 The descriptions of the Random and Random biased models, broken into the components used in the implementation.

4 Discerning model parameters in reinforcement learning

Before evaluating the performance of models across tasks, it is worth understanding how noisy or uncertain the model fitting process is. Ideally, the models would be globally identifiable, i.e. each action sequence could only have been generated by one parameter combination (Moran, 2016). However, in a model where many parameter combinations will provide a non-zero probability for each available action to be chosen, there will be a finite probability for each of those parameter combinations to produce every sequence of actions. The evaluation of the capacity of a model to be fitted can be evaluated, for a given task, by asking: *if I have data that I know was generated with specific parameter values, how likely am I to recover those parameter values when fitting that model to the data?* (Heathcote, Brown, & Wagenmakers, 2015). This would be equivalent to baking a cake in one of a very large number of available cake moulds and then asking how likely is it that somebody else would be able to identify the cake mould used from those available.

Reverdy & Leonard (2015) discuss formally the conditions under which a reinforcement learning model will be able to fit data from a given task. They demonstrate that with a certain kind of task and a sufficient task length, a reinforcement learning model is guaranteed to converge to the correct parameter values. While this is useful to know, it does not tell us if this convergence will be fast enough to be useful for real participant data, where the length of the task is limited by the capacity of participants to stay focused.

The capacity of a model to be fitted can be tested by generating task data using a model with known parameter values. This generated data is then fitted to the same model. From this, the resulting recovered parameter values can be compared, along with the confidence of that fit, to the parameter values used to generate the data. Fitting the same data multiple times gives us an understanding of the capacity of a given fitting process to find the best parameter values. Multiple datasets are generated for the same parameter values to gain an understanding of how the variability of the data affects the quality of the fit. This variability will compound the error found in the fitting process. By repeating this process over a range of parameter value combinations, an understanding can be gained of how the capacity to accurately recover the generating parameters varies for different model responses to the task.

When a model is fitted with real participant data the question being asked is: *given this model, what are the model parameter values that most closely resemble the performance of the participant for this model?* The fitting process is not looking for the 'correct' model parameter values (Box, 1979), as the model will be, at best, an approximation of how the participant approached the task. Any error found in fitting the generated participant data will therefore be an underestimate of the error received when fitting a model with real participant data.

Parameter recoverability for the Expectancy Valence (EV) model (Busemeyer & Stout, 2002) has been examined with the Iowa gambling task, a task similar to the Decks task examined in this thesis. Ahn et al. (2014) found satisfactory parameter recovery for an EV variant. Similarly, Ruud Wetzels, Vandekerckhove, Tuerlinckx, & Wagenmakers (2010) found that on average they were able to recover the parameters generated from 1000 sequences using the most commonly recovered parameter set from previous studies. However, the parameter recovery distribution had a large enough variance that recovery from individual task runs were not considered trustworthy. Lastly, Humphries, Bruno, Karpievitch, & Wotherspoon (2015) found poor parameter recovery when fitting both simulated and participant data, concluding both that individual parameter recovery was poor and there was sufficient within-group variance that group estimates were questionable.

Another approach to assessing parameter recoverability is to examine the performance of parameter values whose task performance was close to those of real participants. This was used to validate the three parameter Q-learning model (Halpern & Gureckis, 2013) used in Gureckis & Love (2009). While they found that those parameter values were accurately reproduced, all were on or very close to parameter values that effectively removed learning from the model, calling into question the effectiveness and validation of their fitting.

Since this study was performed, a new paper has come out examining the identifiability and recoverability of a few Q-learning models, including qLearn and qLearn2 (Spektor & Kellen, 2018). They found very poor recoverability and attempted to improve this using an *empirical-prior* parameter distributions (Gershman, 2016), with only marginal improvement. The results of this paper will be discussed further later in this chapter.

In this chapter, an assessment will be made of how well parameters can be recovered in the best case: recovered by fitting the same model used to generate the data. This is evaluated at first using the simple qLearn model, across a range of different tasks. The source of fitting errors is investigated, and potential solutions are discussed and evaluated.

4.1 Model-task data generation

For the initial exploration of the fitting error, the qLearn model described in chapter 3.2 was used, as it is one of the simplest and most widespread models, while still containing all the components found in more elaborate reinforcement learning models. To summarise, it is a two-parameter reinforcement learning model, updating only the parts associated with the chosen action and active stimulus-cues, with α as the learning rate. The action choice is performed based on probabilities calculated based on a softmax transformation of the reward expectations, with an exploration-exploitation parameter β.

For each of the task variations examined in the rest of the study, a set of datasets were generated with the parameter value combinations from $\alpha = \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and $\beta = \{0.1, 0.3, 0.5, 0.7, 1, 2, 4, 8, 16\}$. For each combination, 30 experiment runs were generated. For the fitting, the model parameters were constrained to the ranges $\alpha = [0,1]$ and $\beta = [0,30]$. The bounds for α are the valid range for the parameter. The lower bound for β is fixed by the parameter only being positively defined. The upper bound for β is more arbitrary. The true upper bound for β is positive infinity, but this is not a practical space to search.

By considering the impact of β in the Softmax, as seen in Figure 4-1, it can be seen that the impact of increasing β diminishes as β increases, such that the benefit in increasing the β support decreases as the support size increases. 16 was chosen as the largest generating β value, as our initial studies had seen changes in the range $\beta = [0, 5]$ and it was expected that 16 would be sufficiently above 5 to show if there were any trends at higher β. The parameter fitting upper bound of 30 was chosen to be far enough from the maximum β used to generate data to allow any

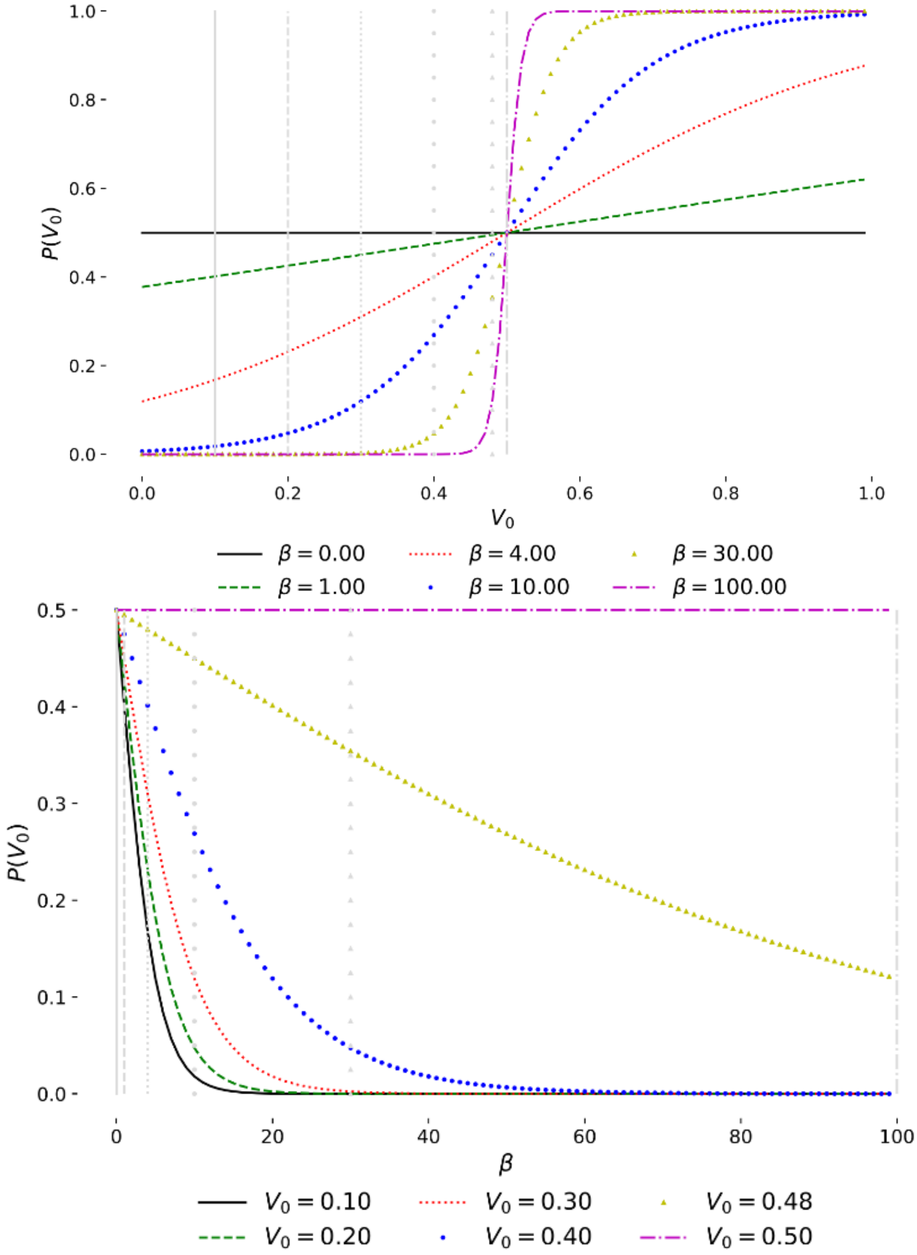


Figure 4-1 Two slices of how the probability for action 0 varies for different action-expectations, V_0 , and different β. The only other action has a constant action-expectation, $V_1 = 0.5$. The grey lines on each plot show how the lines from the other plot interact with it, with the lines using the line marker type found in the key of the other plot. ***Top****: $P(V_0)$ across V_0 for different β.* ***Bottom****: $P(V_0)$ across β for different V_0.*

distribution of fitting errors to be relatively unaffected by the boundary. This is slightly lower, but comparable to Spektor and Kellen's 2018 study choice of 50.

As described in chapter 2.1.1, the model data was recorded to files and then fitted in the same way as participant data.

4.2 Evaluation of fitting capacity for a simple task: Biased coins task

For most of this investigation a simple task called the Biased Coins task was used. This was designed to be a simplified version of the tasks that we later fit data for, in chapters 6, 7 and 8. Four distinct coins are shown to the participant. At the beginning of each trial, one of these is randomly chosen to be flipped. Before it is flipped, the chosen coin is shown to the participant and the participant guesses if it will land on one side or the other: ‘heads’ or ‘tails’. The participant is then told which side the coin landed on. This task can be thought of as a 4-armed bandit, where the participant does not choose the bandit pulled each trail, only predicts the outcome. The simulated participants are rewarded if they predicted the outcome correctly. This approach ensures that the simulated participants learn equally about each of the four coins. Without this, a rational participant would choose more frequently the coins for which they expected higher rewards.

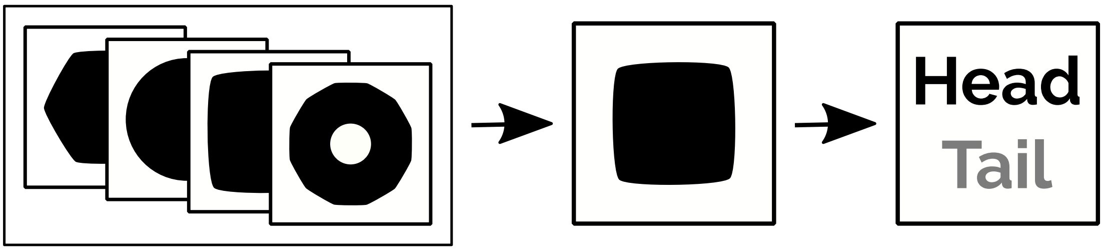


Figure 4-2 The Biased coins task. At the beginning of each trial, from a set of distinct coins, one is randomly chosen to be shown to the participant. Before it is flipped the participant guesses if it will land on side 0 or side 1. The participant is then told, during the feedback trials, which side the coin landed on. Each coin has a different probability of landing on each side.

Each trial is performed 100 times, with the final 20 being performed without feedback. The final trials are performed in this way to match final trials of the Weather and the Probabilistic Selection tasks, where these trials without feedback are used to understand more about what the participants have learnt by the end of the training phase, i.e. the final trial with feedback. Frank et al. (2007) also chose to fit the participant data exclusively on the trials without feedback, arguing that fitting only this part could lead to recovery of more accurate model parameters. This is discussed further in chapter 5.

The probability of landing on one side or another varies from coin to coin, with two of the coins having an 80% chance of landing on side 0 and the other two a 20% chance of landing on side 0. For each generated dataset, the choice of coins and the side the 'coins' fall on is varied. This allows us to get a better estimate of the distribution of the noise in fitting, with not just the noise of the probabilistic decision making of the model but also the impact of the task sequence.

4.2.1 Fits from generated data

To begin looking at the error in the recovered model parameter values, for each generating parameter combination the fit quality values were plotted for each parameter combination explored. Figure 4-3 shows the fits from data generated with two different sets of model parameter values. For $\alpha = 0.7$, $\beta = 4$ the best fit

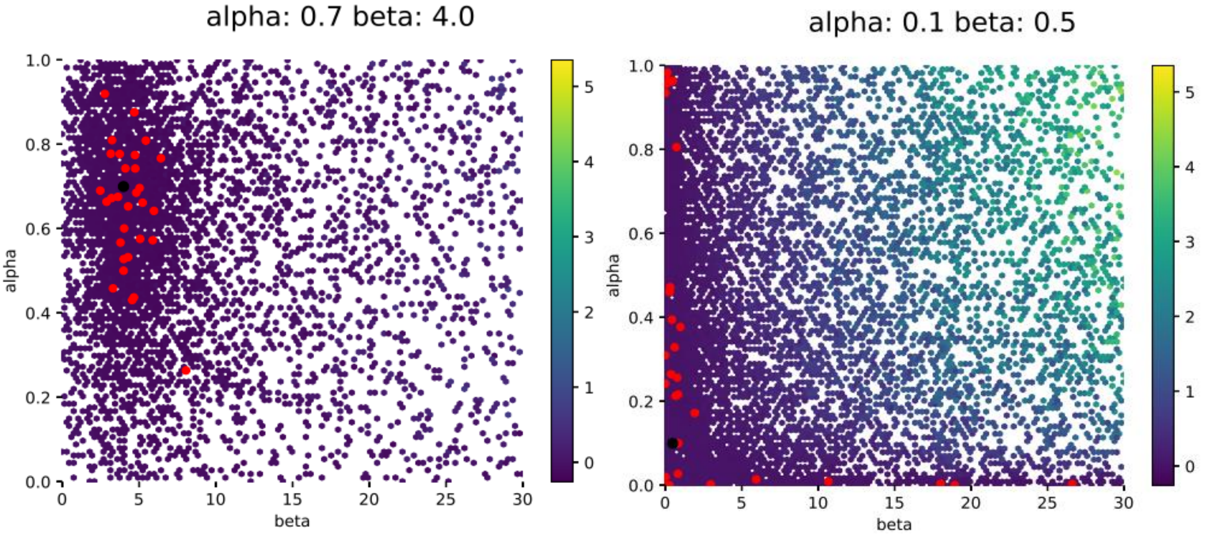


Figure 4-3 Parameter space plots showing the fitting process and results from the biased coins task with 100 trials, of which 20 are without feedback. The titles list the parameters, also marked by a black dot, from which the 30 datasets were generated. The red dots are the parameters of the best fit for each of the 30 generated datasets. The other dots show the search locations during the 30 evolutionary fitting processes, coloured using the base 10 log of the fit quality. The shape of the low fit value area in the parameter space can be seen to vary significantly with different generating parameter values.

parameter values (red dots) are clustered around the parameter values used to generate the data for the 30 datasets. However, they are spread across more than half the possible range for α. Looking at $\alpha = 0.1$, $\beta = 0.5$ the best fit parameter values are spread around the edge of the examined parameter space, close to the α and β axes. The dots of other colours signify the other parameter combinations tested during the evolutionary fitting process, with the colour signifying the log base 10 of the fit quality value described in chapter 2.2. This allows us to see that while the distributions of best fit parameter values are quite different between the two graphs, one common feature is that both sets of best fit parameter values are found in 'valleys' of their respective fit quality values.

Looking more generally, in Figure 4-5, it can be seen that the distributions of recovered parameter values varies depending on those used to generate the data. The distribution transforms as β increases from being one that follows the edges of the fitting parameter-space to one more grouped around a central point and finally stretching out in the direction of higher β. The changes in the means of these best fit parameter value distributions for each generating parameter combination, shown in Figure 4-6, also indicate that the fit for α becomes more accurate as β increases. Unsurprisingly, it also underscores how the changes in distribution are not well described by the mean and standard deviation.

These plots are similar to those produced by Daw (2011), when compared with one of the fits from data generated with the parameters $\alpha = 0.3$, $\beta = 1$, reproduced in Figure 4-4. The same shape of best fit region can be seen in the two plots, suggesting that Daw's result is part of a larger pattern of results.

The conclusion from these plots is that the fitting is only potentially reliable for certain generating parameter values. However, as the spread of best fits from those generating parameter values overlaps significantly with those around them, this does not allow us to treat any recovered parameter values in that region as reliable.

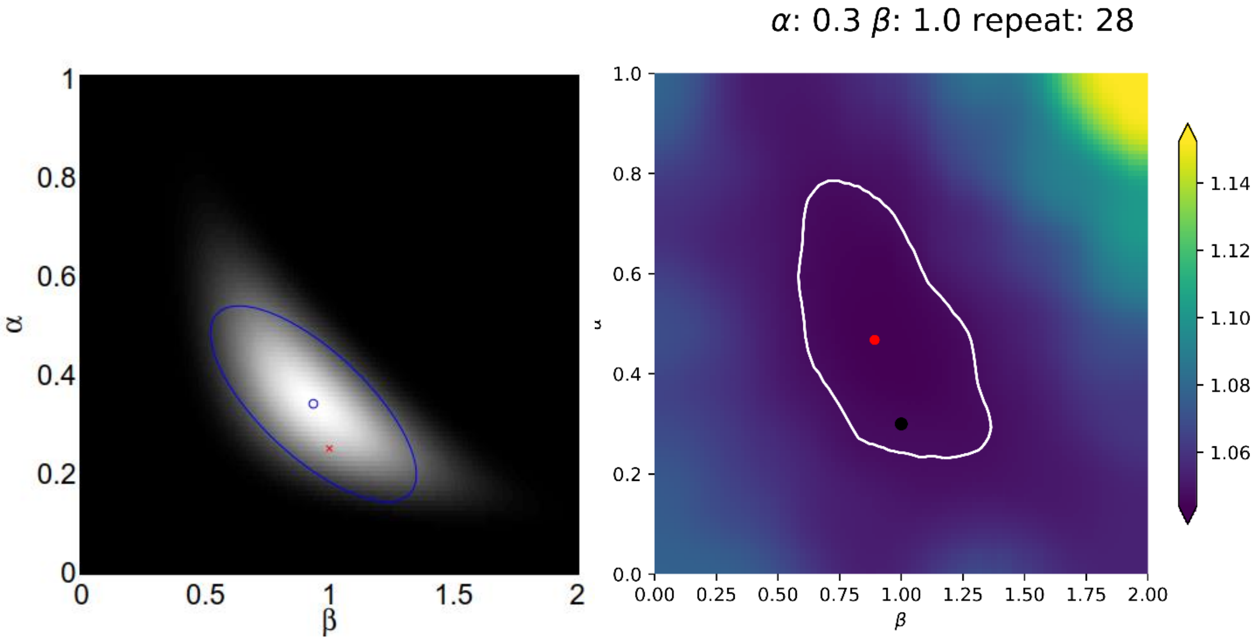


Figure 4-4 ***Left****: A reproduction of figure 1 from Daw (2011) generated by fitting the Q-learning model. From the original figure description "Lighter colors denote higher data likelihood. The maximum likelihood estimate is shown as an "o" surrounded by an ellipse of one standard error (a region of about 90% confidence); the true parameters from which the data were generated are denoted by an 'x'"* ***Right****: The results from fitting one of 30 generated datasets. The dataset was generated with* $\alpha = 0.3$*,* $\beta = 1$*, marked by the black dot. The red dot is the recovered parameters. The background is coloured using the smoothed base 10 log of the fit quality measure. A white contour is shown of equal fit quality.*

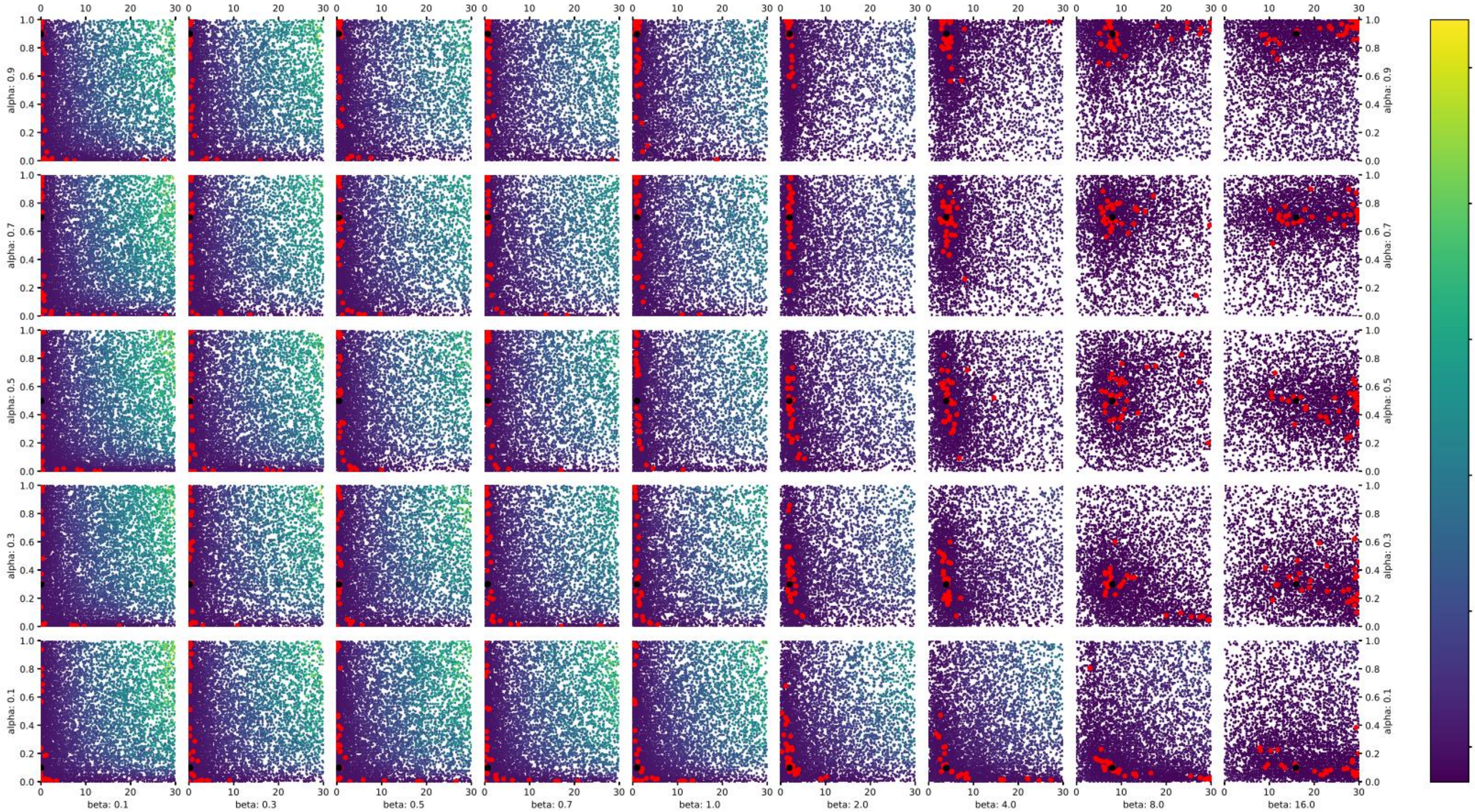


Figure 4-5 A set of parameter space plots showing the fitting process and results from the biased coins task with 100 trials, of which 20 are without feedback. The grid of plots is arranged according to the parameter values of the generated data, also marked by a black dot in each subfigure. The red dots are the parameters of the best fit for each of the 30 generated datasets for each parameter combination. The other dots show the search locations during the 30 evolutionary fitting processes, coloured using the base 10 log of the fit quality.

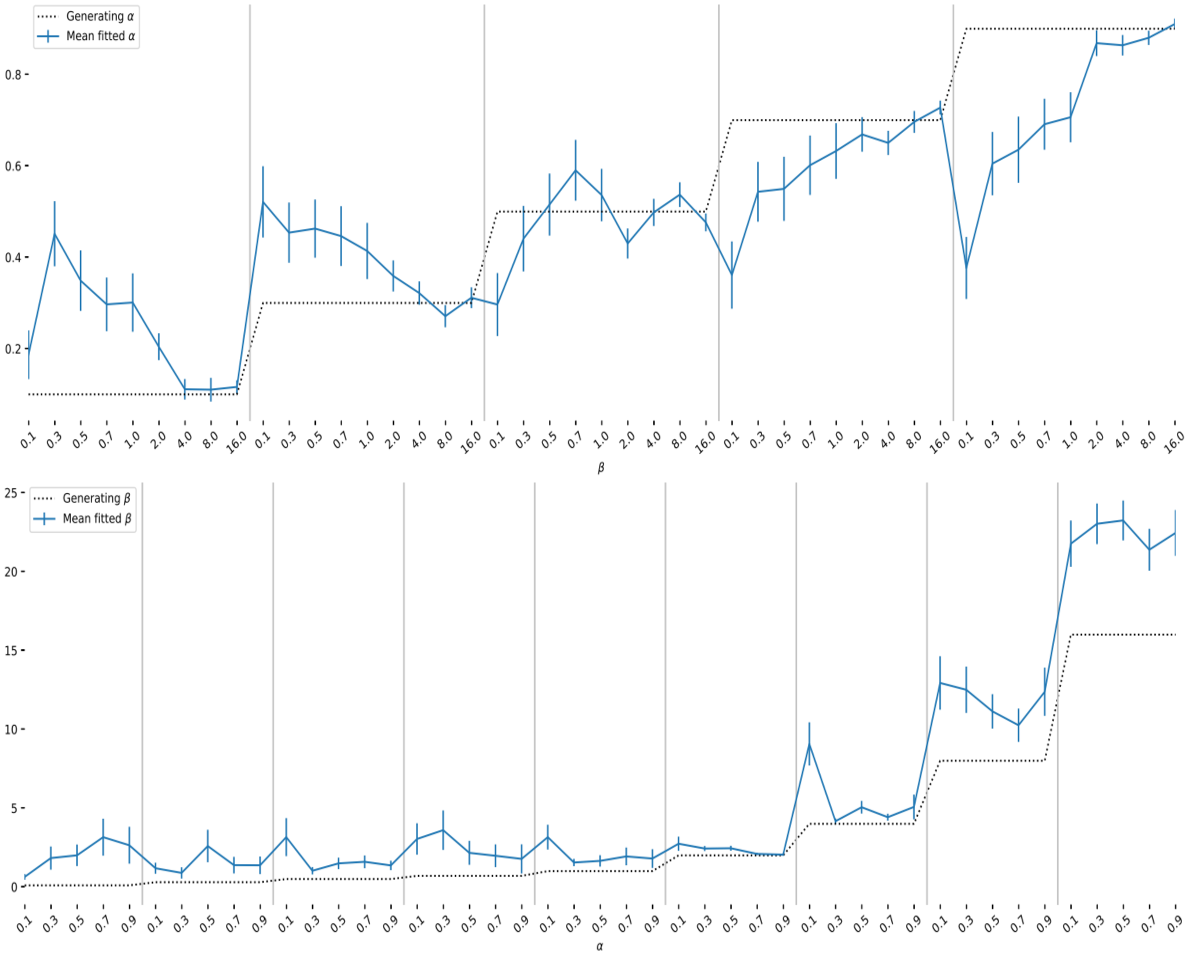


Figure 4-6 The means and standard deviations of the best fit values for α and β for the data generated from the biased coins task with 100 trials, of which 20 are without feedback. The means are calculated for each pair of parameter values used to generate the data. The graphs are plotted with the parameter values increasing in order such that the parameter increasing the fastest is the one shown on the horizontal axis, and with the dotted line denoting the generating values of those parameters. ***Top:*** *Means of α* ***Bottom:*** *Means of β*

The same generated data can be fitted more than once and the recovered parameters can be compared, as shown in Figure 4-8. Variations can be seen between the two sets of recovered parameters. These are due to the evolutionary fitting process probabilistically choosing the locations of each rounds fit quality samples, as described in chapter 2.1.6.2, resulting in different parameter combinations being considered to be the recovered parameters each time the fitting is run. In the final round of fitting the selected sample of parameter sets will be in the vicinity of the globally identified best fit and the parameter sets will have sufficiently small variations in their associated fit quality values that they could all be candidates.

While there are variations across fits in the best fit parameter values, the distributions are the same, suggesting that the area of best fit for the generating parameter combination is being consistently identified. By comparing the difference between the two sets of fits, in Figure 4-7, for α the majority of the generated data sets, the differences in the recovered parameters are less than 0.1, except for a few that jumped from one edge to the other. If this were real behavioural data, these edge fits would be considered bad fits and excluded, as they are in effect fitting to a model without a learning rate parameter. For β the size of the differences depends largely on the size of the parameter value used to generate the data. A good rule of thumb is that the differences are generally at least an order of magnitude smaller than the parameter value used to generate the data. This suggests that a participant's action sequence is associated with specific recovered parameters. As the same action sequence could have been generated by numerous parameter sets, this is unsurprising and results in the model not being locally-identifiable (Schmittmann, Dolan, Raijmakers, & Batchelder, 2010; Spektor & Kellen, 2018).

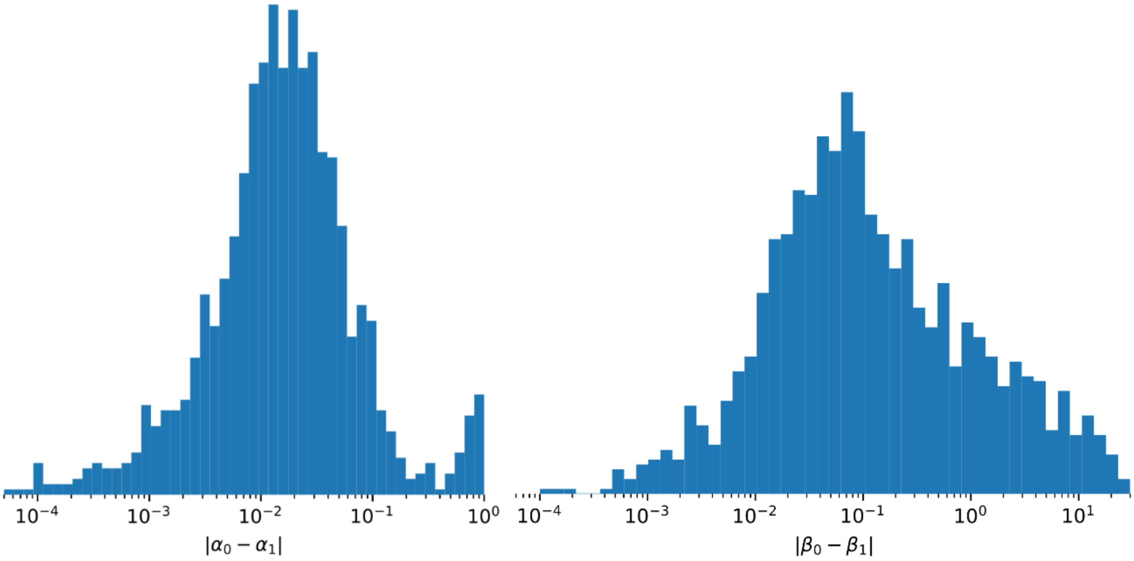


Figure 4-7 Histograms of the differences between two fits of the same data. The fits used are the same as shown in Figure 4-8. As most of the differences are very small, the bins are equally sized on a logarithmic scale (base 10). Both histograms have 50 bins. ***Left:*** *For α the range is between 0.00001 and 1* ***Right:*** *For β the range is between 0.0001 and 30.*

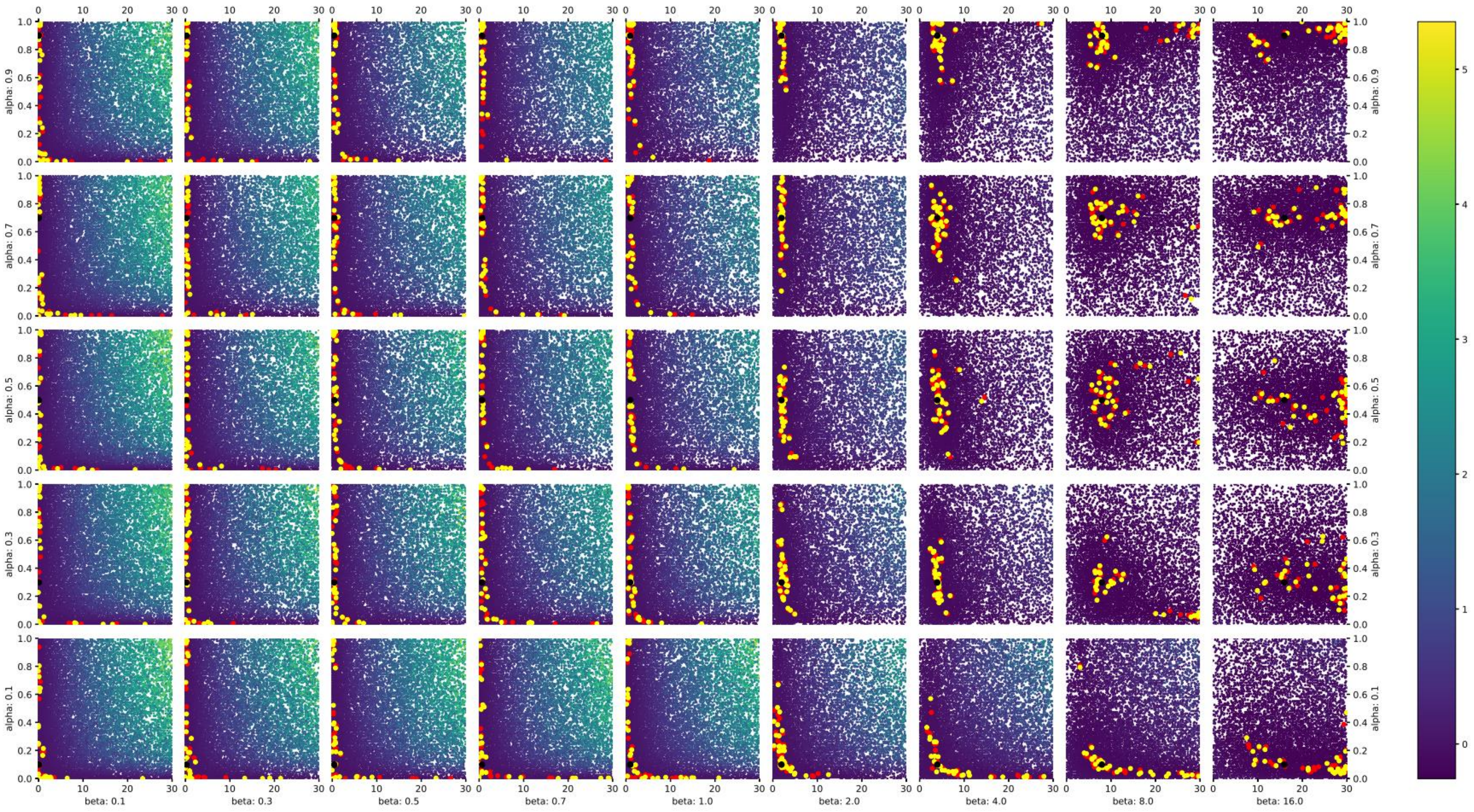


Figure 4-8 A set of parameter space plots showing the fitting process and results from the biased coins task with 100 trials, of which 20 are without feedback. The grid of plots is arranged according to the parameter values of the generated data, also marked by a black dot in each subfigure. The red dots are the parameters of the best fit for each of the 30 simulated datasets for each parameter combination. The yellow dots are the parameters of best fit for a subsequent refit of the 30 simulated datasets for each parameter combination. The other dots show the search locations during the 30 evolutionary fitting processes, coloured using the base 10 log of the fit quality.

To see how the best fit parameter distributions varied with a longer task, data was generated for a longer version of the Biased Coins task, with 1600 trials of which 100 at the end were without feedback. In Figure 4-9 the beginning of the convergence indicated by Reverdy & Leonard (2015) can be seen. However, even with this number of trials it is not possible for us to be certain about parameter value estimations outside of a small range. For us to therefore achieve usable predictions reliably, the tasks would need to be longer than 1600 trials, which is already much longer than those typically performed as human participants would fatigue and get bored over this timescale.

To confirm that this is not caused by evolutionary fitting method, the same dataset was fitted using gradient descent and the resulting best fits, shown in Figure 4-10, have the same distributions as found with the evolutionary fitting. Focusing specifically on $\alpha = 0.7$, $\beta = 4$, the best fit outside of the main 'valley' can still be seen. Given these plots, the fitting method, be that evolutionary fitting or gradient descent, can be ruled out as being the cause of the unusual best fit parameter distributions.

To verify that these results were not caused by an error in the coding of the Python framework, a simplified version was written in MATLAB. As no evolutionary algorithm is available in the standard MATLAB package, the MATLAB function *fmincon* was used for the fitting (Waltz, Morales, Nocedal, & Orban, 2006). This uses gradient descent and is the standard MATLAB fitting function. As can be seen in Figure 4-11, the distribution of fits is similar to those found previously. By comparing it with Figure 4-10, it can be seen that the search patterns of the two gradient descent implementations are similar. From these results, it appears that that the Python framework is working as intended and can be ruled out as the cause of these fit distributions.

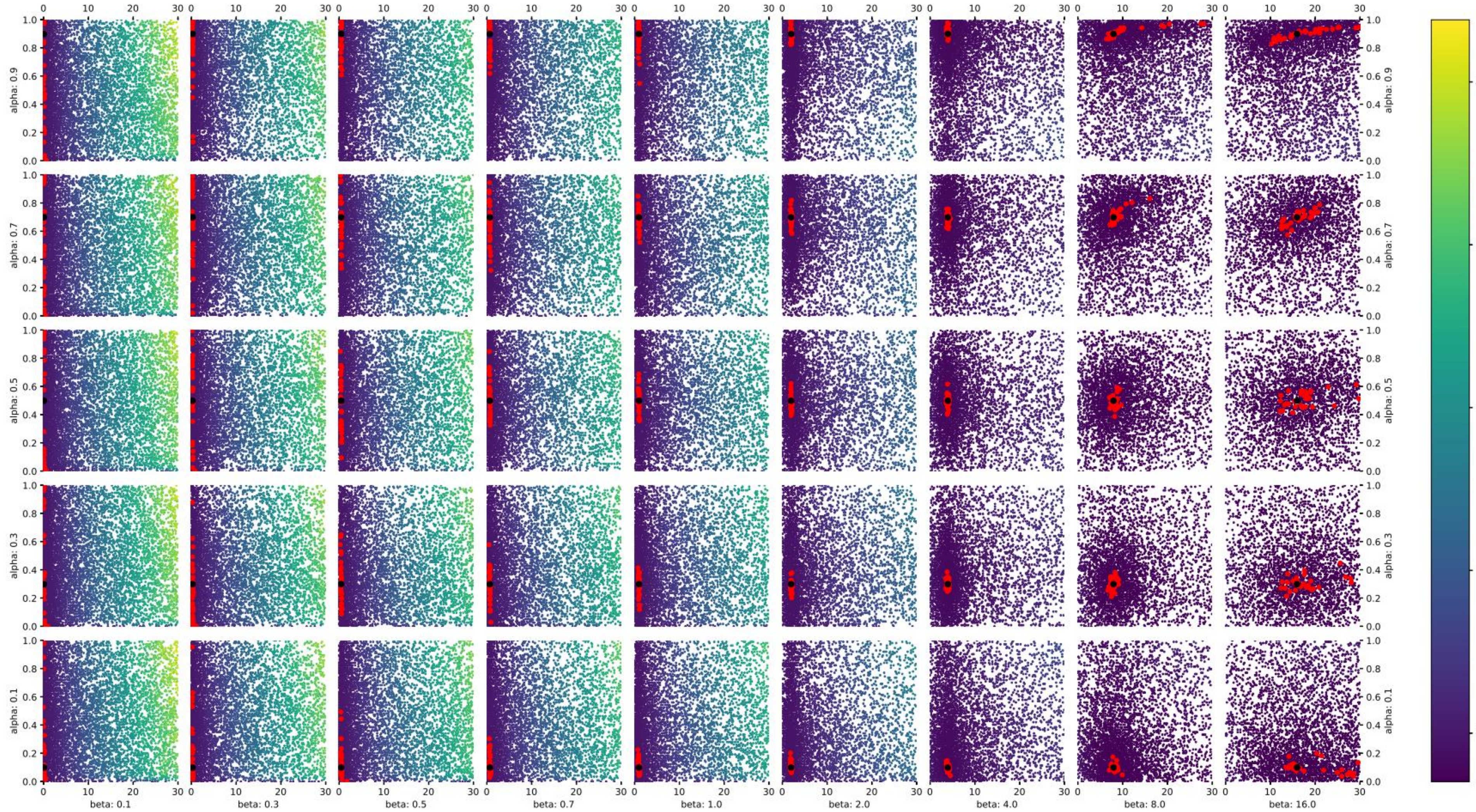


Figure 4-9 A set of parameter space plots showing the fitting process and results from the biased coins task with 1600 trials, of which 100 are without feedback. The grid of plots is arranged according to the parameter values of the generated data, also marked by a black dot in each subfigure. The red dots are the parameters of the best fit for each of the 30 generated datasets for each parameter combination. The other dots show the search locations during the 30 evolutionary fitting processes, coloured using the base 10 log of the fit quality.

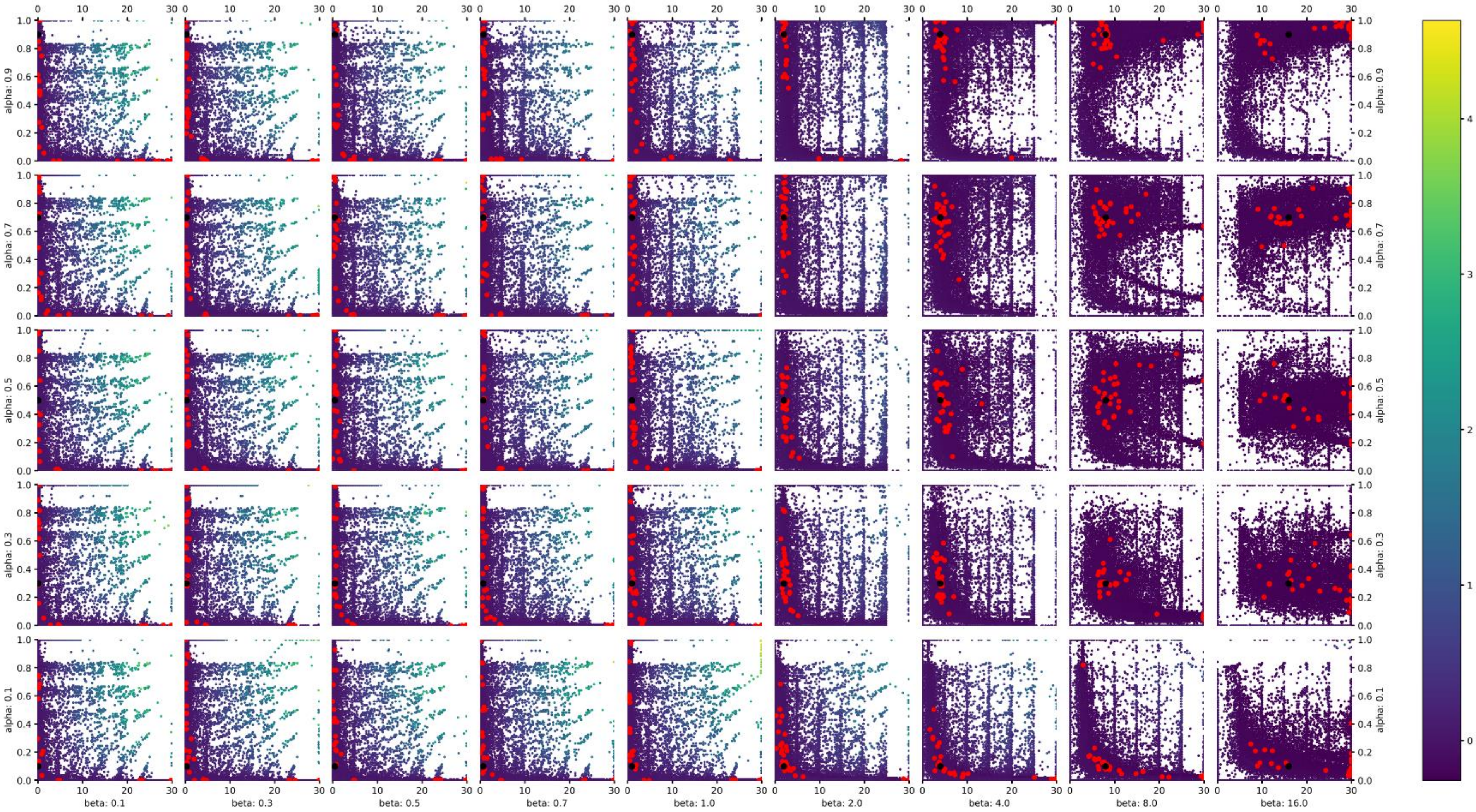


Figure 4-10 A set of parameter space plots showing the gradient descent fitting process and results from the biased coins task with 100 trials, of which 20 are without feedback. The grid of plots is arranged according to the parameter values of the generated data, also marked by a black dot in each subfigure. The red dots are the parameters of the best fit for each of the 30 generated datasets for each parameter combination. The other dots show the search locations during the 30 gradient descent fitting processes, coloured using the base 10 log of the fit quality.

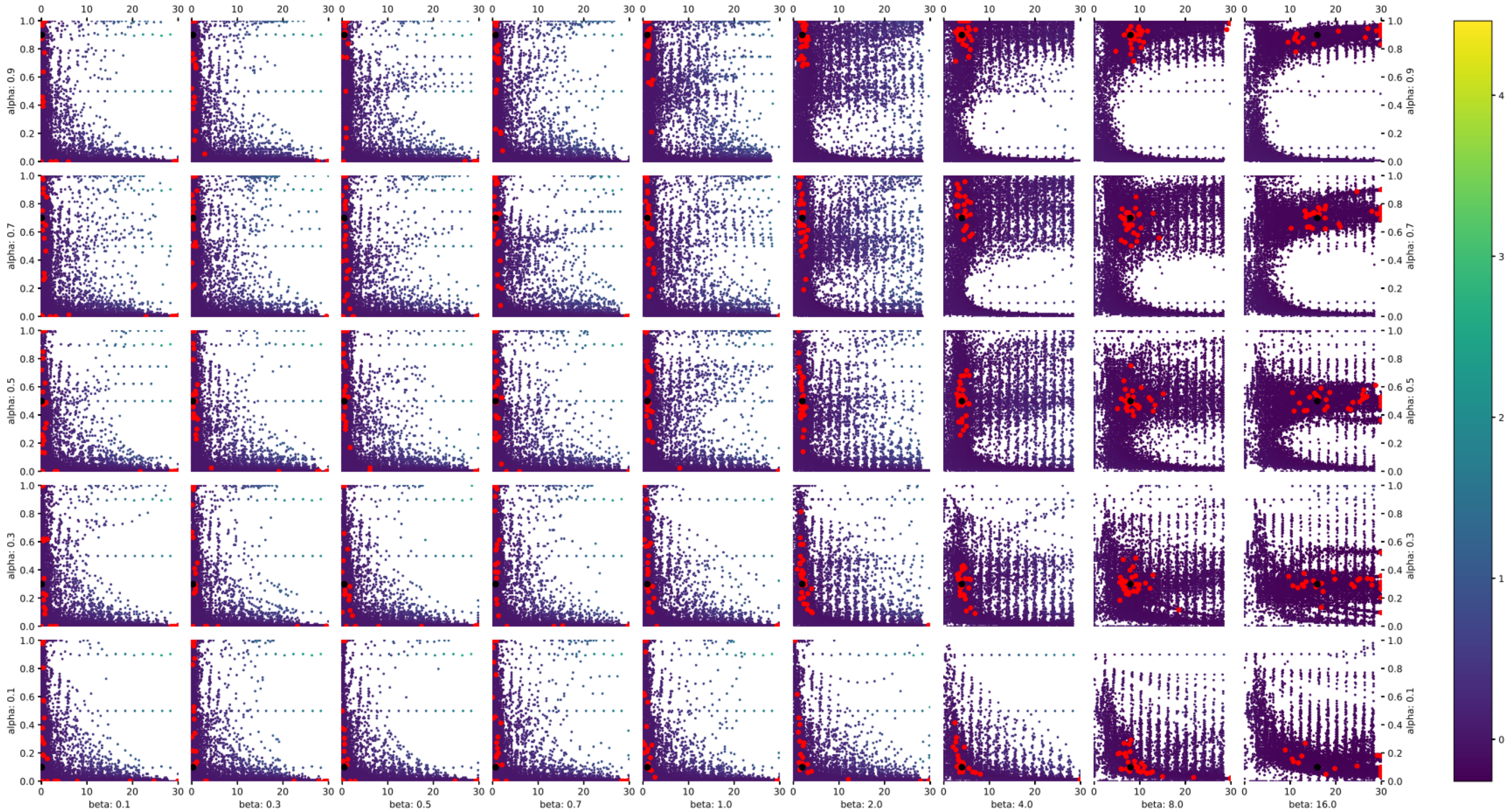


Figure 4-11 A set of parameter space plots showing the gradient descent fitting process and results from the biased coins task with 100 trials, of which 20 are without feedback. Both the data generation and the fitting were performed in MATLAB. The grid of plots is arranged according to the parameter values of the generated data, also marked by a black dot in each subfigure. The red dots are the parameters of the best fit for each of the 30 generated datasets for each parameter combination. The other dots show the search locations during the 30 evolutionary fitting processes, coloured using the base 10 log of the fit quality.

4.3 Fitness capacity for other tasks

To see if this issue is related to the Biased Coins task, data was generated for two other tasks.

The first is the Weather task, data for which is examined in chapter 8. This task is a step up in complexity to the biased coins task. The Weather task is a category learning task based on one described by Gluck & Bower (1988) and later adapted by Knowlton, Squire, & Gluck (1994). It asks participants to associate a series of cues with one of two outcomes. One to three cue cards, from a set of four cards, are presented to the participant in each trial. The participant must decide which one of the two possible outcomes the displayed cards are most likely associated with. Once the participant decides, they are then told if they were correct or not. The cues each have a probabilistic relationship with the two outcomes, with this this version of the task having novel probabilistic relationship, with the probability of an outcome varying depending on the combination of cues displayed, as described in Table 8-1. For example, if the first two cues are displayed, then the first outcome is guaranteed. If only one of them is displayed, then the first outcome will be the correct one 75% of the time. In the first phase of the task, the *learning phase*, participants are given feedback on if their choice was correct. In the second phase, the *testing phase*, participants are not given any feedback. The sequence of cues and the outcomes were fixed beforehand and is the third sequence shown in Figure 8-2, with 56 learning phase and 14 test phase trials, with equal numbers of each of the 14 cue combinations in each task phase.

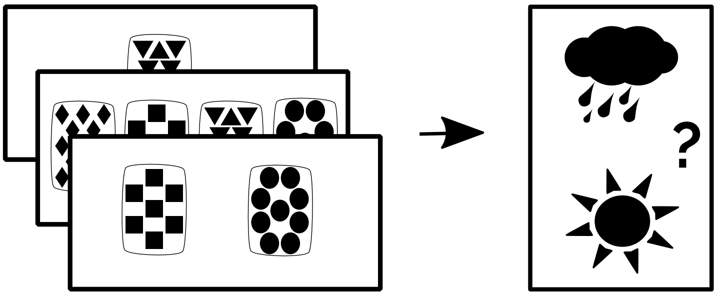


Figure 4-12 The Weather task consists of a series of trials where one to three cue cards, from a set of four cards, are presented to the participant. The participant must decide which of the two outcomes the cues are more likely to predict.

The Decks task provides a different type of probabilistic learning task, with no changes in stimuli and a wider range of reward. Data for this task is examined in chapter 6. It was based on a task used by Worthy, Maddox, & Markman (2007), and similar to the IOWA gambling task (Bechara et al., 1994). Participants were presented with two rectangles on a screen, one red and one blue. These were said to be the top cards of two decks of cards 80 cards long. Each 'card' has a predetermined reward associated with it, whose value was between one and ten. The objective was to maximise the accumulated card values by picking a card from one of the two decks each trial. The associated card value would then be shown to the participant. The sequences were kept the same throughout the task and there were not equal numbers of cards for each reward value. One of the decks was initially advantageous, but over the whole task provided lower rewards. The sequence can be seen in Figure 6-2. The card that was not chosen was not discarded, maintaining the number of available cards in each deck.

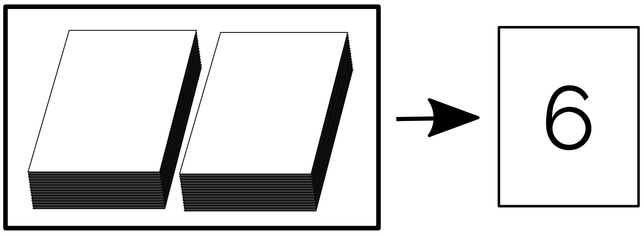


Figure 4-13 The Decks task consists of two decks of 80 cards. Each card has a value between one and ten. Participants choose during each trialstep which deck they thing will provide the most advantageous card, with an aim to accumulate the largest total card value. When a deck is chosen, the 'top' card from that deck is drawn, its associated reward is awarded to the participant and the card is discarded.

Comparing the fitted Weather task data in Figure 4-14 and the fitted Decks task in Figure 4-15 with those previously seen, there are variations in the distributions of recovered parameters for all three tasks, with the same changes occurring in those distributions across the generating parameter values. Therefore, the issues observed appear to be largely similar for these three tasks, two of which will have behavioural data fitted later in this thesis.

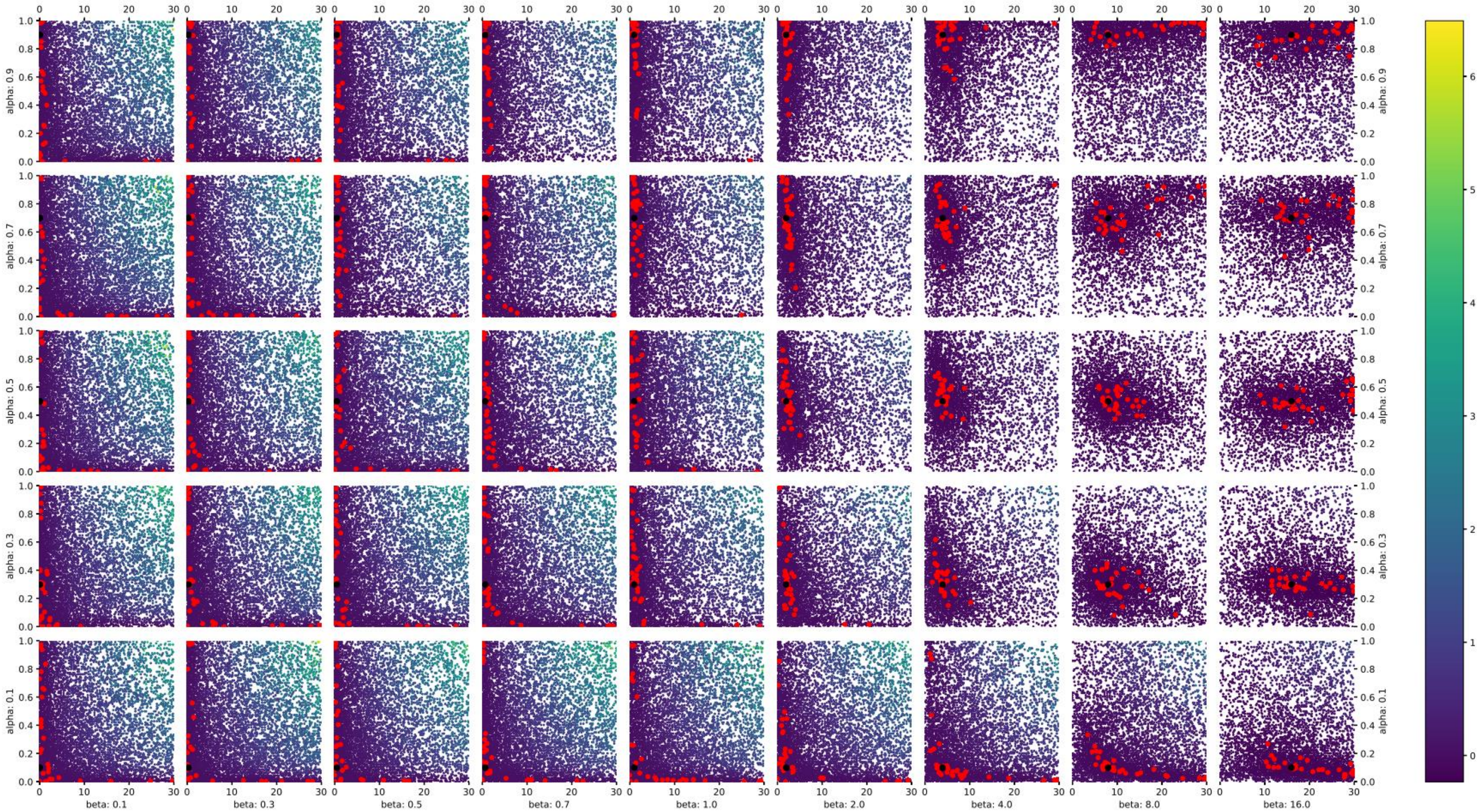


Figure 4-14 A set of parameter space plots showing the fitting process and results from the Weather task with 70 trials, of which 14 are without feedback. The grid of plots is arranged according to the parameter values of the generated data, also marked by a black dot in each subfigure. The red dots are the parameters of the best fit for each of the 30 generated datasets for each parameter combination. The other dots show the search locations during the 30 evolutionary fitting processes, coloured using the base 10 log of the fit quality.

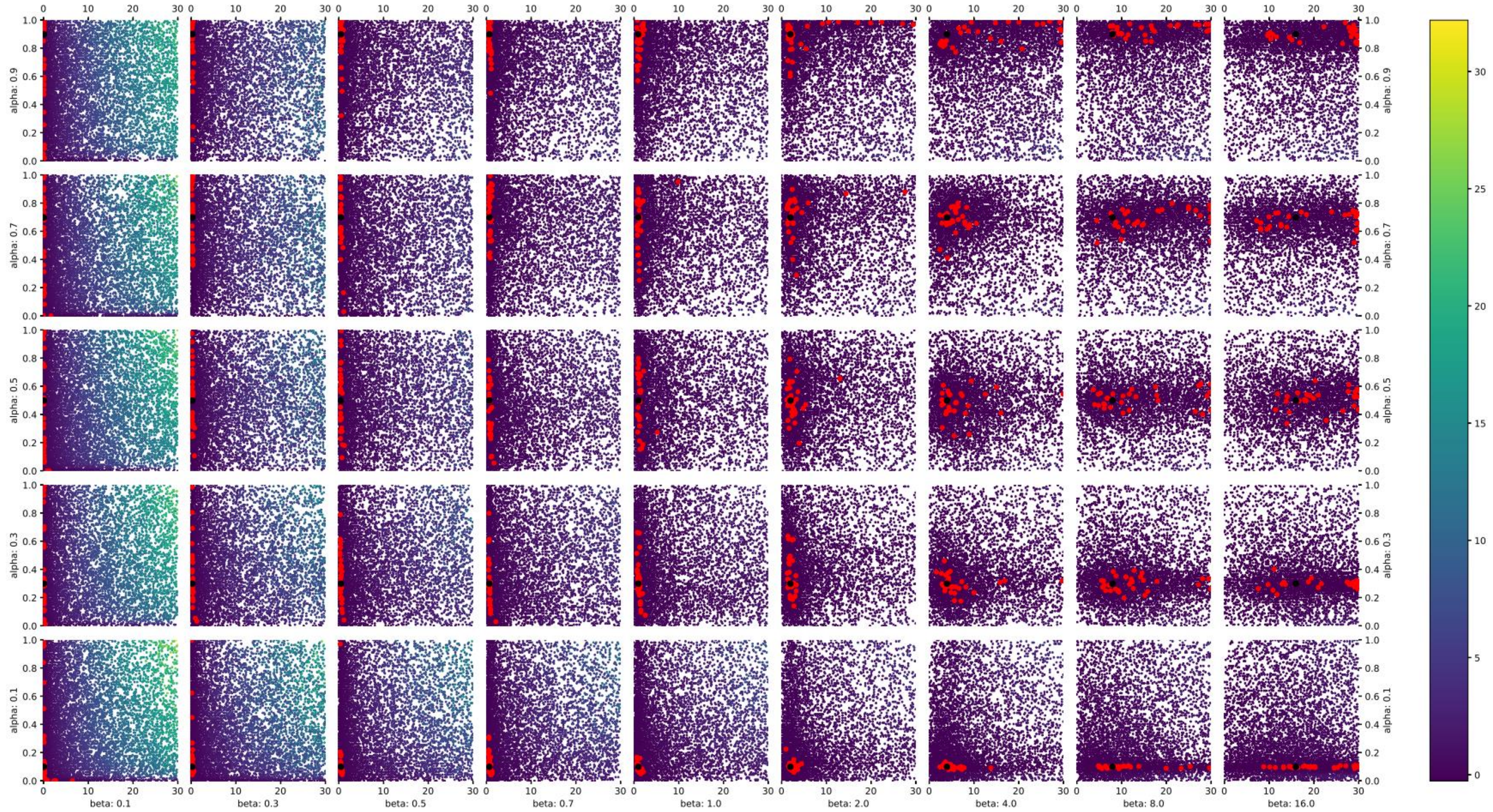


Figure 4-15 A set of parameter space plots showing the fitting process and results from the Decks task with 80 trials. The grid of plots is arranged according to the parameter values of the generated data, also marked by a black dot in each subfigure. The red dots are the parameters of the best fit for each of the 30 generated datasets for each parameter combination. The other dots show the search locations during the 30 evolutionary fitting processes, coloured using the base 10 log of the fit quality.

4.4 Individual fit distributions

To understand better what is going on in the model parameter fit distributions, the individual data fits were examined more closely. Figure 4-17 contains four fits from two different parameter value sets from the dataset generated for the Biased coins task examined in chapter 4.2. To help visualise what is occurring in the search for the minimum fit value, the figures only show the examined parameter value sets whose fit value was within 10% of the lowest value found. This gives us an understanding of the shape of the 'valley' containing the minimum. For $\alpha = 0.7$, $\beta = 4$ we can see that the minimum value is surrounded by the lowest other values found. This suggests that despite the best fit parameter values in repeat 20 being far from the generating parameter values, the recovered parameter values are in the middle of the minimal fit 'valley'. Equally, when looking at the fit value 'valleys' for $\alpha = 0.1$, $\beta = 0.5$ we see that the lowest fit values are surrounding the recovered best fit value. For both sets of parameter values, the individual fit value 'valley' bottoms are not necessarily surrounding the same areas but are clearly subsections of the valley produced when the fit values from all the fits are aggregated together. This suggests that we cannot treat each individual dataset generated with the same parameter values as having the same fit value parameter space shape. Compounding this issue is that these 'valley subsections' are not distributed around the true generating parameter values, as can be seen from Figure 4-6, reproduced in Figure 4-16, so even the average value is inaccurate when calculated from 30 runs of 100 trials.

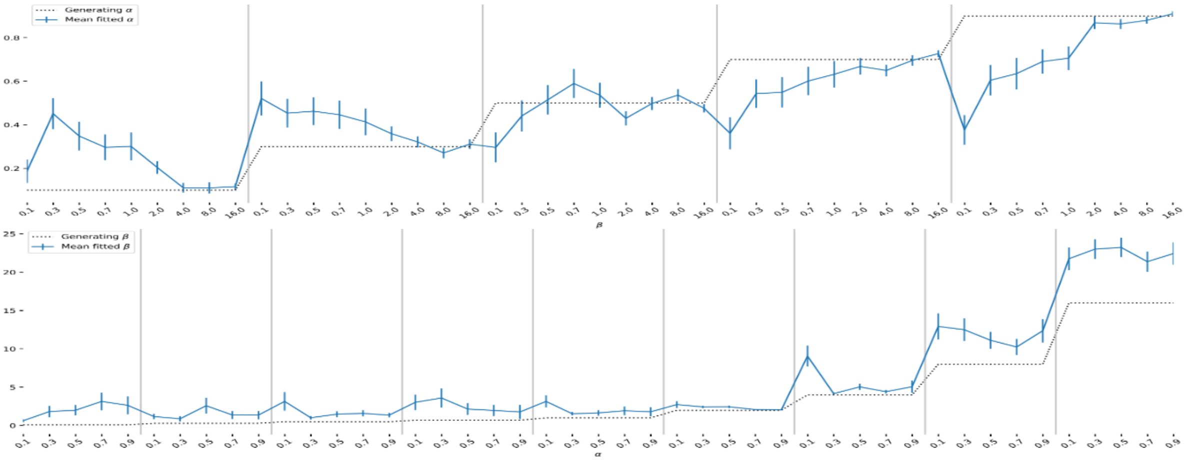


Figure 4-16 The means and standard deviations of the best fit values for α and β reproduced from Figure 4-6. ***Top:*** *Means of α* ***Bottom:*** *Means of β*

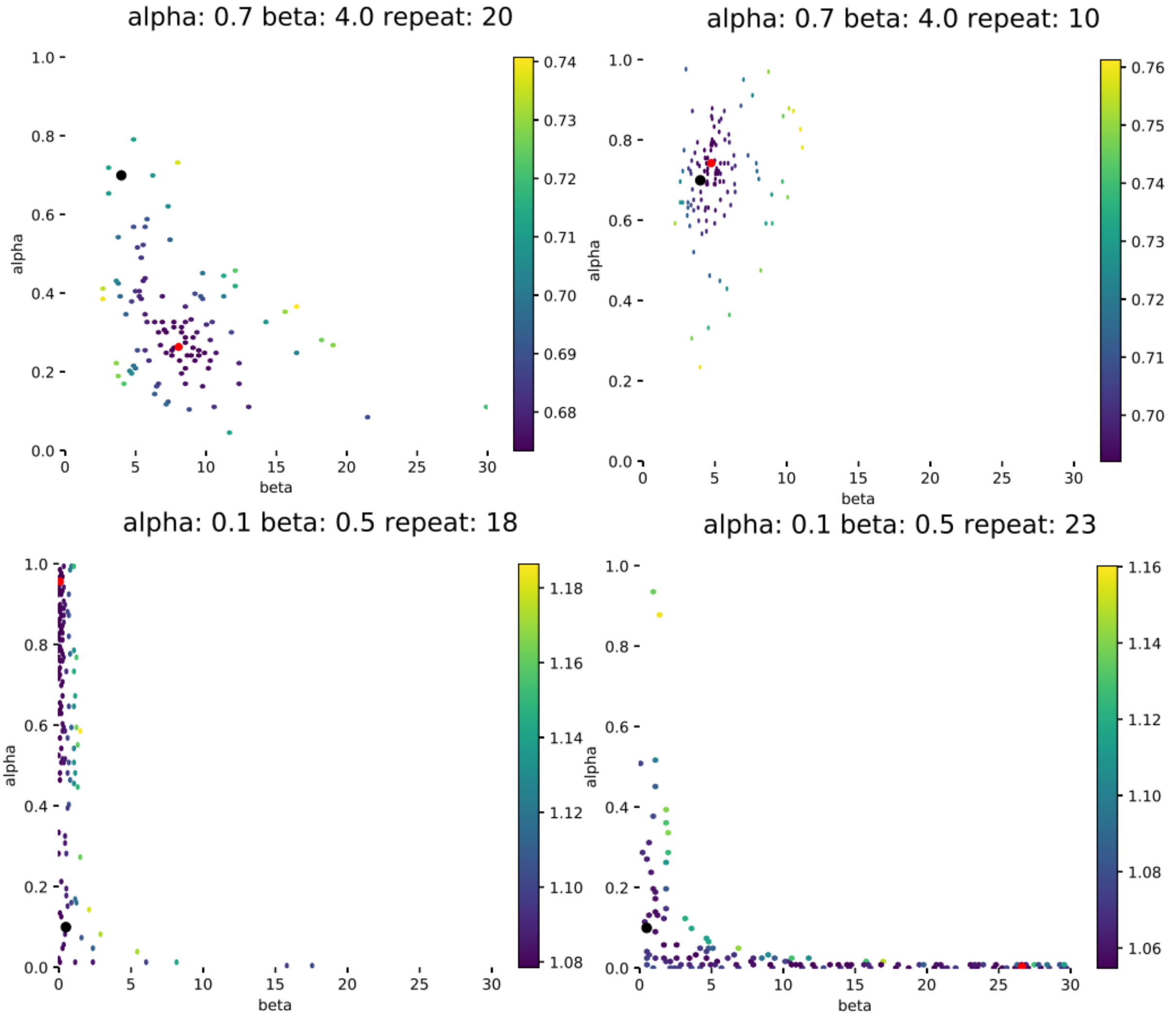


Figure 4-17 Parameter space plots for four of the fits of data generated for the biased coin task with 100 trials, of which 20 are without feedback. The red dots are the parameters of the best fit for each of the simulated datasets. The titles list the parameters, also marked by a black dot, from which the dataset were simulated for that plot. The other dots show the search locations during the evolutionary fitting process, coloured using the fit quality. Only those whose values are within 10% of the lowest fit value are shown, allowing us to have an idea of where the fit value 'valley' is situated.

4.5 Distinguishability of probability distributions

The fit quality value is calculated, as discussed in chapter 2.2, based on an aggregate of the probabilities of the chosen actions. Ultimately, when fitting using a maximum likelihood estimate, we are trying to maximise the likelihood for the model to take the actions that were taken. However, from what we have seen in the fitting process, there is a degree of misidentification of the parameters that generated the participant's actions. By looking at the expected rewards, $V_{d,\,t}$, and the resultant probabilities, $P_{d,t}$, calculated using the softmax function for the actions that were taken in the generated data, it was hoped an insight could be gained into the difficulties in distinguishing the generating parameter values when fitting the generated data.

This was examined for the qLearn model performing the Desk task, shown in Figure 4-18. As α increased, the range of expected rewards reached increased. This is unsurprising given that the largest change in expected reward increases with α; with an initial expected reward of 5.5 for each action, for $\alpha = 0.1$ the largest change at the beginning is $0.1 * (10 - 5.5) = 0.45$, whereas for $\alpha = 0.9$ it would be $0.9 * (10 - 5.5) = 4.05$. The result is that while it is possible to distinguish by eye the $\alpha = 0.1$ distribution from those of all the other α values tried here, this is not the case for the other values examined.

As can be expected from the softmax function, increasing β resulted in the probabilities becoming more sensitive to small changes in expected rewards, with most of the probabilities for an action at high β becoming either 1 or 0. Conversely, for low β, the probability distribution is squashed around 0.5, as seen more clearly in Figure 4-19. The result is that the transformation of the information by β can be thought of as masking the effect of α, thereby leading to the loss in accuracy when fitting α for low or 'high' values of β. More generally, parameters that act in these earlier stages of the model operation, such as α for the Q-learning model, will be harder to fit.

This can be seen most clearly by examining the mean value of the log probability of the chosen actions, shown in Figure 4-19. This is the mean value per trial of the log likelihood used in the model fitting, which can be expressed using the notation of equation 2.3 in chapter 2.2 as $f/2T$. The change in this value across α is smaller than those across β. The consequences of this lack of distinguishability can be seen in the expected rewards when fitting the same model on the action choices, shown in Figure 4-20. In all three of these figures, visual distinguishability increases as α and β decrease. However, for low values of one parameter, the differences in the other are less pronounced. This may explain the "L" shaped distribution of recovered parameters seen earlier in the chapter.

The probability distribution resultant from softmax is bimodal for β values near to or above 1. This is even more pronounced in the distribution of probabilities for the chosen action, which mixes the mirror image distributions of both actions. In none of these cases can these distributions be considered to come close to a normal distribution.

The softmax can therefore be considered to reduce identifiability of both its own β parameter and those parameters affecting the expected rewards earlier in the model. This is exacerbated by a *sloppiness* in the parameters (Brown & Sethna, 2003), as one parameter can compensate for the other, resulting in a negative covariance seen when fitting participant data in chapters 6.2.3, 7.2.3 and 8.2.3 and discussed in published works such as by Daw (2011).

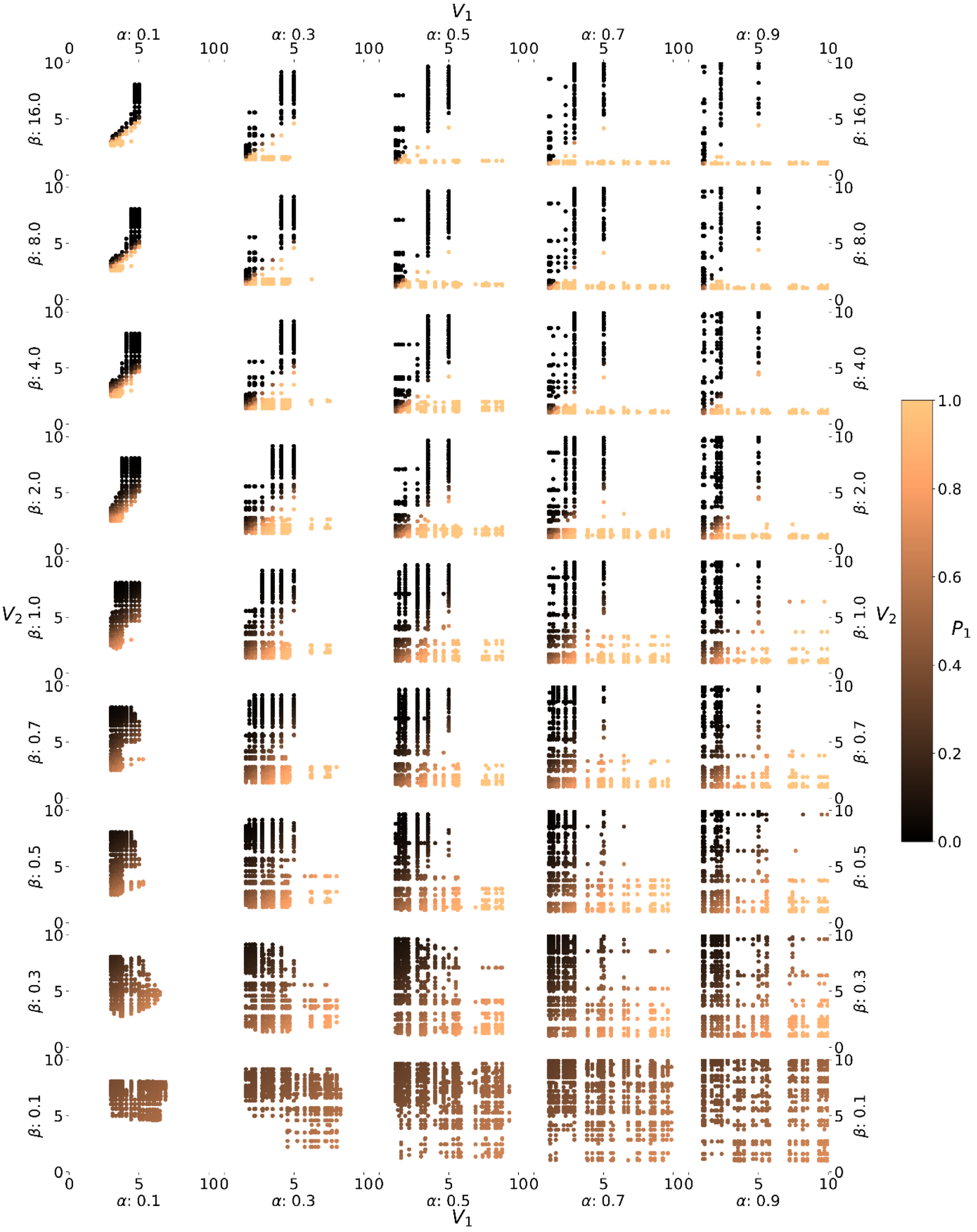


Figure 4-18 A set of scatterplots showing the expected rewards for each trial, $V_{d,t}$, for each action choice taken by the qLearn model when performing the Decks task using the generating parameters listed on the axis. The task had 80 trials, all with feedback and used the same deck sequence throughout. The colours denote the probability of choosing action 1, P_1. The grid of plots is arranged according to the parameter values of the generated data. Each plot contains the results from 30 runs of the task, so 2400 expected rewards.

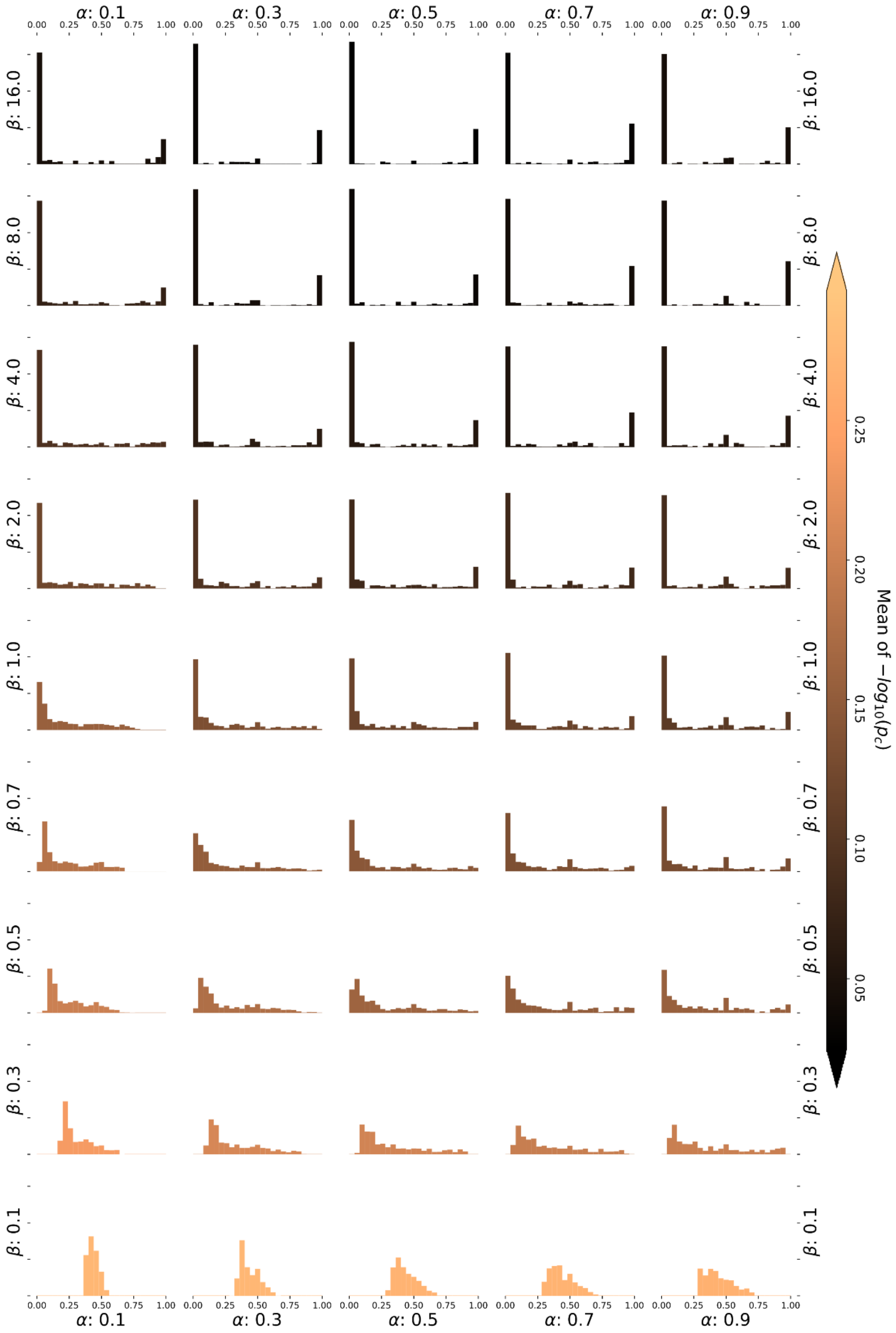


Figure 4-19 The distributions of the probabilities of action 1, for each action that was taken by the qLearn model when performing the Decks task using the generating parameters listed on the axis. The task had 80 trials, all with feedback and used the same deck sequence throughout. The grid of plots is arranged according to the parameter values of the generated data. The colours denote the mean value of the log10 of probability of the action that were chosen during the task run. Using equation 2.3 from chapter 2.2, this is equivalent to $f/2T$ for a perfect fit. Each plot contains the results from 30 runs of the task.

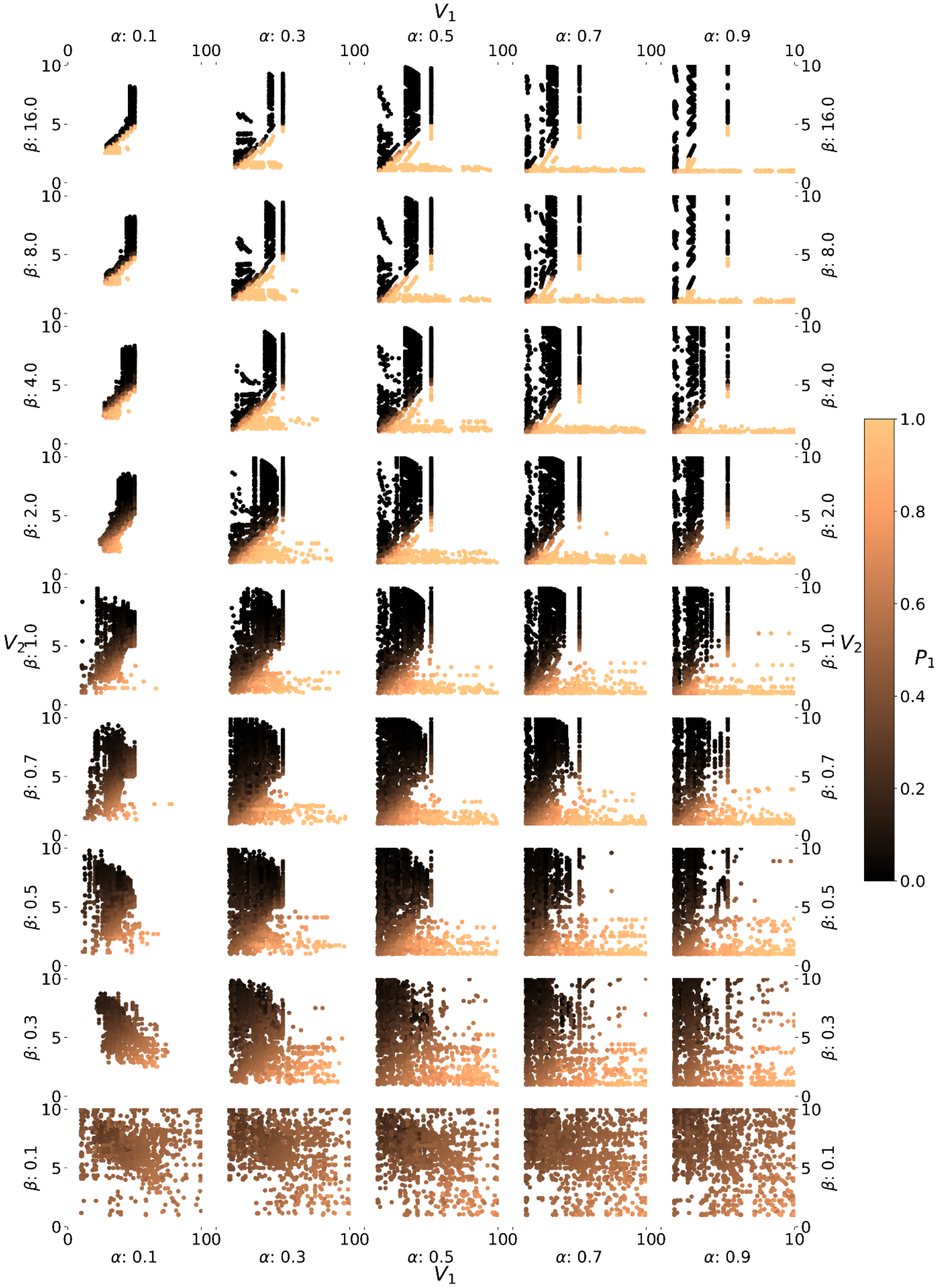


Figure 4-20 A set of scatterplots showing the expected rewards for each trial, $V_{d,t}$, for each action choice taken by the qLearn model when performing the Decks task using the parameters recovered from the task sequences shown in Figure 4-18. The task had 80 trials, all with feedback and used the same deck sequence throughout. The colours denote the probability of choosing action 1, P_1. The grid of plots is arranged according to the parameter values of the generated data. Each plot contains the results from 30 runs of the task, so 2400 expected rewards.

4.6 Potential solutions to distinguishing parameter combinations

The analysis shows that for these tasks when fitting using a maximum likelihood estimate, the recovered parameter values are located in the valley of best fit for the model given the data provided. However, for our datasets generated with known model parameters, the generating model parameters for a single simulation are frequently not close to the best fit region. This is likely due to the loss of information resulting from identifying a model's state using only the stochastic action choices. Several approaches were explored to reduce the noise when recovering model parameters.

One approach is to identify the posterior probability of possible generating parameters given the recovered model parameters. This can be calculated for a given model and task using a brute force method: by examining the fitting progression, the parameter space can be estimated for a given task run. From this the shape of the parameter space can be inferred. Doing so with simulated data, generated using the same model parameters, multiple examples of different parameter spaces can be seen for the same initial parameters. This has been done in the analysis so far in this chapter. From these, it would be possible to calculate for all points in the parameter space a distribution of likelihoods of a given fit quality for given generating parameters. This in turn would allow us to calculate the likelihood that the best fit can be found at each point in the parameter space. Having done this for one set of generating parameters, we could do the same for other combinations of parameters. This would provide us with the likelihoods for each generating parameter combination that the best fit would be found at a particular parameter combination. By combining these, it would be possible to generate a confidence estimate for each point in the parameter space. This was deemed too computationally intensive to be attempted without having a better idea of which models would turn out to be good models to fit to participant data.

In generated task data, the sequence of actions chosen by the model does not represent the sequence of most likely actions for the model, as the action for each trialstep is randomly chosen, with the likelihood of each action being the action choice probability, $P_{d,t}$. However, as during the fitting process this action sequence is the only available information on the participant’s thought process, the aim becomes to uncover the parameters that are the most likely to produce that action sequence. If it were possible to fit using the sequence of the participant’s most likely actions, then the error in the parameter recovery would be diminished. One possible approach to achieving this would be to perturb the actual action sequence, thereby exploring the action space around the sequence. As the consequences of the actions cannot be changed, this perturbation would be limited to adding noise to the model’s action choice probabilities used to calculate the fit quality measure. By changing only a few of these at a given time, it would be possible to calculate a fit quality measure for the model parameters for a small region in the action sequence space. This was explored, but with no success.

One approach proposed by Daw (2011), is to test a series of subjects sufficiently that it would be possible to get accurate estimates of their associated parameter values. The distributions of these parameter sets could then be used to generate prior probabilities for parameter combinations. While this might work, it would require significant effort and so would require confidence in the model being fitted to perform this.

Daw (2011) also suggested that the Bayesian information criterion may in fact be biased when compared to another fit measure Akaike information criterion (Akaike, 1974). However, the common, simplified forms of both of these assume that the errors in the model fitting are normally and identically distributed (Pitt, Myung, & Zhang, 2002). As has been shown, the distribution of action probabilities is far from normally distributed and this may be affecting our ability to fit simulated, as well as real data. As discussed in chapter 2.2, a recently proposed alternative to the BIC that would resolve this issue without significantly increasing the complexity of the fitting is the Widely Applicable Bayesian Information Criterion, WBIC (Watanabe, 2012). While this was explored there were some issues in the implementation that were not resolved.

Spektor & Kellen (2018) investigated how *maximum a-posteriori* (MAP) fitting could help improve parameter recovery. MAP weights the likelihood that parameters are the most likely using an estimate of the prior probabilities for each parameter combination. This is opposed to maximum likelihood estimating (MLE), described in chapter 2.2, which treats all parameter combinations as equally likely, i.e. MLE is MAP with a uniform prior. Spektor & Kellen used an informative prior based on a Gaussian mixture model distribution of recovered parameters from a separate dataset, in an attempt to approximate the population distribution. Gershman (2016), reported some success with this method, but Spektor & Kellen did not find any improvement in parameter recovery in simulated data unless the prior matches the population distribution exactly. As the empirical priors are likely to have been created from recovered parameters that were themselves unreliable, Spektor & Kellen found that using a prior can reinforce the recovery issues. It therefore does not get us closer to an initial understanding of the parameter population distribution, but would help were one to be found.

Another more drastic approach is to change the design of the tasks examined to maximise parameter recovery. Spektor & Kellen (2018), found three promising methods to improve recoverability: increasing the number of trials, increasing the number of available actions each trial and providing participants feedback for the actions they did not choose. As the datasets examined in this thesis were already collected, these options were not considered.

In the previous section, softmax was identified as having a detrimental effect on the capacity of a model to accurately fit task data. One solution to this is to modify or replace softmax with a function that allows for better parameter recovery while still providing the opportunity for individual differences.

4.7 Discernibility of alternatives to softmax

Having identified that the softmax function and its parameter β are a limiting factor to the recovery of model parameters, modifications to the softmax were examined for models based on Q-learning. The ones presented in chapter 3 are examined here: Q-learning autocorrelation, Meta Q-learning and Q-learning with epsilon greedy.

4.7.1 Q-learning autocorrelation

One small modification that could be made to the Q-learning model is to add an autocorrelation term, κ, to the action-choice probability calculation, as described in chapter 0.

As can be seen in Figure 4-21, Figure 4-22 and Figure 4-23, for extreme values of κ, the parameter is not well recovered. We can also see that there is no improvement in the recovery of α and β, so this does not act as a correction for the parameter recovery issues.

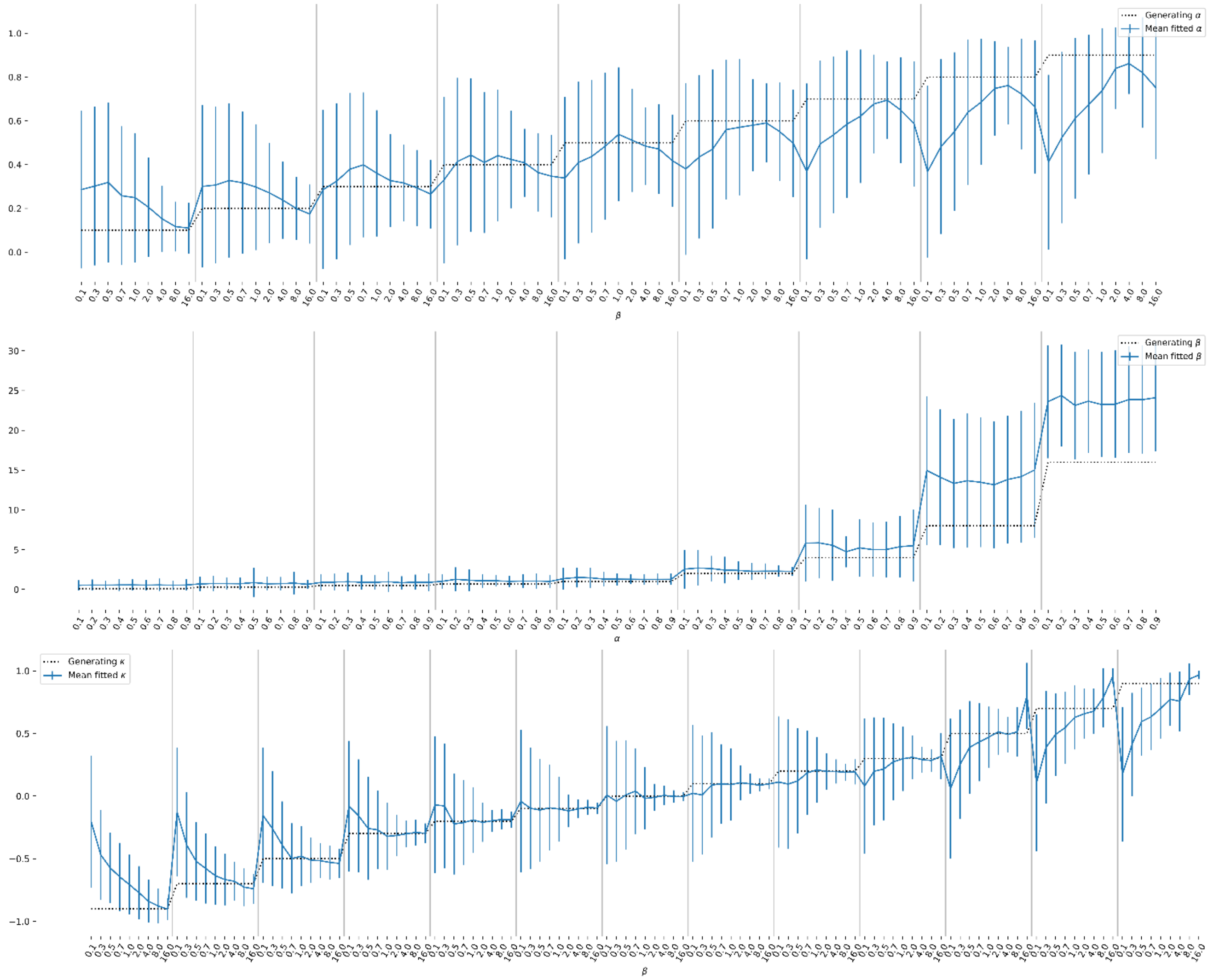


*Figure 4-21 The means and standard deviations of the best fit values for α, β and κ for the data generated from the biased coins task with 100 trials, of which 20 are without feedback. The means are calculated for each pair of parameter values used to generate the data. The dotted line denotes the generating values of those parameters. The graphs are plotted with the parameter values increasing in order such that the parameter increasing the fastest is the one shown on the horizontal axis. **Top**: The means of α for different generating β **Middle**: The means of β for different generating α **Bottom**: The means of κ for different generating β.*

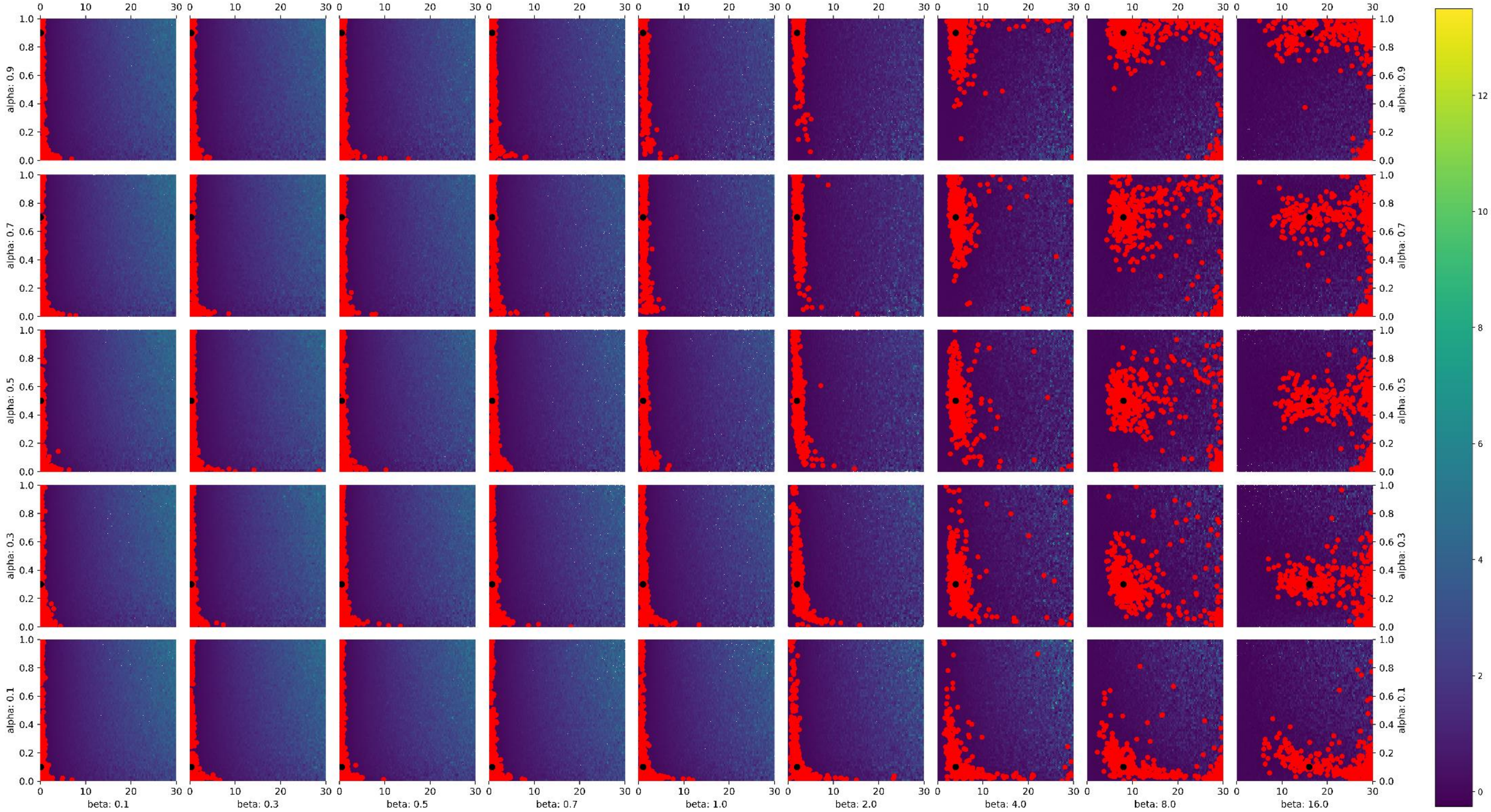


Figure 4-22 A set of parameter space plots showing the fitting process and results from the biased coins task with 100 trials, of which 20 are without feedback. The grid of plots is arranged according to the parameter values of the generated data, also marked by a black dot in each subfigure. The red dots are the parameters of the best fit for each of the 30 generated datasets for each parameter combination of α, β and κ. The other dots show the search locations during the evolutionary fitting processes, coloured using the base 10 log of the fit quality.

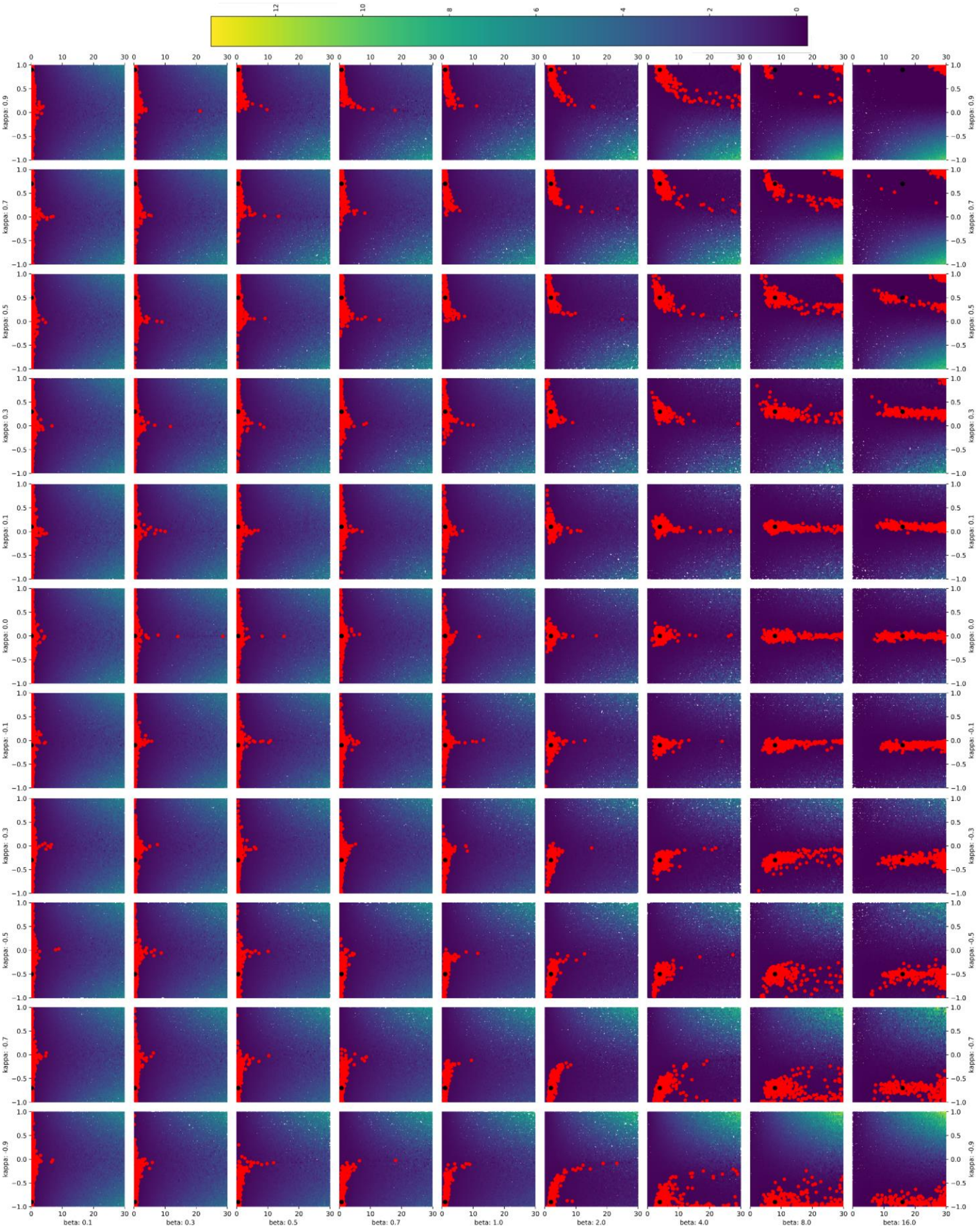


Figure 4-23 A set of parameter space plots showing the fitting process and results from the biased coins task with 100 trials, of which 20 are without feedback. The grid of plots is arranged according to the parameter values of the generated data, also marked by a black dot in each subfigure. The red dots are the parameters of the best fit for each of the 30 generated datasets for each parameter combination of α, β and κ. The other dots show the search locations during the evolutionary fitting processes, coloured using the base 10 log of the fit quality.

4.7.2 Epsilon greedy

An alternative approach is to replace the softmax-method of calculating action choice probabilities and replace it with an ε-greedy method, as described in chapter 3.7. As can be seen in Figure 4-24 and Figure 4-25, the recovery of the ε parameter is quite accurate, while the α parameter recovery is less accurate than that of ε, but still more than that seen in Figure 4-6 for the softmax. It is therefore a viable alternative to softmax that should be considered.

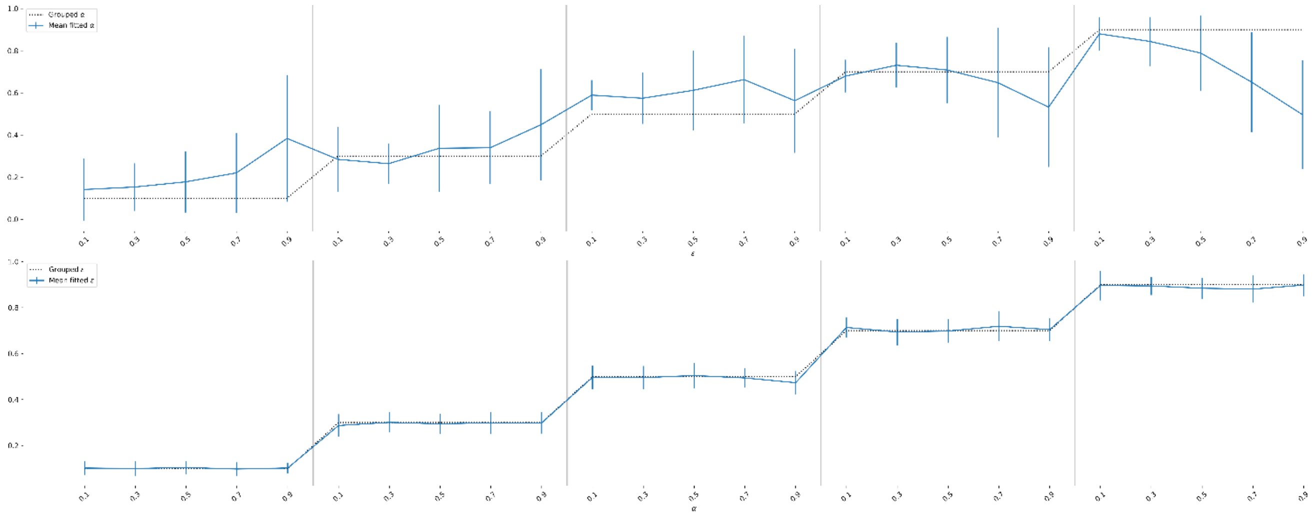


Figure 4-24 The means and standard deviations of the best fit values for α and ε for the data generated from the biased coins task with 100 trials, of which 20 are without feedback. The means are calculated for each pair of parameter values used to generate the data. The dotted line denotes the generating values of those parameters. The graphs are plotted with the parameter values increasing in order such that the parameter increasing the fastest is the one shown on the horizontal axis. ***Top****: The means of α for different generating ε* ***Bottom****: The means of ε for different generating α.*

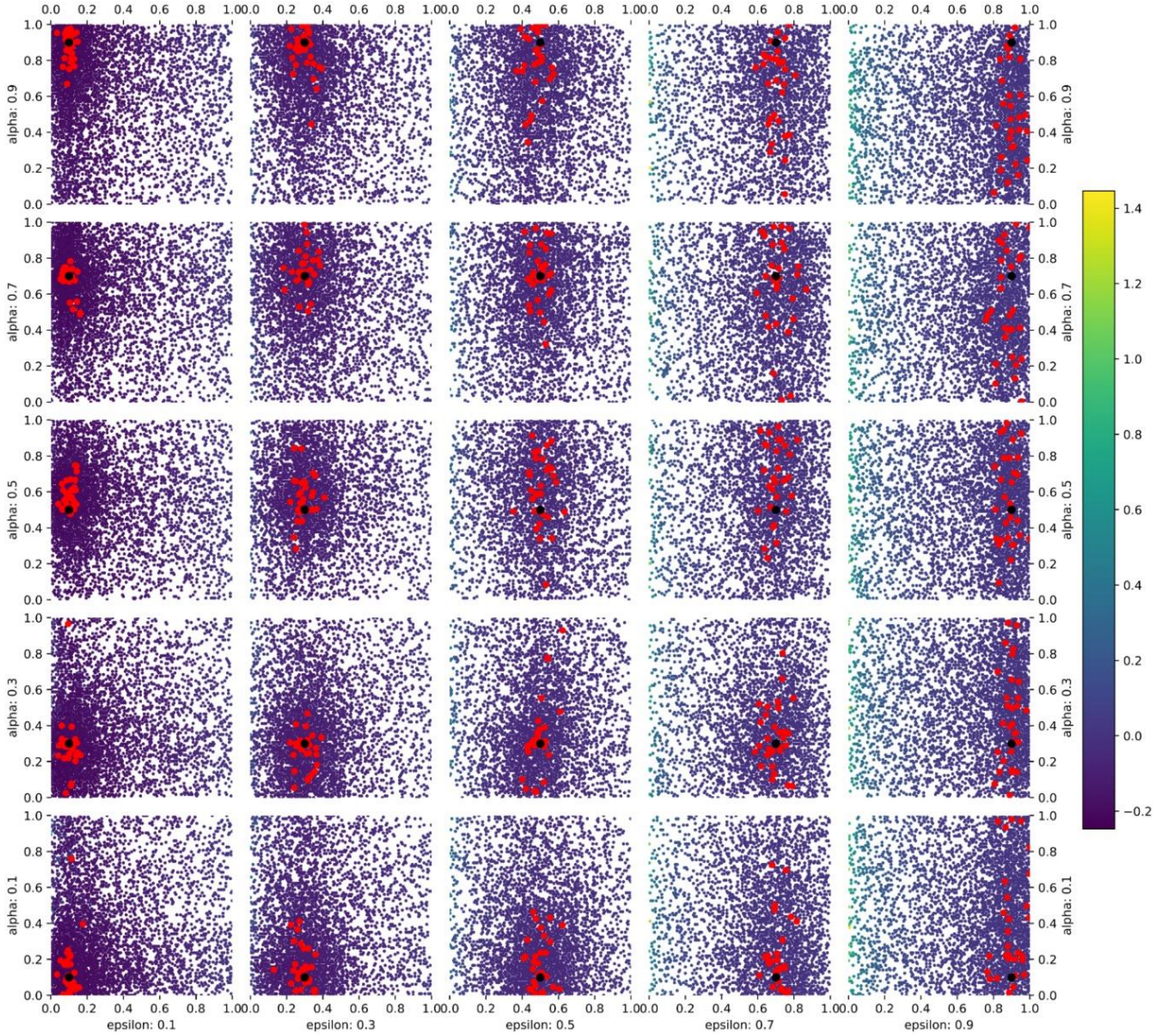


Figure 4-25 A set of parameter space plots showing the best fit parameters resulting from fitting the biased coins task with 100 trials, of which 20 were without feedback. The grid of plots is arranged according to the parameter values of the generated data, also marked by a black dot in each subfigure. The red dots are the parameters of the best fit for each of the 30 generated datasets for each parameter combination. The other dots show the search locations during the evolutionary fitting processes, coloured using the base 10 log of the fit quality.

4.7.3 Meta Q-Learning

The final option was the meta q-learning model, which uses τ as a learning rate for identifying the correct value of β. As can be seen in Figure 4-27 and Figure 4-26, the parameter recovery for τ is very poor for all parameter combinations, so this method is not useful for our purpose.

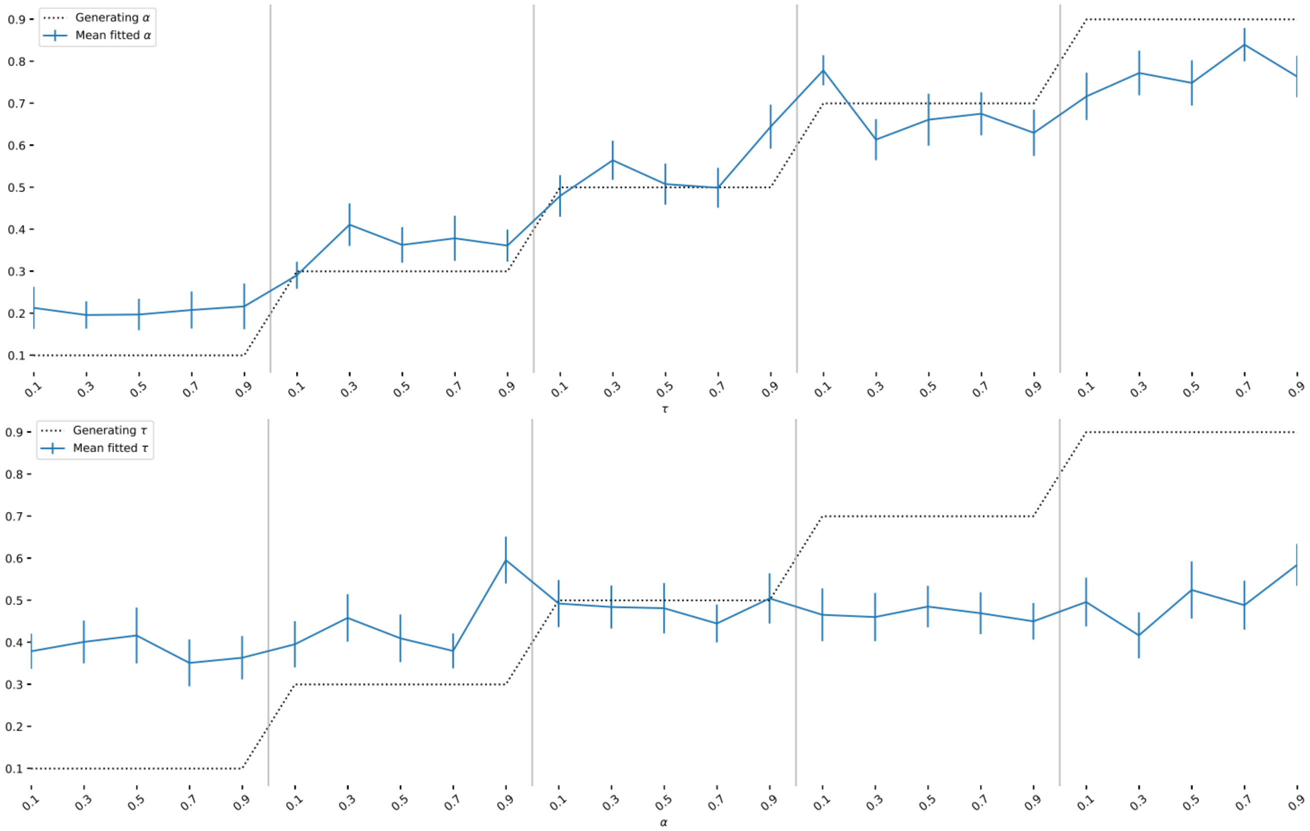


Figure 4-26 The means and standard deviations of the best fit values for α and τ for the data generated from the biased coins task with 100 trials, of which 20 are without feedback. The means are calculated for each pair of parameter values used to generate the data. The dotted line denotes the generating values of those parameters. The graphs are plotted with the parameter values increasing in order such that the parameter increasing the fastest is the one shown on the horizontal axis. ***Top****: The means of α for different generating τ* ***Bottom****: The means of τ for different generating α.*

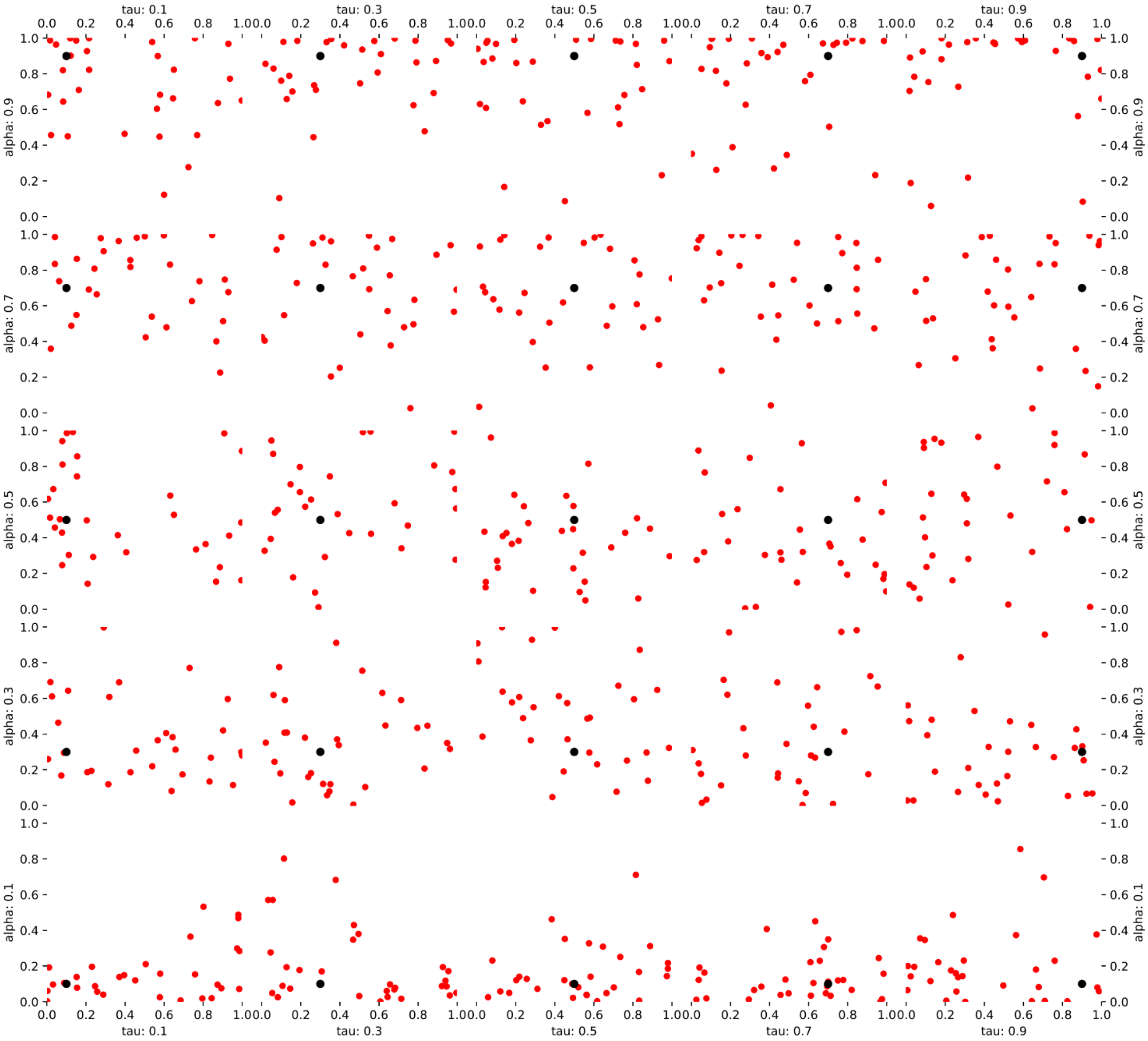


Figure 4-27 A set of parameter space plots showing the best fit parameters resulting from fitting the biased coins task with 100 trials, of which 20 were without feedback. The grid of plots is arranged according to the parameter values of the generated data, also marked by a black dot in each subfigure. The red dots are the parameters of the best fit for each of the 30 generated datasets for each parameter combination.

4.8 Additional models to assess

Based on the alternatives described in chapter 4.7, the only Softmax alternative that provided clearly better parameter recovery is the ε-greedy function. Here, the models discussed and implemented in chapter 3 are modified to take advantage of the ε-greedy function. These will be fitted alongside the original models in the following chapters.

Stages at t	OpAL with epsilon greedy (OpALE)
Reward expectation	$V_{d,t} = \sum_{s \in S_t} s_t E_{s,d,t}$
Action choice	$A^*_{d,t} = \sum_{s \in S_t} s_t A_{s,d,t}$ $B_{d,t} = \begin{cases} 1 & A^*_{d,t} = \max_d A^*_{d,t} \\ 0 & A^*_{d,t} < \max_d A^*_{d,t} \end{cases}$ $P_t = \left(\frac{1-\varepsilon}{\Vert B_t \Vert}\right) B_t + \frac{\varepsilon}{\Vert \mathcal{D}_t \Vert}$ $c_t = \mathcal{C}(P_t)$
RPE	$\delta_t = r_t - V_{c_t,t}$
Critic update	$E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha_C s_t}{\Vert S_t \Vert} \delta_t : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	$\left.\begin{array}{l} G_{s,d,t+1} = G_{s,d,t} + \frac{\alpha_G s_t}{\Vert S_t \Vert} G_{s,d,t} \delta_t \\ N_{s,d,t+1} = N_{s,d,t} - \frac{\alpha_N s_t}{\Vert S_t \Vert} N_{s,d,t} \delta_t \end{array}\right. : d = c_t$ $\left.\begin{array}{l} G_{d,t+1} = G_{d,t} \\ N_{d,t+1} = N_{d,t} \end{array}\right. : \mathrm{d} \neq c_t$ $A_{t+1} = (1+\rho) G_{d,t+1} - (1-\rho) N_{d,t+1}$

Table 4-1 The description of the OpAL-ε model, broken into the components used in the implementation.

Stages at t	OpAL with epsilon greedy without Hebbian update (OpAL_HE)
Reward expectation	$V_{d,t} = \sum_{s\in S_t} s_t E_{s,d,t}$
Action choice	$A^*_{d,t} = \sum_{s\in S_t} s_t A_{s,d,t}$ $B_{d,t} = \begin{cases} 1 & A^*_{d,t} = \max_d A^*_{d,t} \\ 0 & A^*_{d,t} < \max_d A^*_{d,t} \end{cases}$ $P_t = \left(\frac{1-\varepsilon}{\lVert B_t \rVert}\right) B_t + \frac{\varepsilon}{\lVert \mathcal{D}_t \rVert}$ $c_t = \mathcal{C}(P_t)$
RPE	$\delta_t = r_t - V_{c_t,t}$
Critic update	$E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha_C s_t}{\lVert S_t \rVert}\delta_t : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	$\left.\begin{aligned} G_{s,d,t+1} &= G_{s,d,t} + \frac{\alpha_G s_t}{\lVert S_t \rVert}\delta_t \\ N_{s,d,t+1} &= N_{s,d,t} - \frac{\alpha_N s_t}{\lVert S_t \rVert}\delta_t \end{aligned}\right. : d = c_t$ $\left.\begin{aligned} G_{d,t+1} &= G_{d,t} \\ N_{d,t+1} &= N_{d,t} \end{aligned}\right. : \mathrm{d} \neq c_t$ $A_{d,t+1} = (1+\rho)G_{d,t+1} - (1-\rho)N_{d,t+1}$

Table 4-2 The description of the OpAL-ε model without Hebbian update, broken into the components used in the implementation.

Stages at t	OpAL Saturated with epsilon greedy (OpALSE)
Reward expectation	$V_{d,t} = \sum_{s\in S_t} s_t E_{s,d,t}$
Action choice	$A^*_{d,t} = \sum_{s\in S_t} s_t A_{s,d,t}$ $B_{d,t} = \begin{cases} 1 & A^*_{d,t} = \max_{d} A^*_{d,t} \\ 0 & A^*_{d,t} < \max_{d} A^*_{d,t} \end{cases}$ $P_t = \left(\frac{1-\varepsilon}{\lVert B_t \rVert}\right) B_t + \frac{\varepsilon}{\lVert \mathcal{D}_t \rVert}$ $c_t = \mathcal{C}(P_t)$
RPE	$\delta_t = r_t - V_{c_t,t}$
Critic update	$E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha_C s_t}{\lVert S_t \rVert} \delta_t : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	$\begin{array}{l} G_{s,d,t+1} = G_{s,d,t} + \frac{\alpha_G s_t}{\lVert S_t \rVert} G_{s,d,t} \delta_t \left(1 - \frac{G_{s,d,t}}{M}\right) \\ N_{s,d,t+1} = N_{s,d,t} - \frac{\alpha_N s_t}{\lVert S_t \rVert} N_{s,d,t} \delta_t \left(1 - \frac{N_{s,d,t}}{M}\right) \end{array} : d = c_t$ $\begin{array}{l} G_{d,t+1} = G_{d,t} \\ N_{d,t+1} = N_{d,t} \end{array} : \mathrm{d} \neq c_t$ $A_{t+1} = (1+\rho) G_{d,t+1} - (1-\rho) N_{d,t+1}$

Table 4-3 The description of the OpALS-ε model, broken into the components used in the implementation.

Stages at t	Bayesian Probabilistic with epsilon greedy (BPE)
Reward expectation	$V_{d,t} = \mathbb{E}\left[\mathfrak{D}\left(\sum_{s\in S_t} s_t \omega_{r,s,d,t}\right)\right]$
Action choice	$B_d = \begin{cases} 1 & V_{d,t} = \max_{d} V_{d,t} \\ 0 & V_{d,t} < \max_{d} V_{d,t} \end{cases}$ $P_t = \left(\frac{1-\varepsilon}{\lVert B_t \rVert}\right) B_t + \frac{\varepsilon}{\lVert \mathcal{D}_t \rVert}$ $c_t = \mathcal{C}(P_t)$
RPE	-
Critic update	$\omega_{r,s,d,t+1} = \omega_{r,s,d,t} + \frac{\alpha s_t}{\lVert S_t \rVert} : \begin{cases} d = c_t \\ r = r_t \end{cases}$
Actor update	-

Table 4-4 The description of the Bayesian probabilistic model with epsilon greedy, broken into the components used in the implementation.

Stages at t	Temporal difference learning with epsilon greedy (tdE)
Reward expectation	$V_{d,t} = \sum_{s \in S_t} s_t E_{s,d,t}$
Action choice	$B_{d,t} = \begin{cases} 1 & V_{d,t} = \max_d V_{d,t} \\ 0 & V_{d,t} < \max_d V_{d,t} \end{cases}$ $P_t = \left(\frac{1-\varepsilon}{\lVert B_t \rVert}\right) B_t + \frac{\varepsilon}{\lVert \mathcal{D}_t \rVert}$ $c_t = \mathcal{C}(P_t)$ $E_{s_{t-1},d,\,t+1} = E_{s_{t-1},d,t} + \frac{\alpha\gamma s_{t-1}}{\lVert S_{t-1} \rVert} V_{c_t,t} \colon d = c_{t-1}$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_{t-1}$
RPE	$\delta_t = r_t - V_{c_t,t}$
Critic update	$E_{s,d,t+1} = E_{s,d,t} + \frac{\alpha s_t}{\lVert S_t \rVert} \delta_t \colon d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	--

Table 4-5 The description of the simplest version of the Temporal difference learning with epsilon greedy model, broken into the components used in the implementation.

Stages at t	Q-learning-ɛ with 2 learning rate parameters (qLearn2E)
Reward expectation	$V_{d,t} = \sum_{s \in S_t} s_t E_{s,d,t}$
Action choice	$B_{d,t} = \begin{cases} 1 & V_{d,t} = \max_d V_{d,t} \\ 0 & V_{d,t} < \max_d V_{d,t} \end{cases}$ $P_t = \left(\frac{1-\varepsilon}{\lVert B_t \rVert}\right) B_t + \frac{\varepsilon}{\lVert \mathcal{D}_t \rVert}$ $c_t = \mathcal{C}(P_t)$
RPE	$\delta_t = r_t - V_{c_t,t}$
Critic update	$E_{s,d,t+1} = E_{s,d,t} + \begin{cases} \frac{\alpha^+ s_t}{\lVert S_t \rVert} \delta_t & \delta_t > 0 \\ \frac{\alpha^- s_t}{\lVert S_t \rVert} \delta_t & \delta_t < 0 \end{cases} : d = c_t$ $E_{d,t+1} = E_{d,t} : \mathrm{d} \neq c_t$
Actor update	--

Table 4-6 The description of the Q-learning-ε model with two learning rate parameters, broken into the components used in the implementation.

4.9 Comparison to Bayesian model recovery

The issues that have been highlighted with some reinforcement learning models may explain why Bayesian models appear to perform better than reinforcement learning models when compared head to head, such as by Stankevicius, Huys, Kalra, & Seriès (2014). To see if this is the case, the parameter recovery performance for the simple Bayesian model is evaluated. This model, described in chapter 3.11, has the same parameters as the simple Q-Learning model discussed in this chapter, allowing a like-for-like comparison, where the only difference is the way in which the information is stored. The parameter recovery performance for this model, shown in Figure 4-28 and Figure 4-29, is similar to that seen for the Q-learning model, shown in Figure 4-5 and Figure 4-6, suggesting that any differences in performance are unlikely to be due to differences in fitting errors.

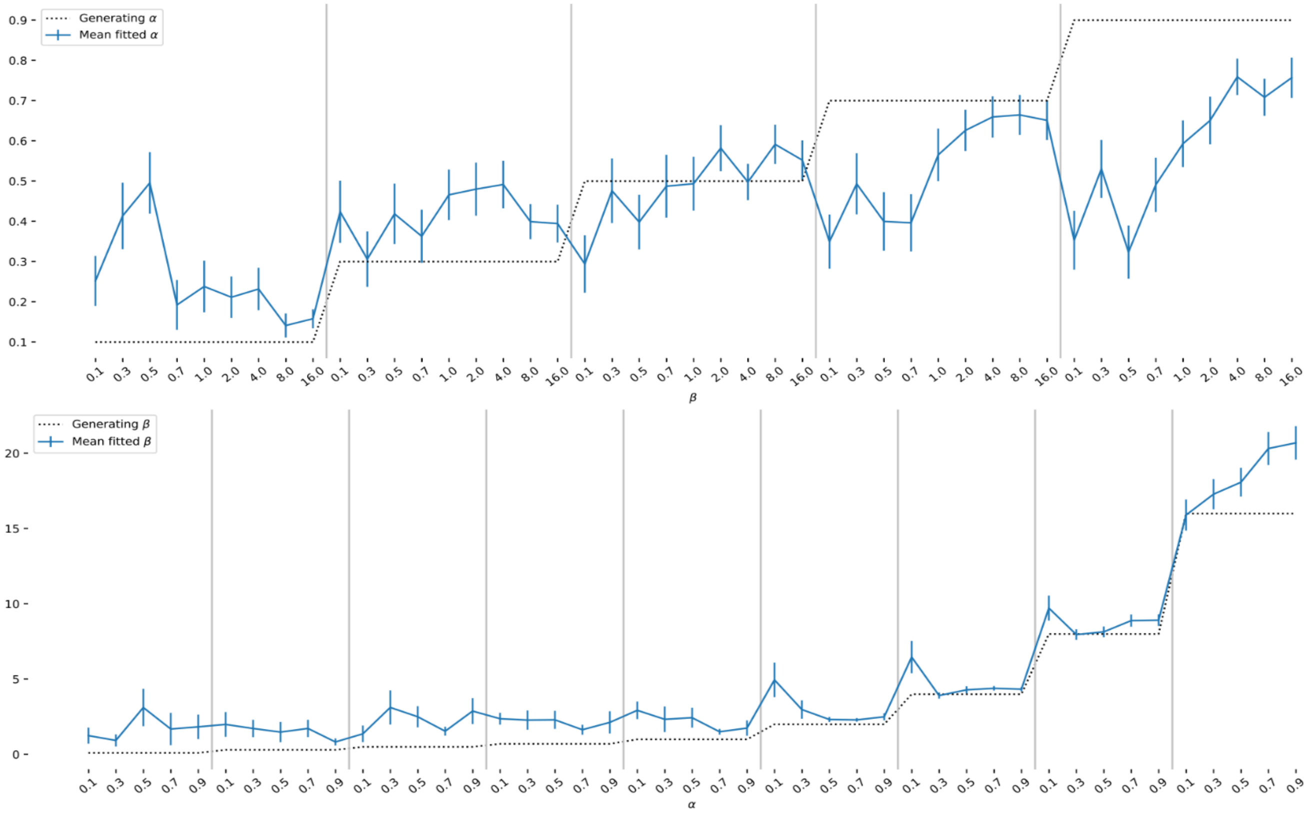


Figure 4-28 The means and standard deviations of the best fit values for α and β for the data generated from the biased coins task with 300 trials, of which 100 are without feedback. The means are calculated for each pair of generating parameter values. The dotted line denotes the generating values of those parameters. The graphs are plotted with the parameter values increasing in order such that the parameter increasing the fastest is the one shown on the horizontal axis. ***Top****: The means of α for different generating τ* ***Bottom****: The means of β for different generating α.*

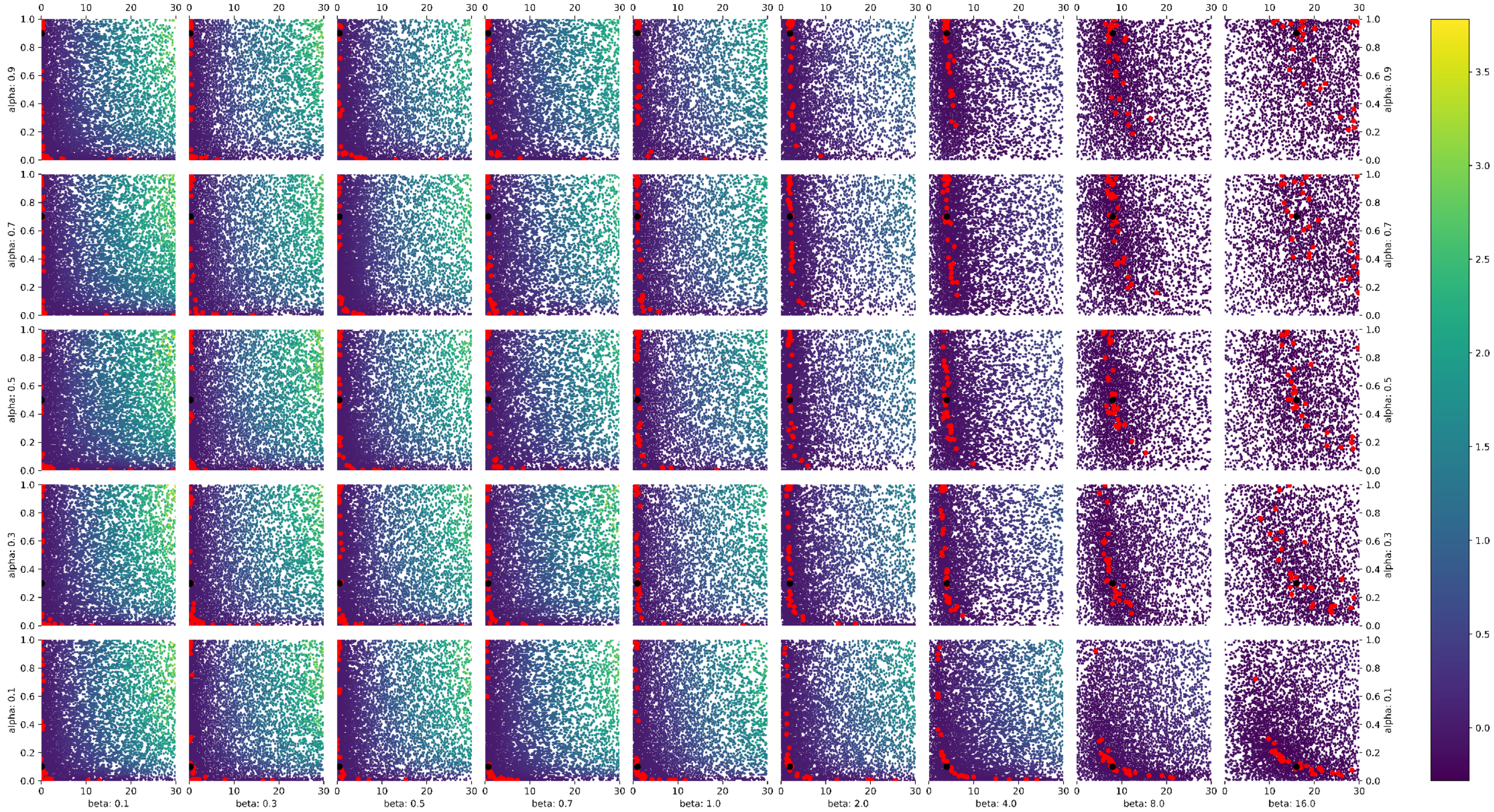


Figure 4-29 A set of parameter space plots showing the best fit parameters resulting from fitting the biased coins task with 300 trials, of which 100 were without feedback. The grid of plots is arranged according to the parameter values of the generated data, also marked by a black dot in each subfigure. The red dots are the parameters of the best fit for each of the 30 generated datasets for each parameter combination. The other dots show the search locations during the evolutionary fitting processes, coloured using the base 10 log of the fit quality.

4.10 Discussion

Here an examination was made of the performance of a standard model fitting method to accurately recover the parameter values from data generated with the same simple reinforcement learning model. Significant variability was found in the recovered parameter values and that the distribution of this variability changed as the generating parameter values changed. This has been shown to be consistent across two different codebases with two different fitting procedures. Repeated fitting of the same data with the same method has shown relatively negligible variation, suggesting that the error is not caused by poor identification of the global minimum, and so cannot be reduced by repeatedly fitting the data. As each action choice sequence could have been generated by many other parameter sets, this consistency suggests that a simple refinement to the fitting process will not be sufficient to accurately recover the parameters.

The generation and subsequent fitting of data across a variety of different tasks showed some variation in the distribution of the fitted parameter values, but the underlying issue was still present in all of them. This suggests that were a prior distribution to be used to improve parameter recovery, it would have to be recalculated for each task-model pair.

By looking at the chosen action probabilities for the generated data it became clear that the softmax β parameter is influencing the parameter recovery, biasing the fit value function to highlight parameter values with higher β as being better fits. Alternatives to the conventional softmax function were explored and compared, with the ε-greedy method found to be the most effective at providing discernible parameter values. Models were modified where possible to provide ε-greedy versions that could be used to evaluate the performance of the models.

Were this fitting approach to be used with this qLearn model on real participant data, where there is only one dataset, and so one fit, there would be sufficient uncertainty in the true parameter values as to not allow any conclusions to be drawn from them. This brings into question any cognitive learning results drawn from fitting individuals with a reinforcement learning model using a softmax, as any such model will have the effects of its other parameters squashed by those of β. As not only the recovered parameters, but their distributions are affected by this effect, studies looking at group level effects are also affected by this issue, as also attested by Humphries, Bruno, Karpievitch, & Wotherspoon (2015) for the expectancy valence model and Spektor & Kellen (2018) for Q-learning models with one or more learning rate parameter.

The simple Bayesian model, with the same parameters as the Q-learning model, exhibited similar issues to those found in the RL model, suggesting that the differences in performance seen between Q-learning and Bayesian models are not the result of different fitting errors.

One qualifier to this modification was found by Nassar & Frank (2016), who compared softmax and ε-greedy and came to the conclusion that irrespective of which is used, if the same one is not used to both generate and fit the data, this will have a significant impact on the types of errors generated when estimating the fit quality and so on the errors in parameter recovery. They also note that all fitting of this kind assumes that the attention of the participant does not slip during the task, as this would result in action choices chosen using another model. These 'attentional lapse' actions are not acknowledged by the fitting process and will add noise that cannot be estimated by the processes described here, but will have an impact on the accurate recovery of parameter values.

5 What parts of a task should be fitted and when?

In the previous chapter, the quality of parameter recovery for a typical reinforcement learning model was shown to be poor across a range of tasks and fitting methods. Parameter recovery was shown to improve with more information, i.e. as the length of the task increased, with good parameter recovery in a simple task requiring several thousand trials, far more than it takes for people to build a model of the task dynamics.

Previous studies have suggested that more accurate model parameters can be gained through fitting the model to a period of the task with no feedback, once the learning has occurred (Frank et al., 2007). They argued that fitting model parameters using the action choice probabilities from different parts of a task resulted in parameters that correlated with variations in different genes, suggesting that there were potentially two learning systems in play, a fast and a slow system. By fitting using the action choice probabilities from the first part of the task, the part with feedback, they argued that it was possible to identify a possible fast adapting learner, whereas by fitting using the model's action choice probabilities once feedback had been removed led to identifying a potential slow learning mechanism.

When considering this possibility, one issue to address is if the action choices for some parts of a task provide more information than others for fitting model parameters. For example, in a task with stochastic consequences that do not change over time, there will be an initial phase where the participant learns the expected feedback for each action. After which, the expected consequences will only be changing due to the variations in the task's feedback. In this second phase, as shown in Figure 5-1, if the model's learning rate is low, then the rate at which the model adapts to new information from the feedback is best identified during this initial exploratory phase, rather than by looking at the later action choices, as the changes in reward expectations due to the fluctuations from the varying feedback are small when the learning rate is low, resulting in barely perceptible changes in the action choice probabilities and so the distribution of chosen actions. However, if the learning rate is high, the convergence of the expected consequences to a stable average value will be very short but will fluctuate more once it has converged, as the information from each new piece of feedback will have a much larger influence on the expectation of the next trial. From this we can also infer that during a task's no-feedback phase, when a model no longer updates its knowledge of the task, a model will be more likely to have an accurate estimation of the expected consequences for a low learning rate than for a high learning rate. This can be seen from the smaller fluctuations in the estimated consequences for low learning rates during the post-convergence phase than for high learning rates, resulting a lower likelihood of the model having its estimation of the expected consequences be significantly different from that of the actual value for low learning rates. Therefore, during the no-feedback phase, the more the distribution of actions deviate from those that would result from an accurate estimate of the consequences, the more likely the higher learning rates are for the model.

As was seen in chapter 4.5, the size of the exploration-exploitation parameter in the softmax can also have a significant impact on the capacity to discern the underlying action-choice likelihoods. For high values of β, both small and large differences in the expected rewards are treated almost identically, resulting in

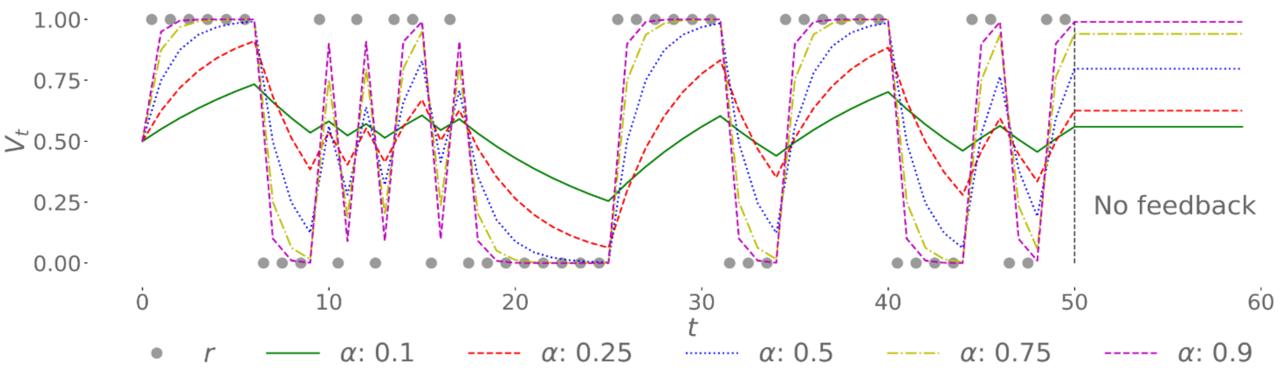


Figure 5-1 A toy example of how the learning rate in a simple Q-learning model affects the expected reward. The feedback, $r \in \{0,1\}$ is randomly chosen such that the expected reward should be 0.5, which is also the initial expected reward, i.e. the model has converged on the expected reward and is now fluctuating due to noise in the feedback. The no feedback portion, beginning on trial 50, has no updating of the expected reward.

similar behaviour to that seen for high α in simple reinforcement learning models such as Q-learning.

The models discussed in this thesis choose actions in the same way during the no-feedback portions of a task as for the feedback portions of the task: by choosing their next action randomly based on the action probability weightings. This approach is assumed in the discussion above. However, it is also possible that a person would use a winner takes all approach, where the most likely action is always taken. This would be equivalent to calculating the action probability weightings using $\beta = \infty$. As the weighted probabilistic approach encompasses both, it will be assumed for the rest of the chapter.

This chapter looks at how the quality of parameter recovery varies when fitting with the action choice probabilities from different parts of a task, and if this varies across models and across tasks. This is examined for the toy task used in chapter 4, the Biased coins task, as well as the two other tasks being examined in this thesis that have a period of no feedback: the Probabilistic Selection task and the Weather task. The Decks task is not examined as it does not have a no feedback portion and the likelihoods of different feedback varies across the task.

The fitting of three tasks was examined with the Q-learning model, as described in chapter 3.2, and the Q-learn-ε model, as described in chapter 3.7. For each task, a set of datasets was generated with the parameter value combinations from $\alpha = \{0.1,\ 0.3,\ 0.5,\ 0.7,\ 0.9\}$, $\beta = \{0.1,\ 0.3,\ 0.5,\ 0.7,\ 1,\ 2,\ 4,\ 8,\ 16\}$ and $\varepsilon = \{0.1,\ 0.3,\ 0.5,\ 0.7,\ 0.9\}$. For each combination, 30 task runs were generated. The model parameters were constrained during fitting to the ranges $\alpha = [0,1]$, $\beta = [0,30]$ and $\varepsilon = [0,1]$.

As has been seen in the previous chapter, the number of trials is an important factor in determining the quality of fits. Typically, the phase without feedback at the end of a task is kept quite short. However, in this analysis, both the parts with and without feedback will need to be of comparable length for us to be able to perform like-for-like comparisons. From past experience, for the tasks examined

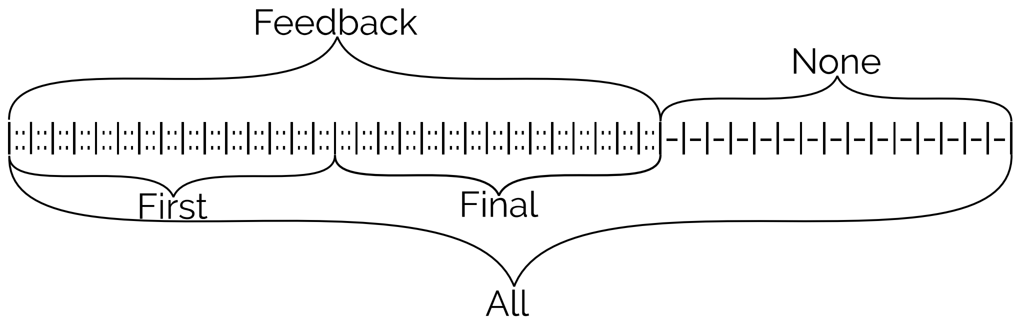


Figure 5-2 The task can be thought of as a series of trials, with each trial transition represented here as a vertical bar. For each trial the participant is either given feedback, denoted by "::", or no feedback, denoted as "-". The model performs the whole task before the performance of a model-parameter combination is evaluated. For each trial, the model's probability for the participant's chosen action is known. From the sequence of these probabilities, a subset can be chosen to evaluate the performance of this model with these parameters. Five of these subsets, labelled in this diagram, are being evaluated here. "All" uses all the task trials, "None" uses only the task trials with no feedback, "Feedback" uses only the task trials with feedback, "First" uses the first half of the task trials with feedback and "Final" uses the second half of the task trials with feedback.

here with stable rewards, it tends to take less than 100 trialsteps for both participants and models to converge on stable expected rewards. Based on this, the tasks are broken into three blocks of 100 trialsteps: the "First" 100, where convergence occurs, the "Final" 100 after convergence but still with feedback, and a last 100 trialsteps, "None", where there is no feedback. With this we can also add two other blocks: "Feedback", containing the 200 trials where the participant has feedback, and "All" the 300 trials, both with and without feedback. In all these five blocks, the model will perform the whole task before its performance is evaluated using the chosen block. This can be seen diagrammatically in Figure 5-2.

In chapter 4, it was shown that for the qLearn model many of the recovered parameters from the same generating parameters were spread across the full length of the parameter's *support*, the range over which the parameter is allowed to vary. Therefore, a useful baseline for comparing the errors in fits is to examine the distribution of errors that would be found if the recovered parameters were to be uniformly distributed across the whole parameter support. Figure 5-3 shows, for different generating parameter values, what a mean error would look like in this case. The maximal error of 50% of the support is at the extremities and the minimal error, of 25% of the support is in the middle. Any parameter recovery errors below this level would suggest a better than random parameter recovery.

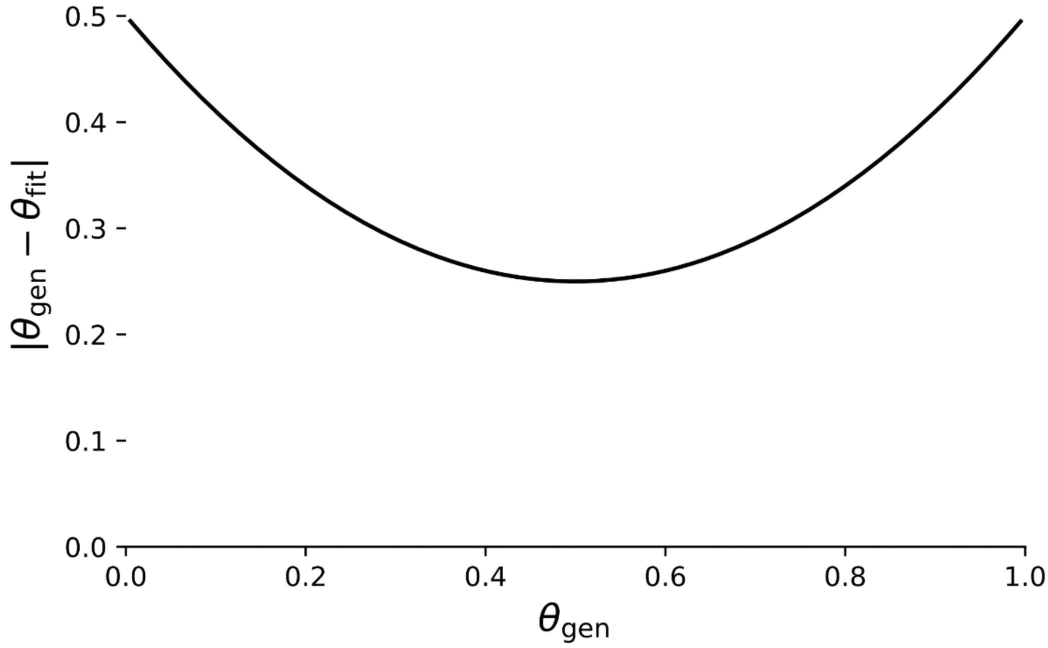


Figure 5-3 The mean absolute difference between numbers randomly picked between [0, 1] using a uniform random number generator, denoted θ_{fit}, and a range of values, acting as fake generating values, across the parameter support, [0, 1], denoted θ_{gen}. One million numbers were drawn with a uniform random number generator and compared to values between 0 and 1 increasing in 0.005 increments, acting as the θ_{gen}.

5.1 Biased coins

To provide a link to the discussion in chapter 4, the fitting performance for the Biased coins task was examined. Four distinct coins are shown to the participant. At the beginning of each trial, one of these is randomly chosen to be flipped and the chosen coin is identified to the participant. Before it is flipped the participant guesses if it will land on side 0 or side 1. The participant is then told, during the feedback trials, which side the coin landed on.

The probability of landing on one side or another varies from coin to coin, with two of the coins having an 80% chance of landing on side 0 and the other two a 20% chance of landing on side 0. For each generated dataset, the choice of coins and the side the 'coins' fall on is varied. This allows us to get a better estimate of the distribution of the noise in fitting, with not just the noise of the probabilistic decision making of the model but also the impact of the random task sequence.

The mean recovered parameters for each qLearn model generating parameter set are shown in Figure 5-5. The worst parameter recovery was provided when using only the no-feedback, None, parts of the task, while recovery using trials with feedback, First and Final, provide noticeably better fits, especially when recovering α from data generated with low α and high β. When fitting data generated with a high β, there is a significant increase in the error in recovering β, with an error of half the parameter support for those recovered using the trials within the Final and None ranges. The parameter recovery error of α decreases as the generating β

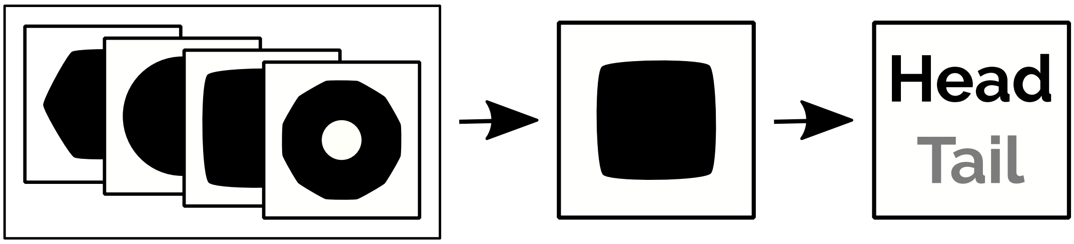
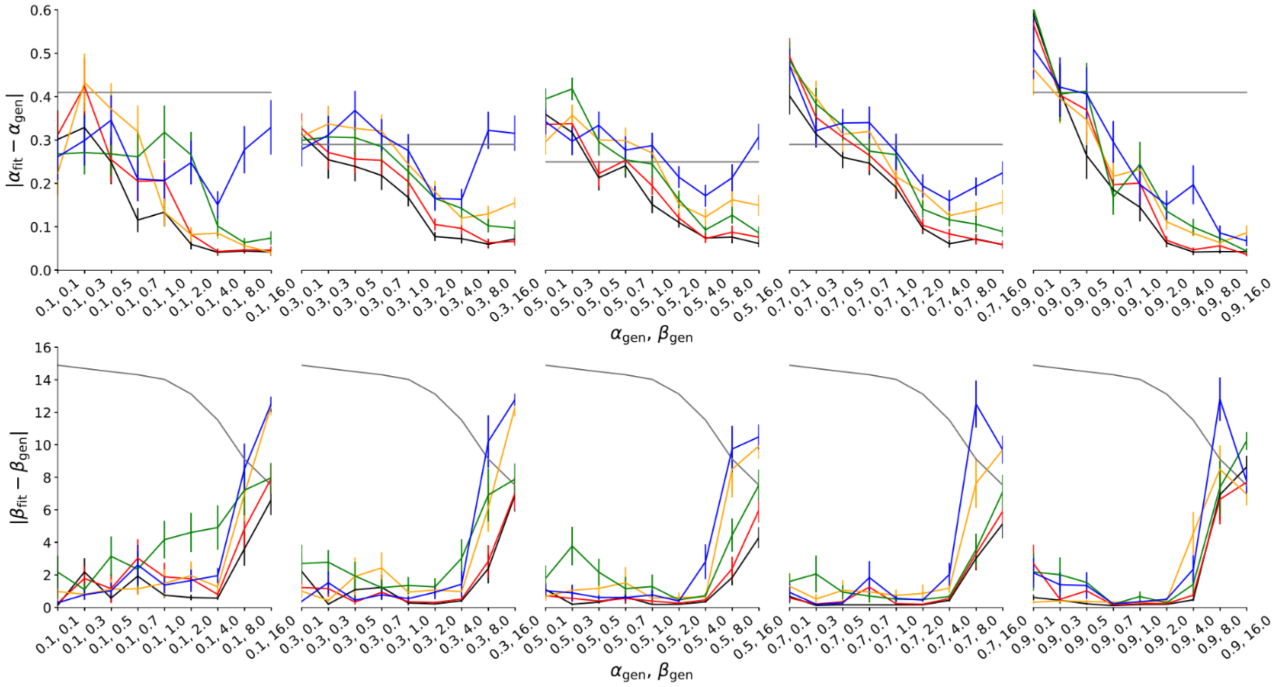


Figure 5-4 The Biased coins task. At the beginning of each trial, from a set of distinct coins, one is randomly chosen to be shown to the participant. Before it is flipped the participant guesses if it will land on side 0 or side 1. The participant is then told, during the feedback trials, which side the coin landed on. Each coin has a different probability of landing on each side.

increases and increases for low β as the generating α increases. For low β values and $\alpha > 0.1$, the α values are recovered worse than if they were randomly selected from a uniform distribution. For high β values, the β recovery is worse than if it were randomly selected from a uniform distribution. However, as a positively skewed distribution is a more reasonable prior for β, the uniform random average error is an underestimate of what could be expected.

As expected from chapter 4, doubling and tripling the number of trials used to recover parameters from the data does improve the accuracy, but this improvement is proportionally lower than the increase in trials. It is not clear if the All performance would have improved were all the 300 trials to be with feedback,



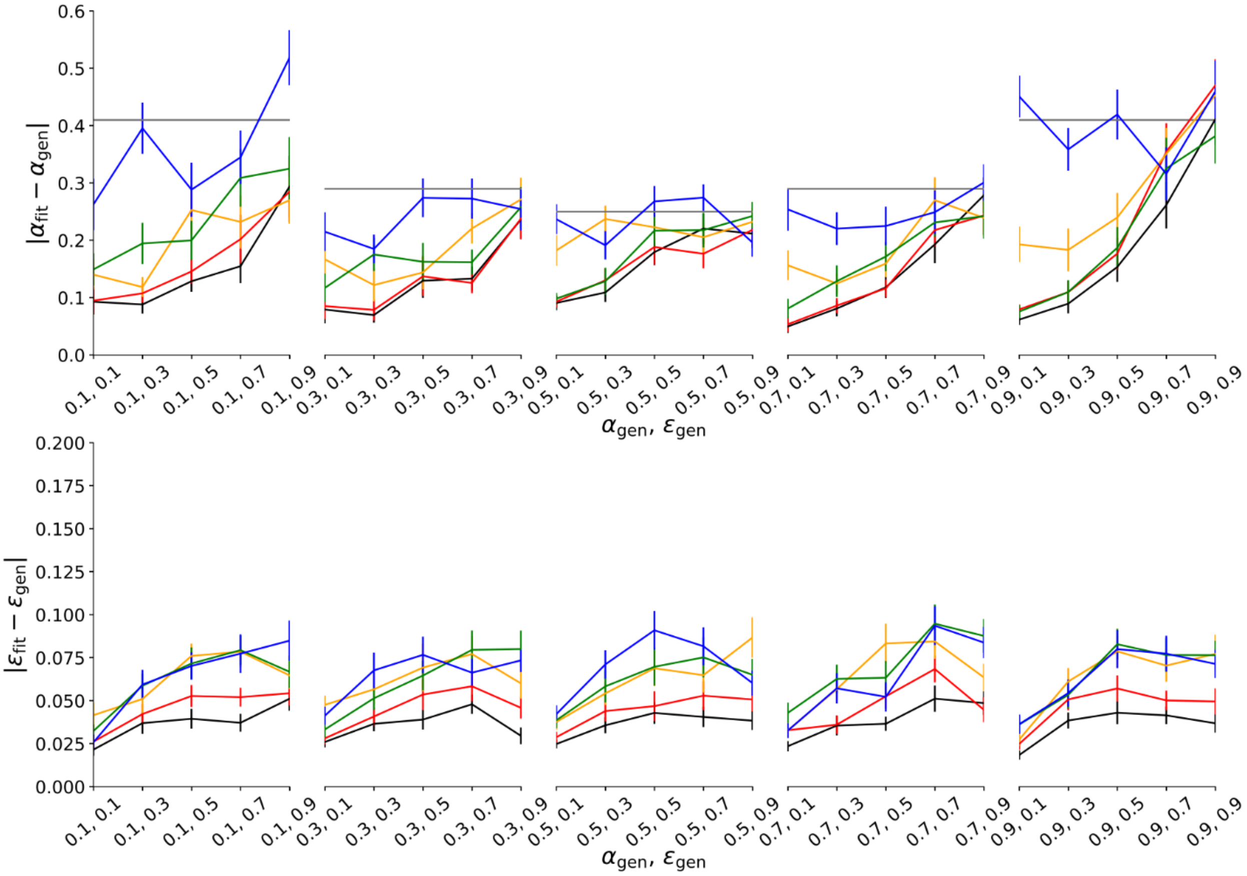
	mean $|\alpha_{\text{fit}} - \alpha_{\text{gen}}|$	mean $|\beta_{\text{fit}} - \beta_{\text{gen}}|$
All	0.18±0.01	1.54±0.10
Feedback	0.20±0.01	1.84±0.11
Final	0.23±0.01	2.81±0.15
First	0.23±0.01	2.98±0.15
None	0.27±0.01	3.22±0.16
randomly recovered	0.33	12.64

Figure 5-5 A plot of the mean absolute difference between the generating and recovered parameter values for the qLearn model performing the Biased coins task and fitted using the action choice probabilities for selected sections of the task, labelled as "All" (black), "Feedback" (red), "First" (green), "Final" (orange) and "None" (blue). Each point is the mean across 30 generated task runs with the same generating parameter values, listed on the horizontal axis. The error bars are the based on the standard error of the mean. The points are ordered by increasing generating parameter values, with β increasing before α. ***Top****: The α parameter values.* ***Bottom****: The β parameter values.*

but the relatively poor performance in fitting with only the no feedback section of the task does suggest that this would be the case.

With the qLearnE model in Figure 5-6, the parameters recovered using None are noticeably worse when fitting α, especially for those generated with low and high values of α. This is especially noticeable when comparing the recovered parameters from Feedback to those from All, which are almost identical, suggesting that there is little value in having a portion of the task with no feedback. However, there is a difference when recovering ε, especially for parameters recovered from data generated with values of ε around 0.5, where the errors tend to be largest.

Comparing the recovered parameters from the two models, the most striking difference is that the errors for α increase as the generating ε values increase with the qLearnE model, when the opposite is true for qLearn with the β parameter. This may be due to the inverse roles ε and β perform in their respective functions. The parameter fit errors are smaller for α when generating and fitting with the qLearnE model, which is in line with what was expected from chapter 4. This difference continues when comparing the errors found with β and ε relative to the sizes of the supports for the two parameters: up to 50% of the support for β and up to 10% for ε.



	mean $|\alpha_{\mathrm{fit}} - \alpha_{\mathrm{gen}}|$	mean $|\varepsilon_{\mathrm{fit}} - \varepsilon_{\mathrm{gen}}|$
All	0.16±0.01	0.04±0.00
Feedback	0.17±0.01	0.05±0.00
Final	0.22±0.01	0.06±0.00
First	0.20±0.01	0.06±0.00
None	0.30±0.01	0.07±0.00
randomly recovered	0.33	0.33

Figure 5-6 A plot of the mean absolute difference between the generating and recovered parameter values for the qLearnE model performing the Biased coins task and fitted using the action choice probabilities for selected sections of the task, labelled as "All" (black), "Feedback" (red), "First" (green), "Final" (orange) and "None" (blue). Each point is the mean across 30 generated task runs with the same generating parameter values, listed on the horizontal axis. The error bars are the based on the standard error of the mean. The points are ordered by increasing generating parameter values, with ε increasing before α. ***Top****: The α parameter values.* ***Bottom****: The ε parameter values.*

5.2 Probabilistic Selection task

This is based on the task as described in Frank et al. (2007). In the first stage of the task, the participant is shown a series of pairs of symbols and asked to pick which of the two will give the reward. They are then told if they are correct. In this version there are three pairs, each with their own unique symbols. The pairs have normalised probabilities of providing a reward: (80%, 20%), (70%, 30%) and (60%, 40%). In the second stage of the task, the participant is again shown pairs of symbols, but the pairs are made up of symbols that were unpaired in the first stage. The participants are not given any feedback as to their performance in this second stage. As for the previous task, new data is generated for each run of the task. No attempt at counterbalancing has been made when generating the data.

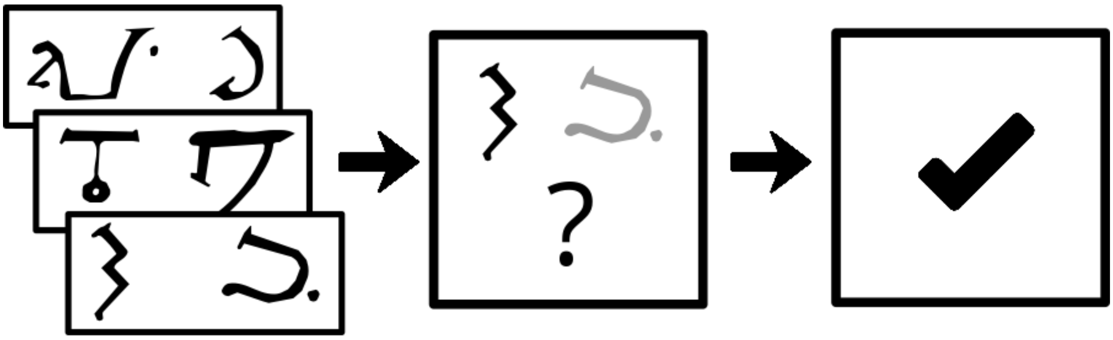
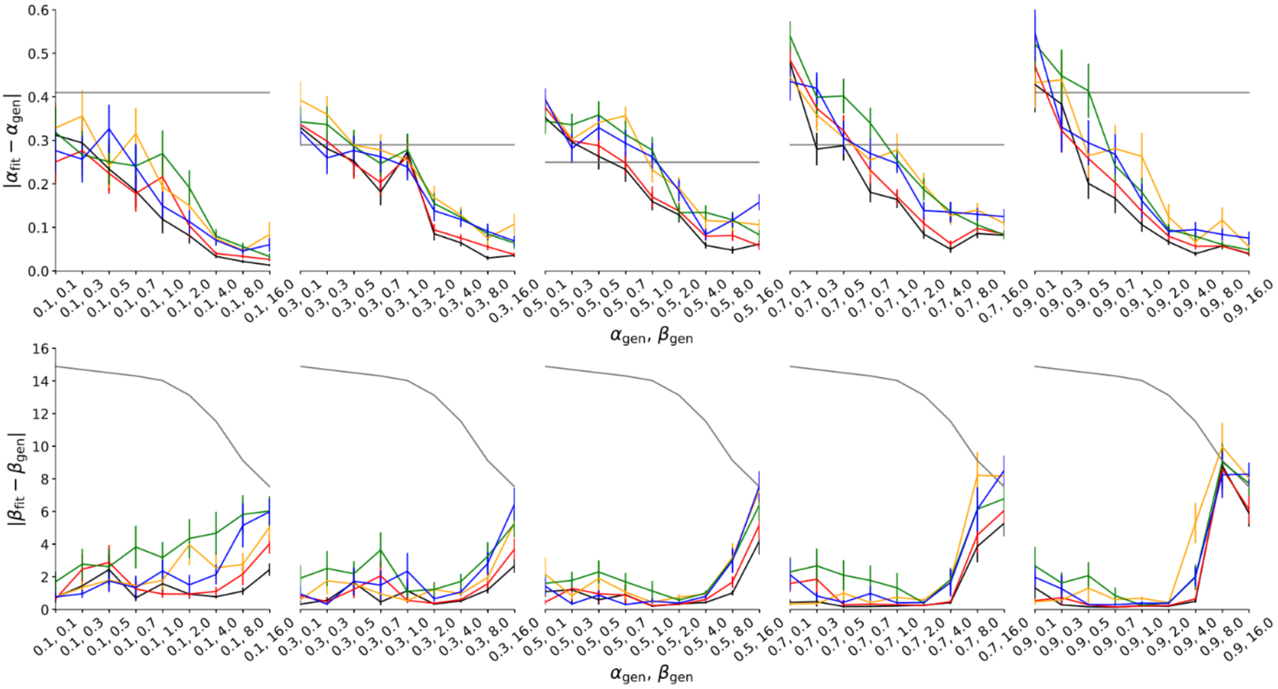


Figure 5-7 The Probabilistic Selection task. Participants are shown pairs of characters, from a set of six, and asked to pick the correct one. Each character has a different likelihood of being correct. During the initial learning phase, the characters are shown three pairs, with complementary reward likelihoods, multiple times and participants are given feedback. In the test phase, participants are presented with every combination of pairs of characters but are not given feedback.

For this task, Figure 5-8 shows that the qLearn model generating parameters are most accurately recovered when fitted over the no-feedback phase of the task. As expected, the fits performed using more of the task, All and Feedback, do recover the parameters better, although there is minimal difference despite the increase in trials for All. This underlines that the improvement in accuracy in recovering parameters over the no feedback region is minimal. For low β values and $\alpha > 0.1$, the α values are recovered worse than if they were randomly selected from a uniform distribution. When compared to the results from fitting the biased coins task in Figure 5-5, the distribution in fitting errors for α and β are similar, with β errors increasing and the α errors decreasing as the generating β increases.

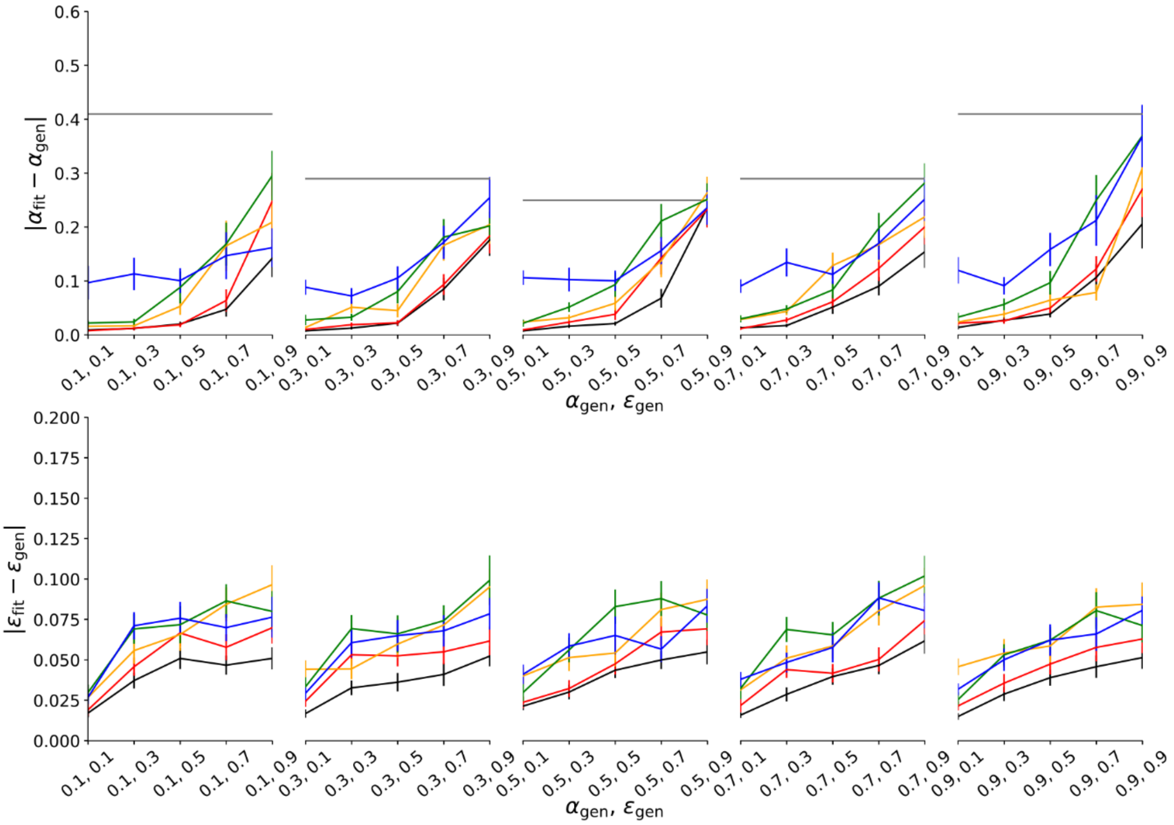
With the qLearnE model, shown in Figure 5-9, most of the difference can be found in the recovery of α, where the errors steadily increase as both α and ε increase, with the exception of when fitting using the None section of the task, where there is less variation in the error size across the parameters and with an average recovery error size higher than almost all of the other error fits. For the ε fits, the largest errors are found in the middle of the ε parameter range. Overall, this gives the impression that for the recovery of α the no feedback region hinders as much



	mean $|\alpha_{\mathrm{fit}} - \alpha_{\mathrm{gen}}|$	mean $|\beta_{\mathrm{fit}} - \beta_{\mathrm{gen}}|$
All	0.17±0.01	1.34±0.09
Feedback	0.18±0.01	1.62±0.09
Final	0.23±0.01	2.32±0.12
First	0.23±0.01	2.88±0.13
None	0.21±0.01	2.22±0.12
randomly recovered	0.33	12.64

Figure 5-8 A plot of the mean absolute difference between the generating and recovered parameter values for the qLearn model performing the Probabilistic Selection task and fitted using the action choice probabilities for selected sections of the task, labelled as "All" (black), "Feedback" (red), "First" (green), "Final" (orange) and "None" (blue). Each point is the mean across 30 generated task runs with the same generating parameter values, listed on the horizontal axis. The error bars are the based on the standard error of the mean. The points are ordered by increasing generating parameter values, with β increasing before α. ***Top****: The α parameter values.* ***Bottom****: The β parameter values.*

as it helps the recovery accuracy, as there is little difference between All and Feedback across all generating parameters despite the 50% increase in trials used to fit the model. This hindrance is not seen in the parameter recovery accuracy for ε, where all three short fitting sections were shown to recover the parameters similarly well.



	mean $|\alpha_{\mathrm{fit}} - \alpha_{\mathrm{gen}}|$	mean $|\varepsilon_{\mathrm{fit}} - \varepsilon_{\mathrm{gen}}|$
All	0.06±0.00	0.04±0.00
Feedback	0.08±0.01	0.05±0.00
Final	0.10±0.01	0.06±0.00
First	0.13±0.01	0.07±0.00
None	0.15±0.01	0.06±0.00
randomly recovered	0.33	0.33

Figure 5-9 A plot of the mean absolute difference between the generating and recovered parameter values for the qLearnE model performing the Probabilistic Selection task and fitted using the action choice probabilities for selected sections of the task, labelled as "All" (black), "Feedback" (red), "First" (green), "Final" (orange) and "None" (blue). Each point is the mean across 30 generated task runs with the same generating parameter values, listed on the horizontal axis. The error bars are the based on the standard error of the mean. The points are ordered by increasing generating parameter values, with ε increasing before α. ***Top****: The α parameter values.* ***Bottom****: The ε parameter values.*

5.3 Weather task

Having seen that there were differences across tasks in the usefulness of fitting parts of a task, it made sense to examine the performance of the Weather task that will be examined in more detail in chapter 8. The Weather task is a category learning task based on one described by Gluck & Bower (1988) and later adapted by Knowlton, Squire, & Gluck (1994). It asks participants to associate a series of cues with one of two outcomes. One to three cue cards, from a set of four cards, are presented to the participant in each trial. The participant must decide which one of the two possible outcomes the displayed cards are most likely associated with. Once the participant decides, they are then told if they were correct or not. The cues each have a probabilistic relationship with the two outcomes, with this this version of the task having novel probabilistic relationship, with the probability of an outcome varying depending on the combination of cues displayed, as described in Table 8-1. For example, if the first two cues are displayed, then the first outcome is guaranteed. If only one of them is displayed, then the first outcome will be the correct one 75% of the time. Across the whole task, the first two cues having a 64% chance of being associated with the first outcome and the second two having the inverse. In the first phase of the task, the *learning phase*, participants are given feedback on if their choice was correct. In the second phase, the *testing phase*, participants are not given any feedback. For this task, the sequence of cues and the feedback were kept the same for all participants, with 200 learning phase trials and 100 test phase trials.

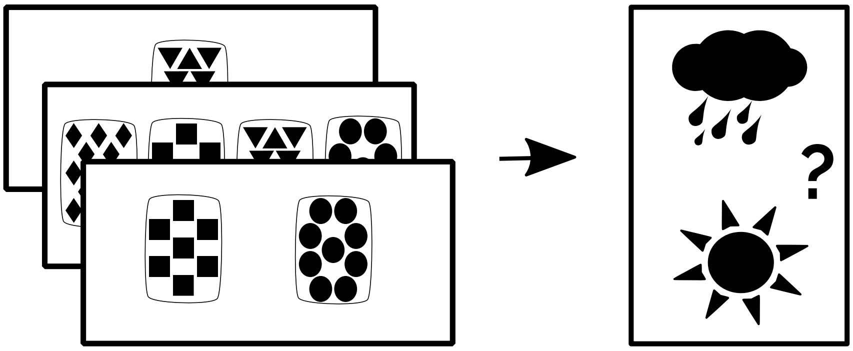
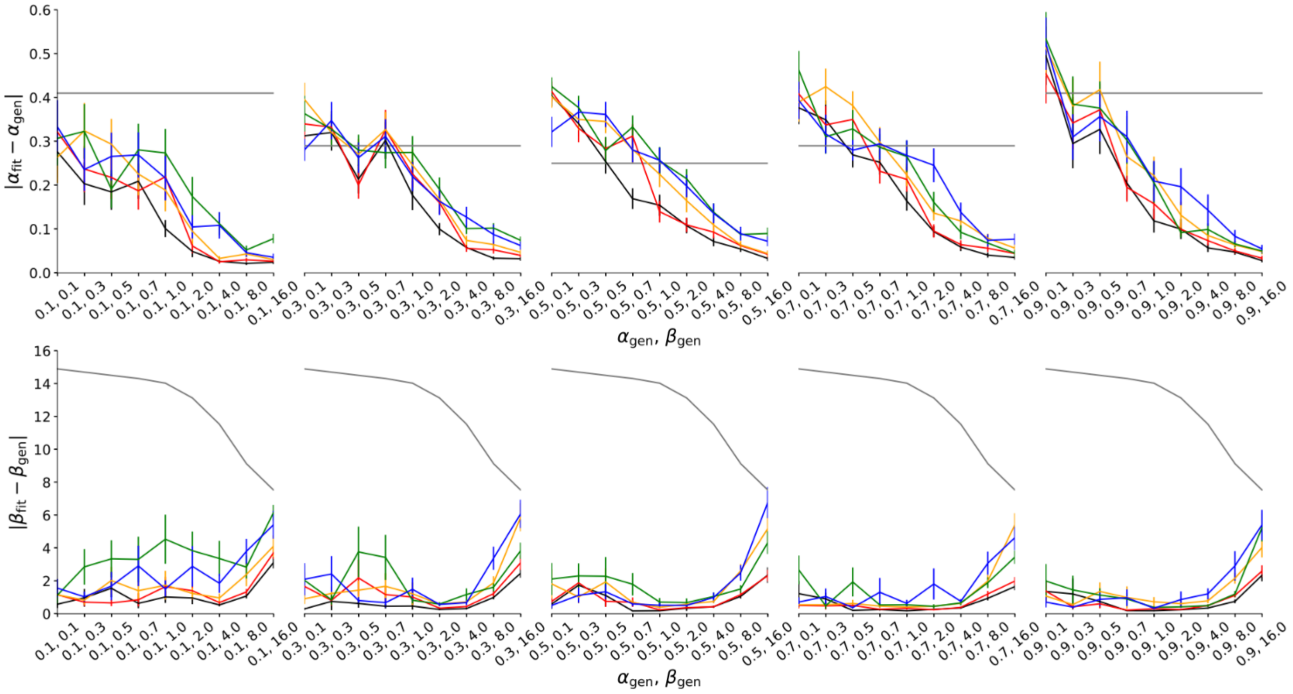


Figure 5-10 The Weather task consists of a series of trials where one to three cue cards, from a set of four cards, are presented to the participant. The participant must decide which of the two outcomes the cues are more likely to predict.

With the qLearn model, shown in Figure 5-11, the fitting errors are similar in distribution to those found for the Probabilistic Selection task. As before, there is minimal difference between the recovery of parameters using all the tasks responses and those using only those from the parts with feedback. For low β values and $\alpha > 0.1$, the α values are recovered worse than if they were randomly selected from a uniform distribution.

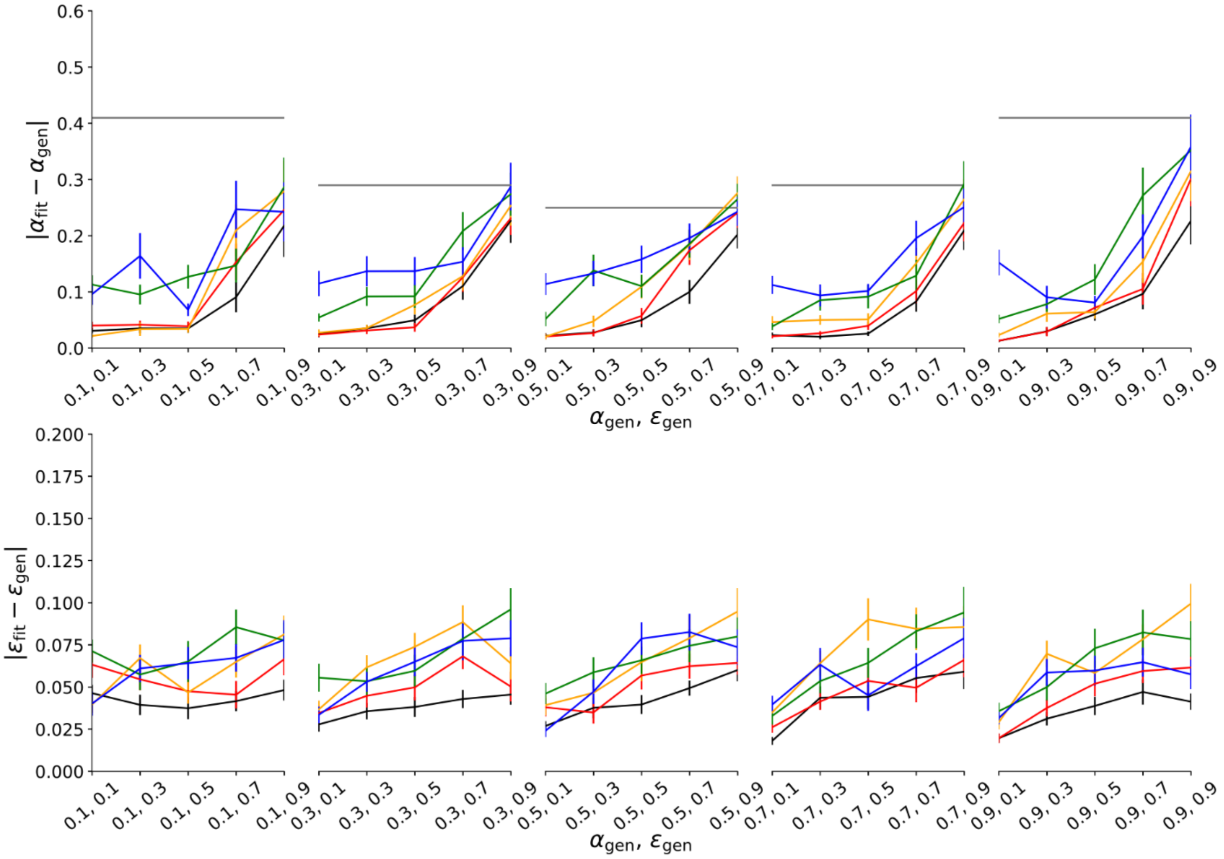
For the qLearnE model, show in Figure 5-12, there is very little difference in the ε recovery errors when fitting using any of the three short sections of the task.



	mean $|\alpha_{\text{fit}} - \alpha_{\text{gen}}|$	mean $|\beta_{\text{fit}} - \beta_{\text{gen}}|$
All	0.17±0.01	0.85±0.05
Feedback	0.19±0.01	1.02±0.06
Final	0.21±0.01	1.55±0.08
First	0.22±0.01	2.05±0.12
None	0.22±0.01	1.86±0.10
randomly recovered	0.33	12.64

*Figure 5-11 A plot of the mean absolute difference between the generating and recovered parameter values for the qLearn model performing the Weather task and fitted using the action choice probabilities for selected sections of the task, labelled as "All" (black), “Feedback” (red), “First” (green), “Final” (orange) and “None” (blue). Each point is the mean across 30 generated task runs with the same generating parameter values, listed on the horizontal axis. The error bars are the based on the standard error of the mean. The points are ordered by increasing generating parameter values, with β increasing before α. **Top**: The α parameter values. **Bottom**: The β parameter values.*

However, variation is seen in the fitting of α, where the parameter recovery using the no feedback portion provides the largest errors and the smallest errors by fitting the post-convergence action feedback portion of the task. From this, it appears that parameter recovery benefits most from maximising the portion of the task with feedback, as this improved the recovery of α.



	mean $|\alpha_{\mathrm{fit}} - \alpha_{\mathrm{gen}}|$	mean $|\varepsilon_{\mathrm{fit}} - \varepsilon_{\mathrm{gen}}|$
All	0.08±0.01	0.04±0.00
Feedback	0.10±0.01	0.05±0.00
Final	0.12±0.01	0.07±0.00
First	0.15±0.01	0.07±0.00
None	0.17±0.01	0.06±0.00
randomly recovered	0.33	0.33

Figure 5-12 A plot of the mean absolute difference between the generating and recovered parameter values for the qLearnE model performing the Probabilistic Selection task and fitted using the action choice probabilities for selected sections of the task, labelled as "All" (black), "Feedback" (red), "First" (green), "Final" (orange) and "None" (blue). Each point is the mean across 30 generated task runs with the same generating parameter values, listed on the horizontal axis. The error bars are the based on the standard error of the mean. The points are ordered by increasing generating parameter values, with ε increasing before α. ***Top***: *The α parameter values.* ***Bottom***: *The ε parameter values.*

5.4 Discussion

This chapter examined how the accuracy in the recovery of parameters varies when fitting using different portions of a task and how this variation changes across a range of tasks. This was done for two models: qLearn and qLearnE.

To provide an indication of the trends found in the results, the overall mean parameter recovery error values have been reproduced in Table 5-1. In this case, the mean errors in α range between 10-30% of the support size compared to between 4-11% for β and ε. This is in line with the results from chapter 4 that the parameters later in the model's trial calculations would be easier to recover. The mean errors for β are the same size as those of ε, but the variation in these errors is much larger for β, which is in line with the result in chapter 4 that ε-greedy allows for more consistent parameter recovery than the softmax function.

The no feedback portion of the task was found to only be helpful for parameter recovery in the Probabilistic Selection task, where the no feedback portion provided different stimulus-cue pairs to those in the feedback portion. This suggests that the noise typically found in a no-feedback testing phase might be reduced by having the testing phase have trials that ask different questions from those in the learning phase. In this case this was done by asking the participant to make the same type of choices, between two options, but in this phase the options were paired differently.

Overall, the parameter recovery errors were found to be lower for the Weather task. One possible reason for this is that it is the only task where the relationship between the action and the feedback was less clear, as there were varying numbers of interacting stimulus-cues. This increased complexity might make it easier to tease out the differences between different parameter combinations on repeated trials.

One limitation in using the models described in chapters 3 and 4 is that they assume that the action choice probabilities cannot change during the no feedback portion of the tasks. However, there are indications that even when people are not provided with feedback for their actions, they may still update their reward

		Biased coins		Probabilistic Selection		Weather	
		qLearn	qLearnE	qLearn	qLearnE	qLearn	qLearnE
First	α	0.23 ± 0.01	0.20 ± 0.01	0.23 ± 0.01	0.20 ± 0.01	0.22 ± 0.01	0.15 ± 0.01
	β/ε	2.98 ± 0.15	0.06 ± 0.00	2.88 ± 0.13	0.10 ± 0.00	2.05 ± 0.10	0.07 ± 0.00
Final	α	0.23 ± 0.01	0.22 ± 0.01	0.23 ± 0.01	0.20 ± 0.01	0.21 ± 0.01	0.12 ± 0.01
	β/ε	2.81 ± 0.15	0.06 ± 0.00	2.32 ± 0.12	0.11 ± 0.00	1.55 ± 0.08	0.07 ± 0.00
None	α	0.27 ± 0.01	0.30 ± 0.01	0.21 ± 0.01	0.30 ± 0.01	0.22 ± 0.01	0.17 ± 0.01
	β/ε	3.22 ± 0.16	0.07 ± 0.00	2.22 ± 0.12	0.11 ± 0.00	1.86 ± 0.10	0.06 ± 0.00
Feedback	α	0.20 ± 0.01	0.17 ± 0.01	0.18 ± 0.01	0.19 ± 0.01	0.19 ± 0.01	0.10 ± 0.01
	β/ε	1.84 ± 0.11	0.05 ± 0.00	1.62 ± 0.09	0.08 ± 0.00	1.02 ± 0.06	0.05 ± 0.00
All	α	0.18 ± 0.01	0.16 ± 0.01	0.17 ± 0.01	0.17 ± 0.01	0.17 ± 0.01	0.08 ± 0.01
	β/ε	1.54 ± 0.10	0.04 ± 0.00	1.34 ± 0.09	0.06 ± 0.00	0.85 ± 0.05	0.04 ± 0.00

Table 5-1 The means and standard error of the means for the absolute parameter recovery errors across all generating parameter values. These are shown for each trialstep fitting region, parameter, task and model. The colours signify the relative size of the errors from low (dark blue) through to very high (dark red). The relative sizes are evaluated across trialstep fitting regions for the same parameter, same task and same model, i.e. each column and each parameter are treated separately.

expectations, reinforcing those that they have chosen and discounting those they have chosen not to choose (Lieberman, Ochsner, Gilbert, & Schacter, 2001).

It should be reiterated that this chapter is only discussing parameter recovery, not the appropriateness of models. As discussed at the beginning of the chapter, it is plausible that there are multiple learning models or policies being used simultaneously, each of which dominate under different circumstances, such as when certain kinds of feedback, such as corrective or rewarding, are provided or withheld (Frank et al., 2007). If the assumption when recovering model parameters is that a participant is using the same model for the feedback and the no feedback parts of the task, then there is no benefit, for parameter recovery accuracy, to having a no feedback part of the task. However, if there is a possibility that participants may be using a different model, or different model parameters, for when there is and is not feedback, then these should be fitted separately, with the awareness that this will impact the accuracy of the parameters recovered.

6 DECKS TASK

The first task used for evaluating model performance was the Decks task, a modified version of the one used by Worthy, Maddox, & Markman (2007), and similar to the IOWA gambling task (Bechara et al., 1994). Participants were presented each trial with two stimuli on a screen, one red and one blue. These were said to be the top 'cards' of two decks of cards 80 cards long. Each 'card' had a predetermined reward associated with it, whose value was between one and ten. The objective was to maximise the accumulated card values, with the chance to enter a lottery for a prize, described below, if the participant collected more than 450 points across the experiment. For each pair of cards, the participants chose one. The card that was not chosen was not discarded, maintaining the number of available cards in each deck. This version of the task was therefore what Worthy et al. refer to as a gain only version of their task under the promotion focussed motivational framing. A promotion-focus serves to motivate participants to win points by providing a potential higher payoff if enough points were accumulated over the course of the experiment.

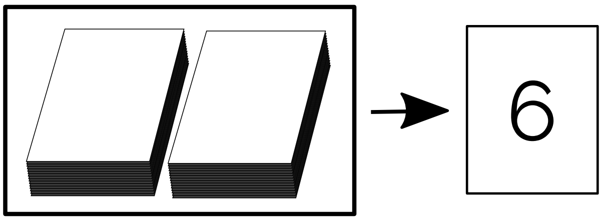


Figure 6-1 The Decks task consists of two decks of 80 cards. Each card has a value between one and ten. Participants choose during each trialstep which deck they thing will provide the most advantageous card, with an aim to accumulate the largest total card value. When a deck is chosen, the 'top' card from that deck is drawn, its associated reward is awarded to the participant and the card is discarded.

Throughout the experiment, a fixed card value sequence was kept for both decks, shown in Figure 6-2, and there were not equal numbers of cards for each reward value. One of the decks was initially advantageous, but overall worse. It provided an average of eight points over the first thirty cards drawn from that deck, five points for the following twenty and two for the final thirty cards. The other deck

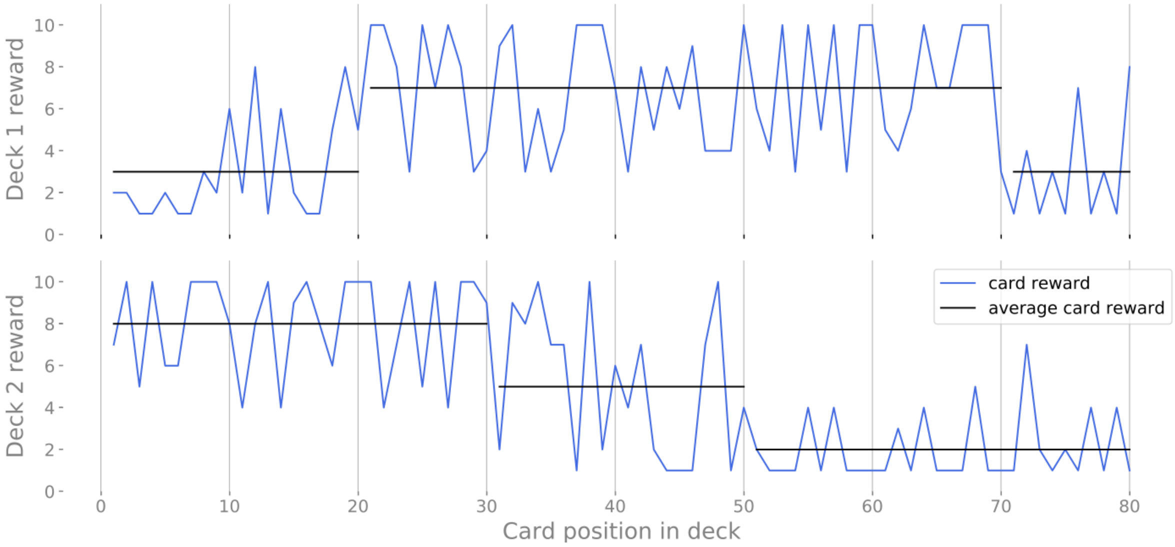


Figure 6-2 The rewards received for choosing each card in the two decks. The black lines show the average rewards for each deck in each of their payoff "phases".

became steadily better and then reverted to providing low payouts just at the end, with an average of three points for the first twenty, an average of seven over the following fifty cards and an average of three for the final ten cards.

Since the deck that was initially advantageous became less advantageous later on, this meant that participants who wanted to reach 450 points would have to select at least 25 cards from the initially poorer deck and at least 3 cards from the initially

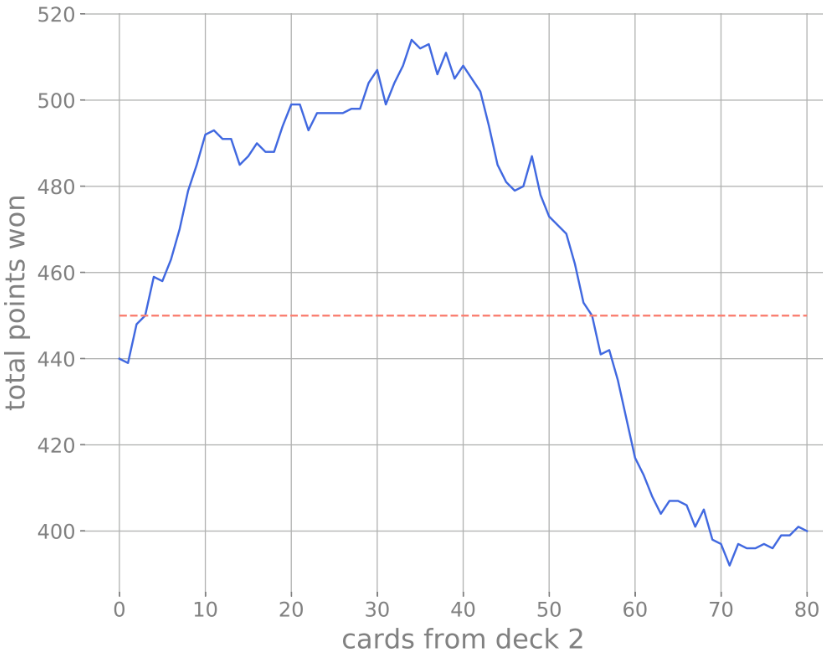


Figure 6-3 The total number of points won as more cards are chosen from deck 2 across the 80 trials of the task. The points for each card are those shown in Figure 6-2. 450 points were necessary to qualify for the bonus reward, as shown by the pink dotted line.

better deck to be able to reach this total, as shown in Figure 6-3. The task therefore required the participant to explore, or sample, both decks even when a decision on the better deck was made, as the best deck to choose changed as more cards were chosen. Those participants who initially favoured deck 2 are given two encouragements to explore deck 1: the reduction in average reward after 30 cards and a second reduction after 50 cards. The second of these reductions results in an average reward that is lower than that of deck 1 at any point.

6.1 Data properties

The results from three different undergraduate student research projects were available for analysis, each with different sets of participants, and all run with the same version of the task. Studied 1 and 2 were conducted at Goldsmiths and study 3 was conducted at the University of Greenwich. The lottery prize for the two Goldsmiths studies was £50 and at Greenwich the prize was £20. A detailed description of these studies can be found in Pickering (2011). In total, there were 166 participants.

Checks were performed on the sample characteristics to verify the suitability of combining the three datasets together. An overview of the participants for the three studies can be seen in Table 6-1. We can see that all three studies had about the same number of participants. The age range was higher for the third study, where participants 75% of participants were under the age of 33, whereas for the other studies 75% of participants were under the age of 23 and 24 respectively. The gender distribution of all three studies, shown in Table 6-2, is very similar across studies, with 70% of participants being female.

	Study	*Total number*	*Mean*	*Standard deviation*	*Distribution*
Age	Combined	166	25	10	
	1	54	23	8	
	2	55	24	8	
	3	57	29	11	

Table 6-1 A description of the age of the participants in the three Decks task studies.

	Study	*Total number*	*Percentage of men*	*Percentage of women*
Gender	Combined	166	29	71
	1	54	30	70
	2	55	27	73
	3	57	28	72

Table 6-2 An overview of the genders of the participants in the three Decks task studies.

In each study the EPQ-R was given to each participant. The focus for this chapter will be on the Extraversion scale, as this personality trait has been linked to variations in the sensitivity to reward, discussed in chapter 1.1. The results in Table 6-3 show the results of the EPQ-R tests that were administered. They show similar results across the studies. The results from tables Table 6-1, Table 6-2 and Table 6-3, suggest that the data from these experiments can be combined for more power, as the underlying samples are similar.

	Study	*Total number*	*Mean*	*Standard deviation*	*Distribution*
EPQ-R Extraversion	Combined	166	15.5	4.9	
	1	54	16.2	4.9	
	2	55	14.9	4.6	
	3	57	15.3	5.3	
EPQ-R Psychoticism	Combined	166	7.8	3.8	
	1	54	8.2	3.9	
	2	55	7.2	3.5	
	3	57	7.8	3.9	
EPQ-R Neuroticism	Combined	166	12.9	5.5	
	1	54	12.8	5.5	
	2	55	12.8	5.5	
	3	57	13.0	5.5	
EPQ-R Li scale	Combined	166	7.8	3.8	
	1	54	7.5	3.7	
	2	55	7.3	3.4	
	3	57	8.6	4.0	

Table 6-3 The EPQ-R participant properties for the three Decks tasks studies.

The participants' performance was generally very similar across the three studies, as seen in Table 6-4, with participants having similar distributions of card picks from the better deck. The number of points won also had a consistent bimodal distribution, with peaks around 400 and 470 points. This reinforces the idea that the data from the three studies can be combined to provide a dataset with greater power.

	Study	*Total number*	*Mean*	*Standard deviation*	*Distribution*
Number of cards from good deck	Combined	166	25	11	
	1	54	27	11	
	2	55	25	8	
	3	57	23	12	
Points won	Combined	166	446	36	
	1	54	450	34	
	2	55	447	36	
	3	57	440	38	

Table 6-4 An overview of the performance of the participants across the three Decks tasks studies. The target number of points for the participants was 450, which required a minimum of 25 cards selected from the better deck and no more than 78 to accumulate a point total of more than 450.

Based on these results, the data was considered acceptable to be analysed as one dataset. As the three datasets produced very similar results, it seems likely that these are typical samples of British psychology undergraduates who signed up for research studies for course credit.

As extraversion will be compared later to model parameters, a comparison was made between each participant's measured extraversion and the points they won during this task. A Pearson correlation found a -0.04 correlation (p=0.64, BF_{10}=0.11), showing that extraversion was unlikely to be an indicator of the number of points won.

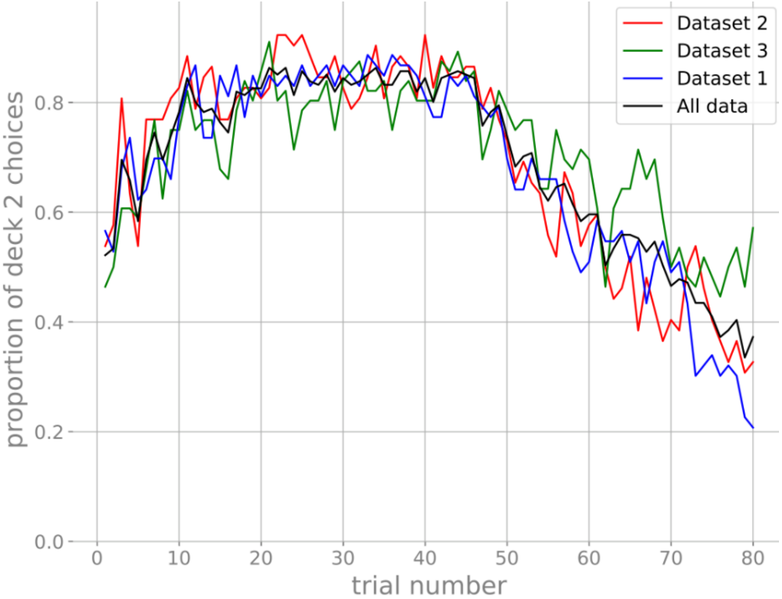


Figure 6-4 The proportion of deck 2 choices for each dataset for each trialstep.

There was a tendency for participants to select more cards from deck 2 at the start of the task, once a difference had been identified between the two decks, and select more from deck 1 as the task continued, as shown in Figure 6-4. This tendency can also be seen to be similar across the three datasets. A Bayesian paired samples t-test was used to compare the means of the first and last 40 trials, as the differences between them can be seen, in Table 6-5, to be significantly non-normal, using the Shapiro-Wilk test of normality. The t-test found very strong evidence that there was a difference in the average action choices between the first and second 40 trials, shown in Figure 6-5. This suggests that the task did perform as designed: participants identified the need to switch from mostly selecting from deck 2 at the start of the task to mostly selecting from deck 1 by the end. This difference in the average action choices between the first and second 40 trials did not correlate with extraversion r= -0.09, p=0.27, BF_{10}=0.18.

Study	*Shapiro-Wilk*		*Bayesian Paired samples t-test*
	W	*p*	*Bayes factor*
Combined	0.98	0.02	3.8×10^{24}
1	0.94	0.01	3.8×10^{12}
2	0.96	0.07	2.8×10^{9}
3	0.96	0.09	8.6×10^{2}

Table 6-5 The results of the Shapiro-Wilk test of normality and the Bayesian paired samples t-test for each of the datasets, along with the combined dataset.

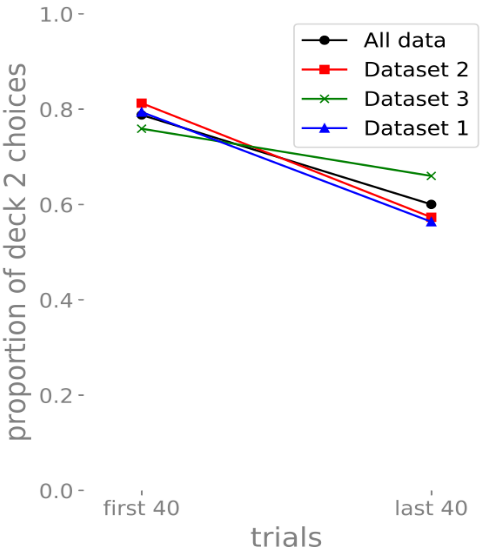


Figure 6-5 The proportion of deck 2 choices for each dataset the mean values across the first and second 40 trials. The standard errors of the means were comparable or smaller to the symbol sizes, so have been omitted.

All but two of the participants selected cards from both decks. These two participants were excluded from the rest of the analysis.

One final way of evaluating the performance of the participants is to see their state at key payoff transition points in the task, marked by the change in average payoff for selecting cards from that deck. The two that most participants can be expected to have gone through are the first decrease in Deck 2 payoffs after 30 card selections from that deck, and the significant increase in Deck 1 average card payoffs after 20 Deck 1 card selections. Figure 6-6 shows the number of cards had been taken from the other deck before reaching these transition points, and so the number of cards remaining to be chosen. The 20th card from Deck 1 can be seen to have been chosen quite late in the task by many of the participants, with 48 others never selecting this card at all. A total of 75 participants had chosen at least 74 cards before reaching the 20th Deck 1 card and so were continuing to choose Deck 2 cards when their average payoff was less than those of Deck 1. Furthermore, those participants would have had very little opportunity to identify the increase in average payoffs that occurs in Deck 1 after the 20th card. The 30th card in Deck 2 was chosen quite quickly by most of the participants, suggesting that all but a few identified Deck 2 as providing the highest average rewards with a modal number of cards selected from deck 1 of 4. This, along with the results from Figure 6-4,

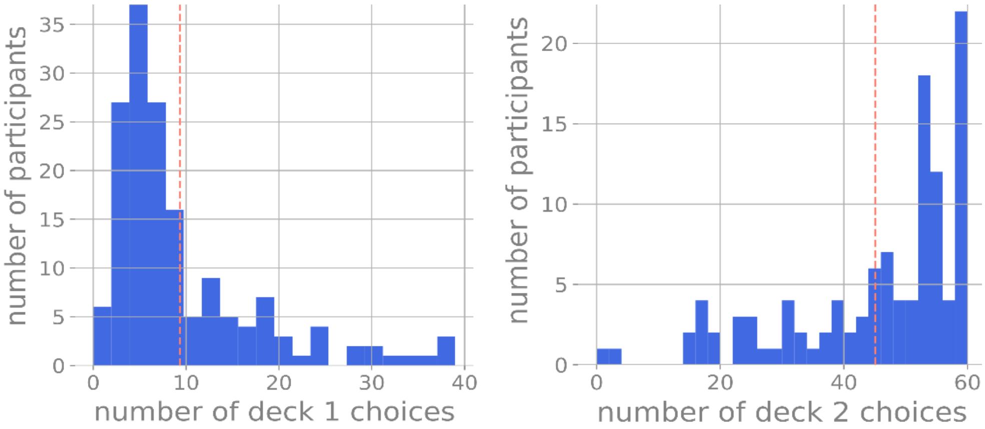


*Figure 6-6 The graph shows the number of choices each participant had taken from a given deck, before reaching the first payoff transition point in the other deck, as shown in Figure 6-2, namely **left:** 30 cards from Deck 2 **right:** 20 cards from Deck 1. This can also be thought of as "Upon having chosen 30 cards from Deck 2 (left) or 20 cards from Deck 1 (right) how many cards had each participant taken from the other deck?" The red vertical dashed lines mark the mean number of cards chosen.*

suggests that Deck 2 was quickly identified as the deck providing the best initial rewards, but as these rewards decreased cards were more frequently sampled from Deck 1. The task therefore performed as expected.

6.2 Fitting the models to the data

As the rewards in this task had values in the range [1, 10], within the models, the initial expected reward for both choices were set to 5. The upper bounds for infinitely bounded model parameters were truncated. The softmax β parameter was limited to 30, the Kalman model parameters σ_{α}^{2} and σ_{λ}^{2} were limited to 150 and the OpALS and OpALSE saturation parameters M were set to 20 and 49 respectively. Participants were fitted over their full task action sequence.

TD0, qLearnF, ACBasic, and the OpAL models have features that caused numerical overflows for rewards larger than 1. For these models, the rewards were scaled to the range [0.1,1] and the initial expected reward was accordingly modified to 0.5. In spite of this, during the fitting process, the model OpAL experienced numerical overflows for certain parameter combinations, as discussed in chapter 3.4. A model choosing randomly would have a probability of 0.5 for each action choice. The parameter combinations where there has been an overflow are treated as worse than random fits. For these, a probability of 0.4 was returned for each action choice. If these provided a plateau of global minima, then the model could be discarded as being a worse representation of the participant’s actions than a purely random model.

6.2.1 Boundary recovered parameters

The crudest measure of acceptable model fits is to measure the number of them that fail to recover valid parameters, i.e. a parameter combination that does not intersect with any of the parameter validity boundaries. To account for numerical errors in the fitting, a boundary, or edge, fit is defined as a recovered if the parameter is within 0.1% of either side of its range. This was chosen to be close enough to the boundaries to be unlikely to interfere with good parameter recovery, while still accounting for the approximate nature of numerical fitting.

The results from fitting the Decks dataset, in Figure 6-7, show the proportion of participants with recovered parameters on the edge of the parameter boundaries. The proportions are generally below 15%, with the ε-greedy models having a significantly lower proportion of boundary parameter values of around or below 5%. This suggests that ε-greedy does provide better parameter recover than SoftMax, as was discussed in chapter 4. In spite of the very broad parameter ranges places on the qLearnK model, still 35% of the participant's sets of recovered parameters included at least one boundary parameter value, suggesting that this model is very poorly fitted.

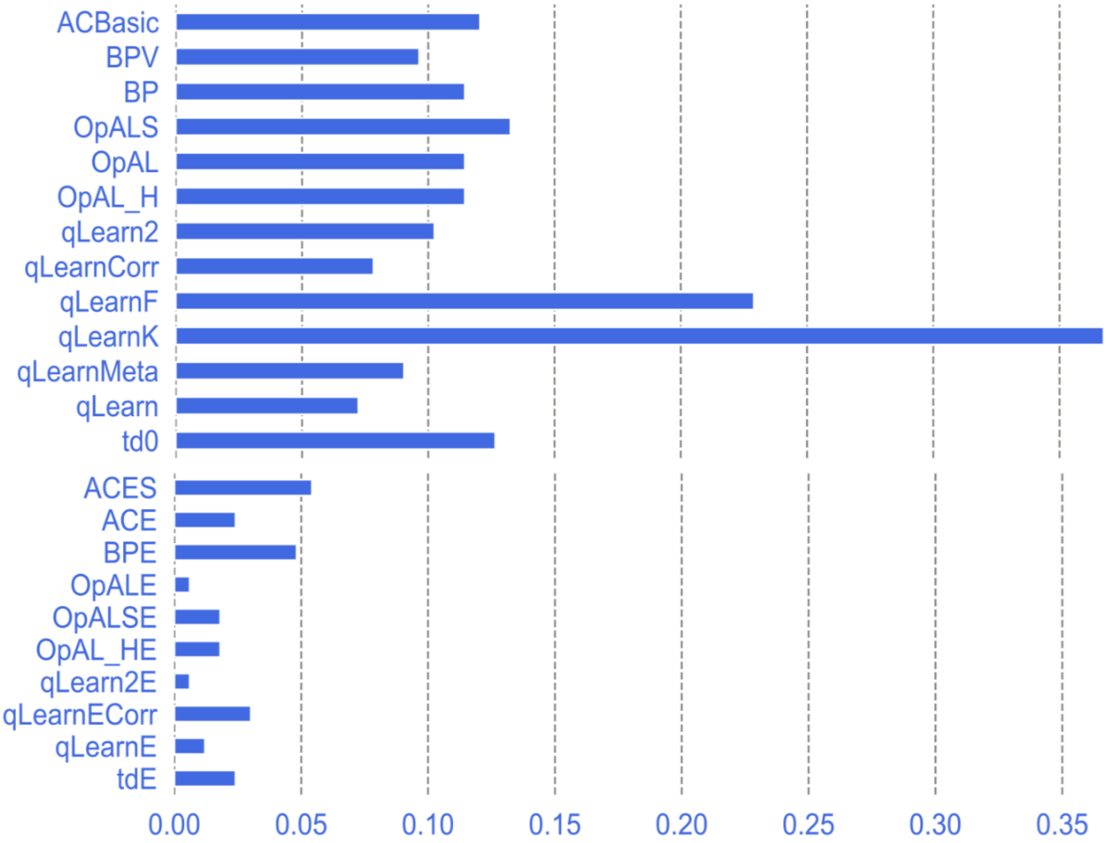


*Figure 6-7 The proportion of the 166 participants fitted to each model whose fits had at least one recovered parameter within 0.1% of its boundary. The models have been grouped into those using softmax (**top**) and those using epsilon-greedy (**bottom**).*

6.2.2 Goodness of model fits

A model’s performance can be compared using a Bayes factor, as described in chapter 2.2, by comparing its performance to that of another model, such as a random choice model. The Decks task encourages participants to learn to prefer choosing one deck for the first half of the task and then switch preference. The randomBias model can be used to capture any participant’s tendency to prefer one deck more than another, while not accounting for any switch in preference they might perform. It therefore acts in this instance as a stricter, non-learning, baseline than the pure random model. To simplify the comparison with other task datasets in chapters 7 and 8, the initial performance comparisons will use the random choice model.

For the Decks task dataset, the performance can be seen, for the Bayes factor, in Figure 6-8, and for the normalised Bayes factor, in Figure 6-9. What is most striking is how models of the same class (Q-learning, Bayesian, OpAL) appear to perform, on average, similarly to each other. The Bayes-inspired models appear to perform

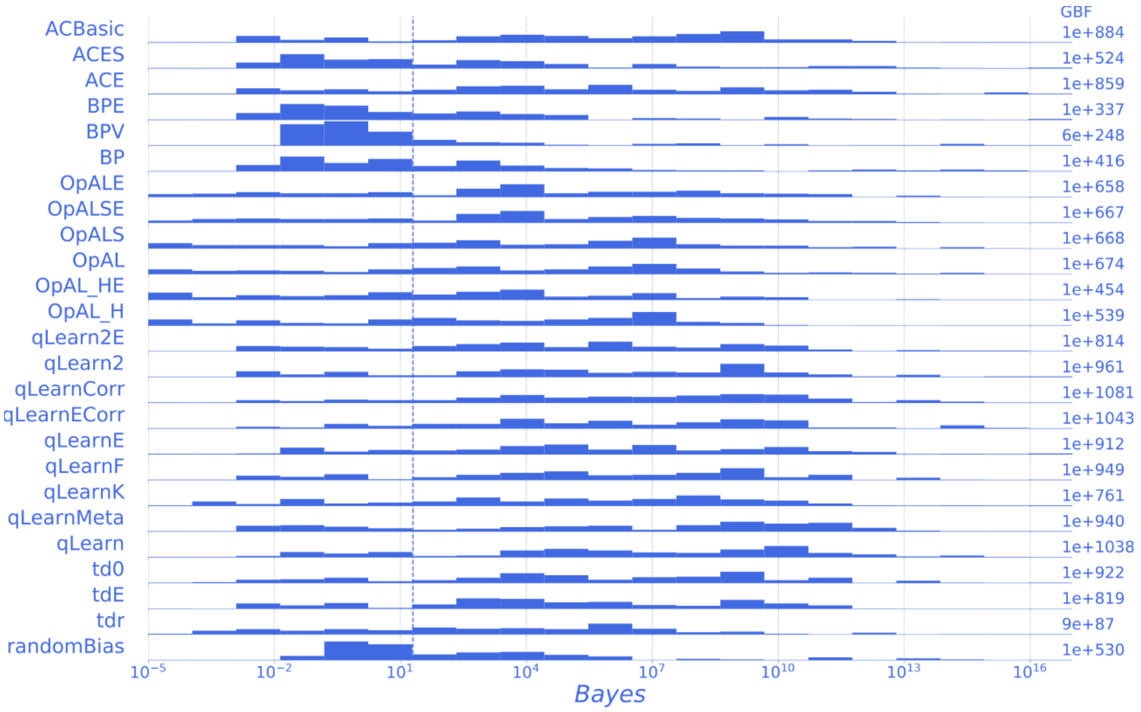


Figure 6-8 The distribution of the values of the fit quality Bayes factor from fitting the Decks dataset when compared to a pure random model. The dashed vertical line marks a Bayes factor of 20. Values above 20 have strong evidence that the model can match the participant’s actions better than a pure random model.

the worst, with the majority of participant fits not having strong evidence, as defined in chapter 2.2, that they are better than a purely random model. The OpAL derived models also frequently provide poor fits. Better fits are recovered for the variations on the qLearn model, with the majority of participants fits having strong evidence of being better compared to a purely random model.

The Bayes factor values for each participant's model fits can be compared between models, as shown in Figure 6-10. In this figure, identical Bayes factors for the same participant will be located along the diagonal line in each miniplot. This parity between Bayes factors is the case for almost all participants when comparing, for example, the OpAL and the OpALS models, also shown expanded in Figure 6-11. If one model fits a participant better than another model, then the participant will be shown away from the diagonal line. For example, the qLearn model fits have

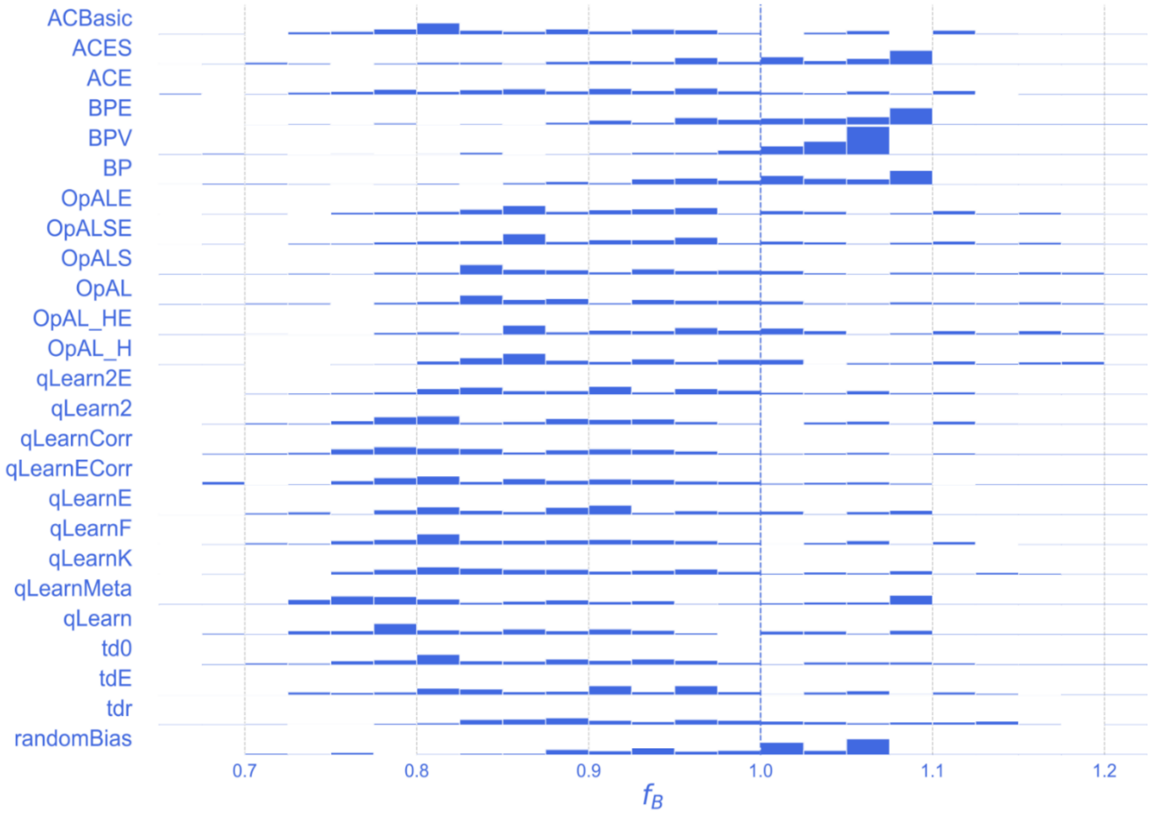


Figure 6-9 The distribution of the values of the fit quality normalised Bayes factor from fitting the Decks dataset when compared to a pure random model. Fits with values below 1 have a Bayes factor of over 20, so have strong evidence that the model can match the participant's actions better than a pure random model.

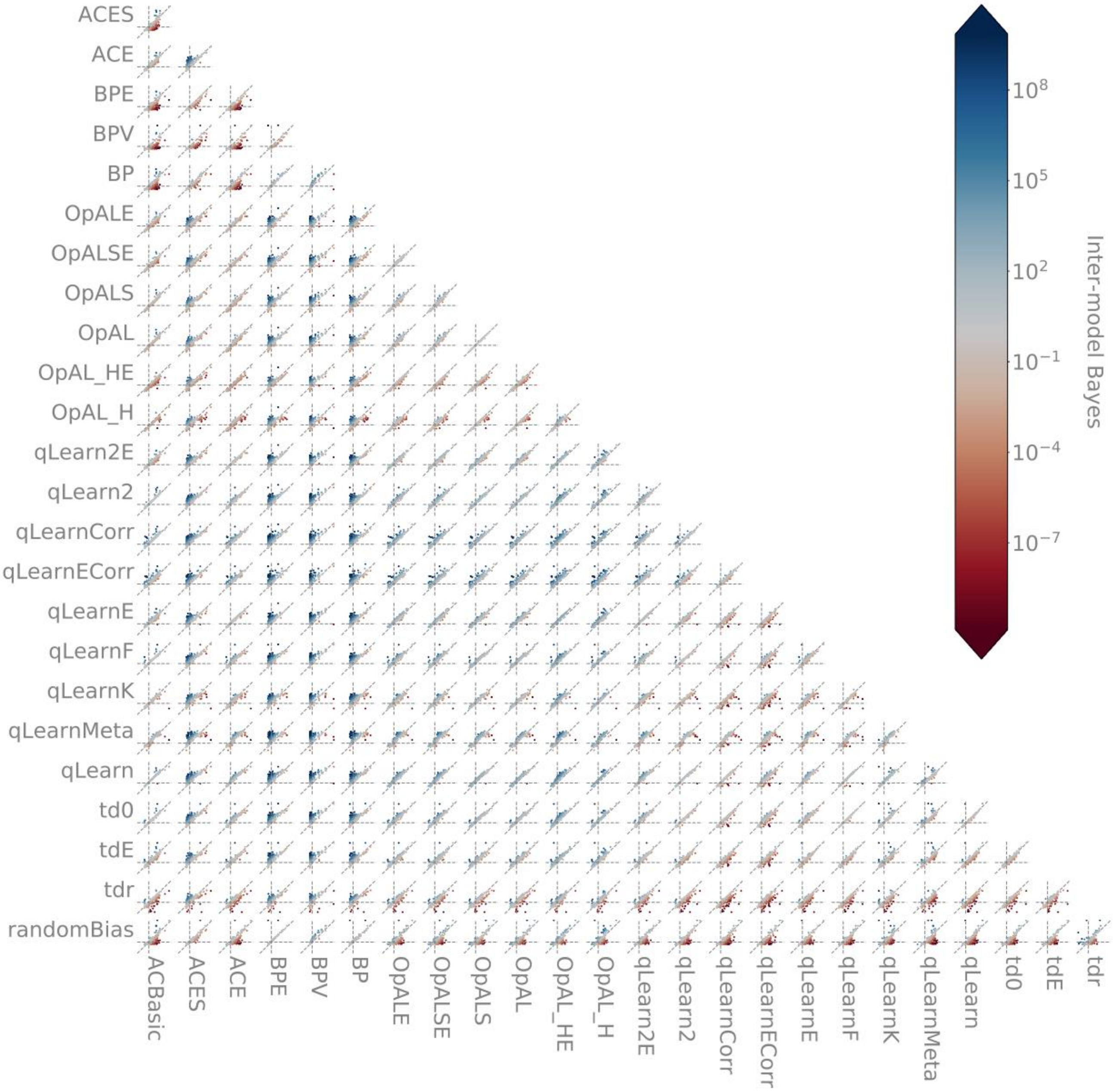


Figure 6-10 A comparison between models of the Bayes factor values for each participant when compared to a pure random model. Both the horizontal and vertical axes of each model comparison use a log scale ranging from 10^{-10} to 10^{23}. The horizontal and vertical lines denote a Bayes factor of 20 and the diagonal line follows the line of equal value for both axes. The dots are coloured with the inter-model Bayes factor, such that a Bayes factor of 20 signifies that there is strong evidence that the vertical model fits better the participant's actions than the horizontal model.

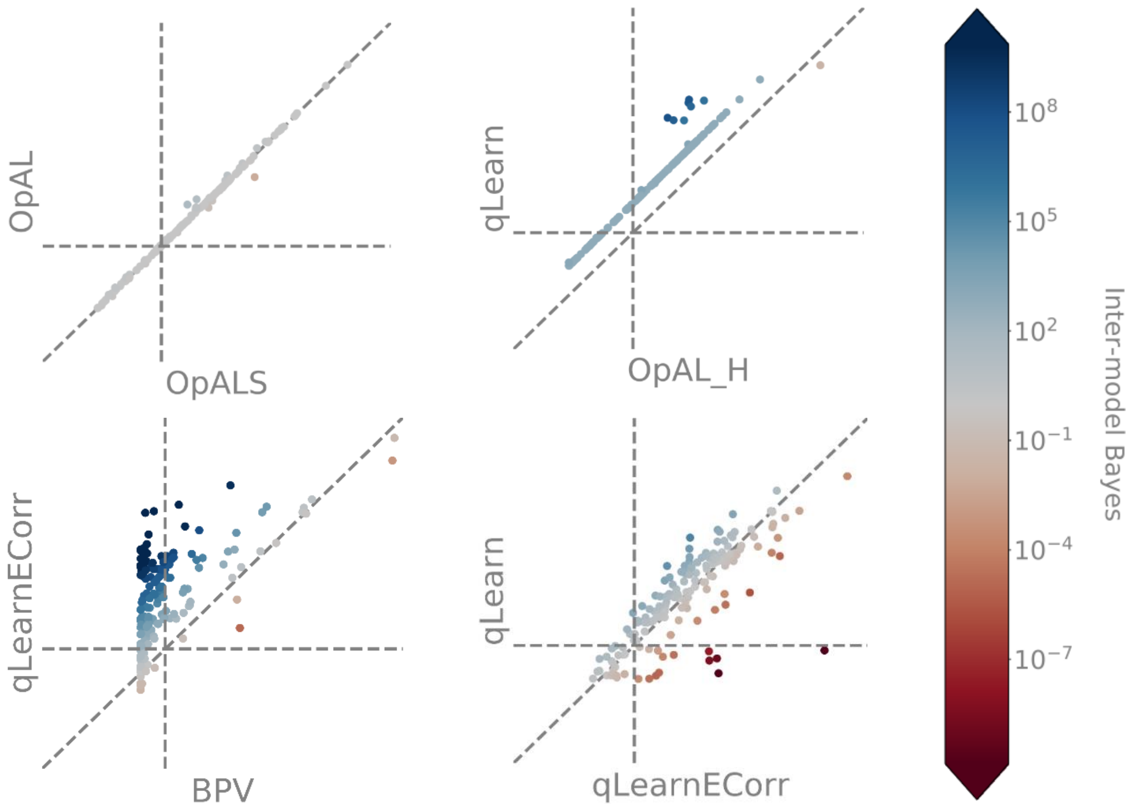


Figure 6-11 A selection of expanded miniplots from Figure 6-10.

stronger evidence than those of OpAL_H, as they are on the qLearn side of the diagonal line. Nevertheless, they are still related, as most of the dots form a line that is parallel to the diagonal line of equal Bayes factor. The further away a dot is from the diagonal line, the greater the difference in the Bayes factors of the two models for that participant data. The horizontal and vertical lines mark a Bayes factor of 20 for the fit of the model axis they intersect with, i.e. the vertical axis model fit values are associated with the horizontal dashed line. Therefore, for most participants, the Bayes factor values for the BPV model relative to a pure random model are lower than 20, whereas most participants for the qLearnECorr model have values higher than 20. This results in the dots in their comparative plot not only to be mostly on the qLearnECorr side of the diagonal line, but also beyond the vertical dotted line.

The relative difference between model fits be expressed as a between-model Bayes value, calculated using the model fit BIC values by taking inspiration from equations 2.4 and 2.8 in chapter 2.2, so that:

$$\mathcal{B} = 2^{\frac{\mathrm{BIC}_{model\,1} - \mathrm{BIC}_{model\,2}}{2}}$$

These are shown in Figure 6-10 by the colour of each participant's dot, with the vertical model as model 2 and the horizontal model as model 1. Therefore, the participant dots with a Bayes factor of 20 or higher, coloured blue, have strong evidence that the vertical model is a better fit for the participant's actions than the horizontal model. Conversely, with a Bayes factor of 1/20 or lower, coloured red, there is strong evidence that the horizontal model is a better fit for the participant's actions than the vertical model. The median values for each inter-model comparison are shown in Figure 6-12. From both figures, it can be seen, for example, that qLearnECorr fits the participants better than BPV. There are occasions where Figure 6-12 can be misleading, such as when comparing qLearn and qLearnECorr, where Figure 6-12 suggests there is somewhat strong evidence

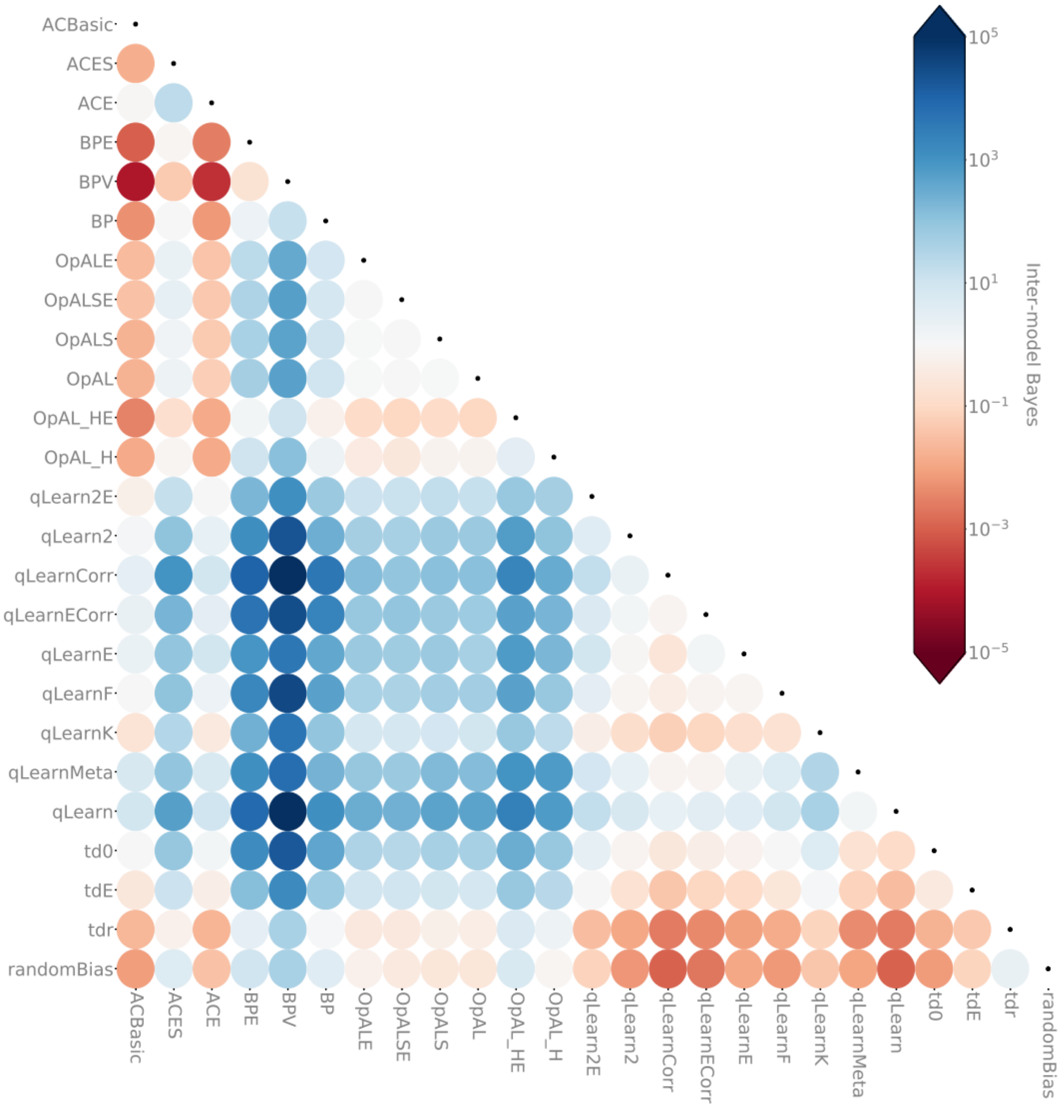


Figure 6-12 The median inter-model Bayes values for the Decks dataset participant fits. High values signify that the model on the vertical axis had a lower BIC value, and so a better match of the participant's actions, than the horizontal axis model.

that qLearn fits the participants better, whereas Figure 6-10 shows that there is strong evidence that qLearn and qLearnECorr each fit different participants.

One striking result is how strong the correlations are between different model Bayes factors for the same participants, as shown by the distributions forming lines in Figure 6-10 and Figure 6-11. This suggests that the actions of some participants are better fitted by these models than the actions of others.

Another approach to comparing the model's performance is to examine the expectation of the model frequencies, EF, as discussed in chapter 2.4. This assess the relative frequencies with which two models could have generated participant data in the dataset. The probability that this relative frequency is above chance is estimated using the protected exceedance probability (Rigoux, Stephan, Friston, & Daunizeau, 2014). Both the expectation of the model frequencies and the related protected exceedance probability were calculated using the VBA toolbox (Daunizeau et al., 2014). This used as inputs the BIC values calculated for each model's fit to each participant's task action sequence. These comparisons can be seen in Figure 6-13. This reinforces our previous conclusion that the BP models were unlikely to have generated the participant data and that the qLearn model has the strongest evidence, followed by qLearnCorr.

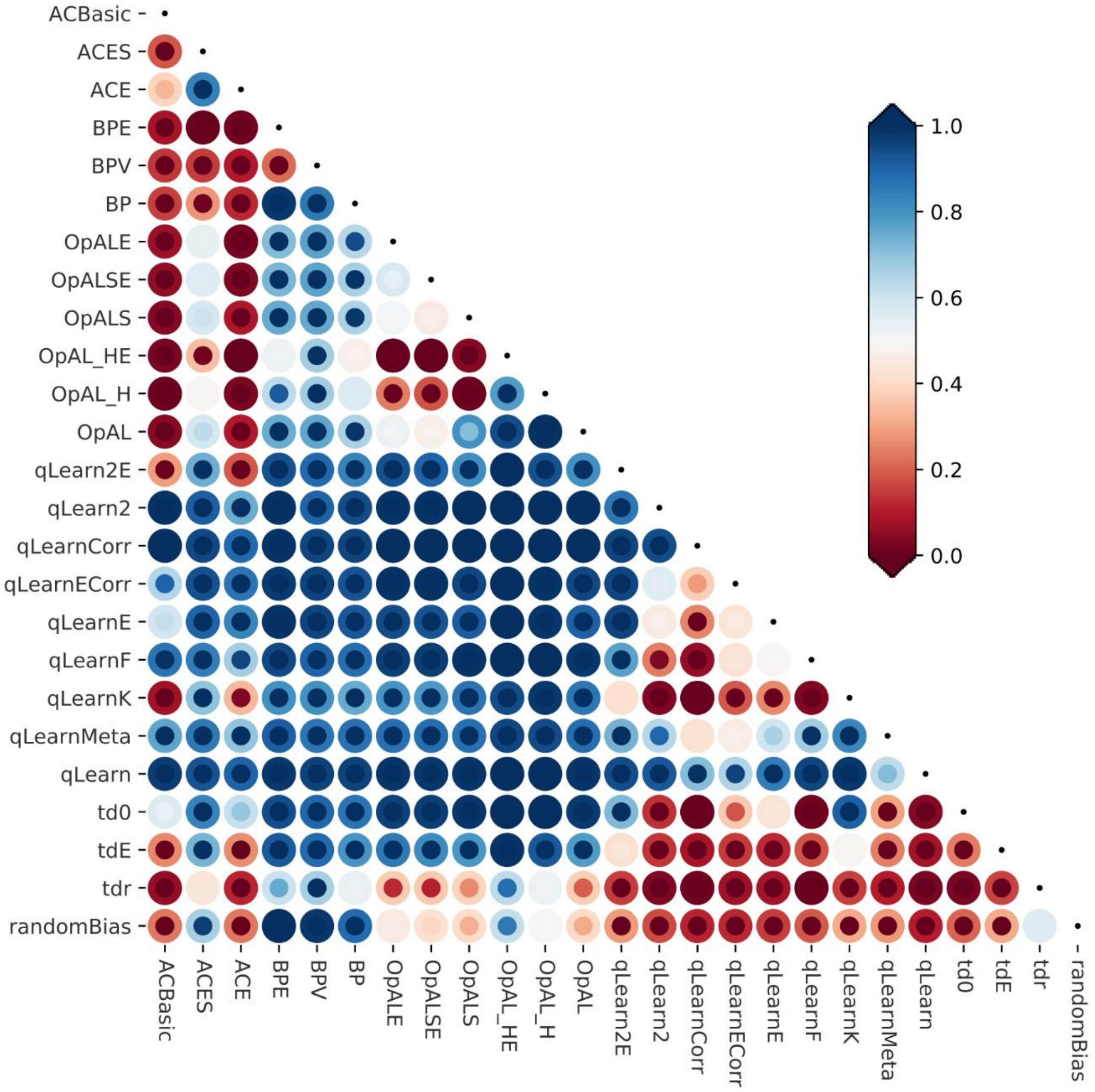


Figure 6-13 The expectation of the model frequencies (EF) and the associated protected exceedance probabilities (pEP) for model pairs. Each pair of circles shows the EF and pEP for the vertical axis model relative to that of the horizontal axis model. The outer, larger circle is the EF and the inner circle is the pEP. Both are scaled between [0,1].

By comparing the points won by each participant against the Bayes factor of their model fit, it was found that each model had a significant, Bonferroni corrected, negative correlation: participants who won the most points had the weakest evidence compared to a pure random model and the participants with the fewest points were fitted with strong evidence for the models. One of the clearest of the relationships is for the biased random model, shown in Figure 6-14. This provides further evidence that using the biased random model as the baseline for the Decks task will provide a stricter baseline than the pure random model, while still not having any learning components.

Using this new baseline for model performance, the relationship between points won and the Bayes factor of their model fit, shown in Figure 6-15, is no longer so clear. The remaining models did not have strong evidence for their explanations of the majority of both the highest (above 485) and lowest (below 425) points earning participants, but the models had stronger evidence for the middle points earning participants. The distinction between the high and medium points earners is quite marked. Figure 6-3 shows that this represents a selection of between 9 and 44 deck 2 cards. As this transition is in the same place across models, it suggests that participants who managed to gain more than 485 points were doing so using methods of exploration and preference switching that are not properly captured by these models. In this respect, the Bayesian models do seem to capture the participant's responses more consistently, even if it is not well captured.

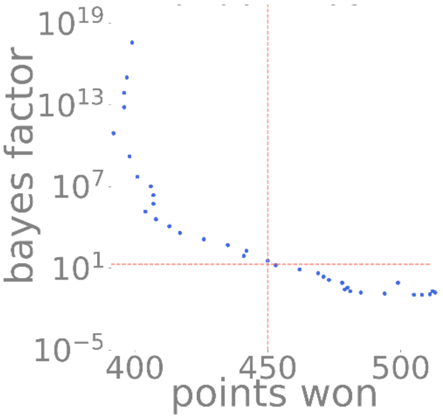


Figure 6-14 A comparison of the points won by each participant vs the model fit Bayes factors for the biased random model. The vertical line marks 450 points and the horizontal line a Bayes factor of 20, above which the model fits have more than strong evidence that they are better than the pure random model.

Figure 6-16 shows that using the biased random model as the baseline, compared to the pure random model baseline in Figure 6-8, has spread out the participant fit Bayes factors, with the exception of the Bayesian models whose fits have more consistent Bayes factors. This difference in treatment can be understood using the model comparisons in Figure 6-10, where the Bayesian model fits can be seen to closely match those of the randomBias model.

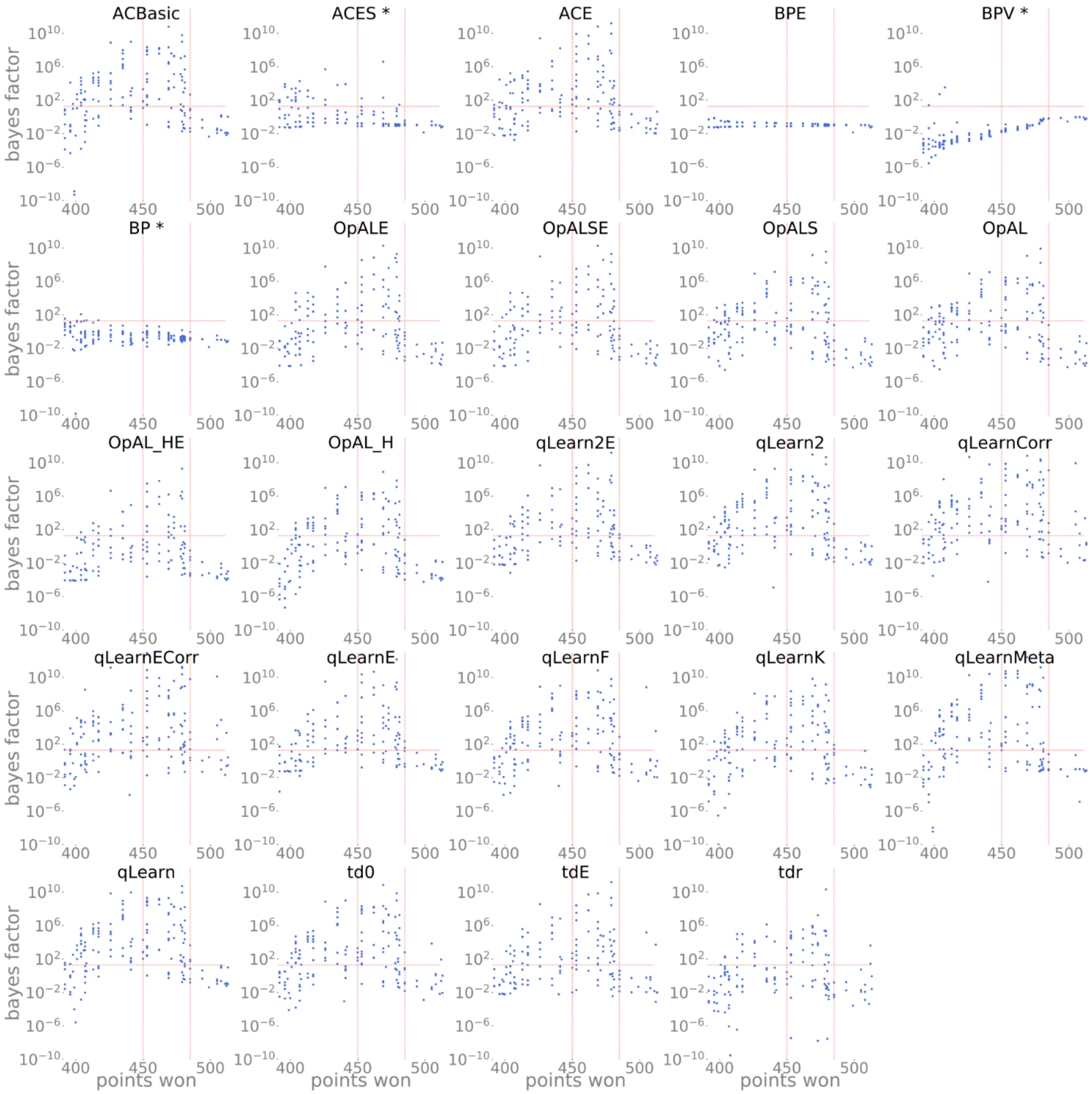


Figure 6-15 A comparison of the points won by each participant vs the model fit Bayes factors for each of the models. The vertical lines mark 450 and 485 points and the horizontal line a Bayes factor of 20. The models are labelled above each plot. The star denotes a Bonferroni corrected Spearman's rank correlation with a p <0.05

When removing all the edge fits found in chapter 6.2.1, the distributions of fit qualities do not change significantly, suggesting that model fits that result in parameters recovered at a boundary do not have particularly higher or lower fit quality values than others.

To provide an estimate of the number of participant fits that may be considered good fits, we can use the proportion of participants with a fit that has a Bayes factor of 20 or more, and with recovered parameters not on the edge of the parameter boundaries. The proportion of participants with not good fits, for each model, can be seen in Figure 6-17. With these two criteria combined, the difference in performance between the ε-greedy and SoftMax models disappears. The Bayes models are the worst fitted, with almost 100% of participants rejected by our criteria. The best good fit proportions are from the qLearn model variants that have between 35-45% of rejected fits.

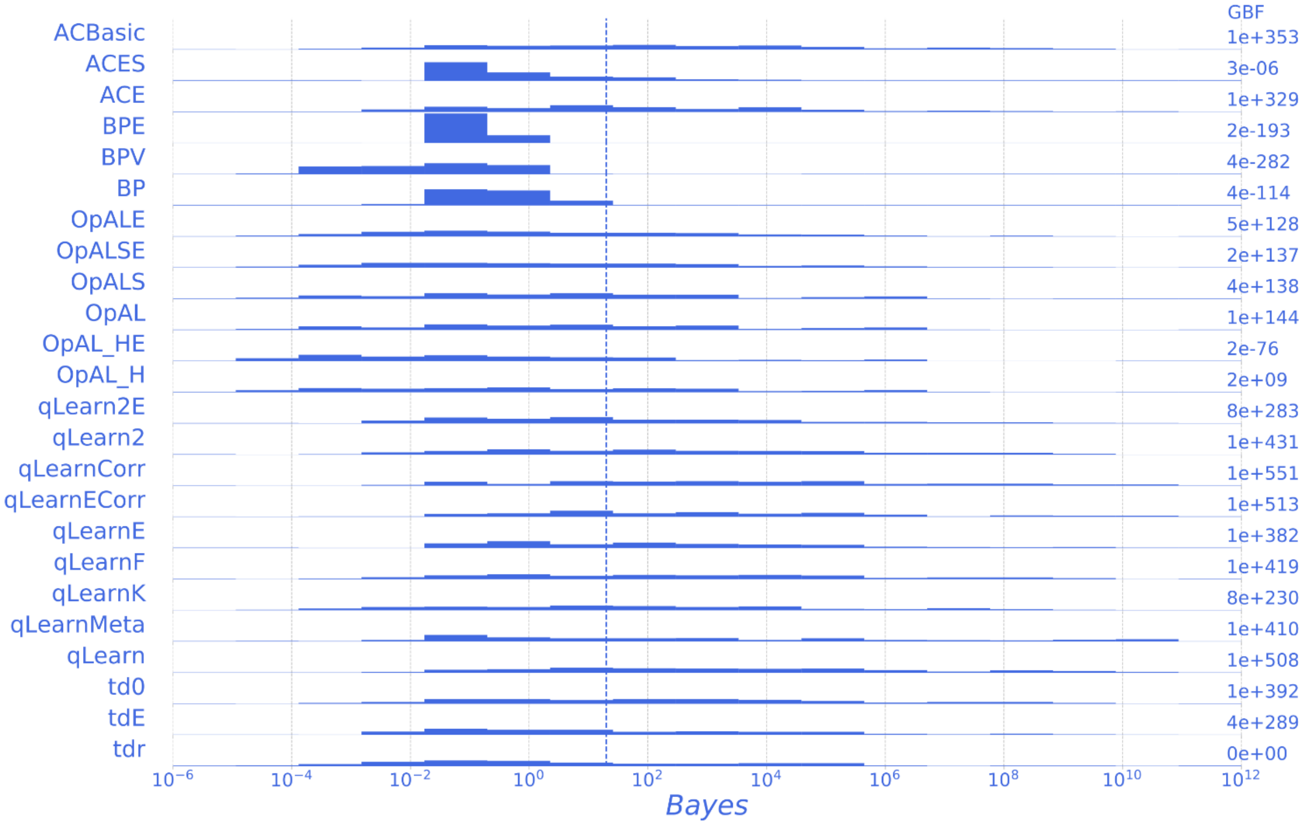


Figure 6-16 The distribution of the values of the fit quality Bayes factor from fitting the Decks dataset when compared to the biased random model. The dashed vertical line marks a Bayes factor of 20. Values above 20 have strong evidence that the model can match the participant's actions better than the biased random model.

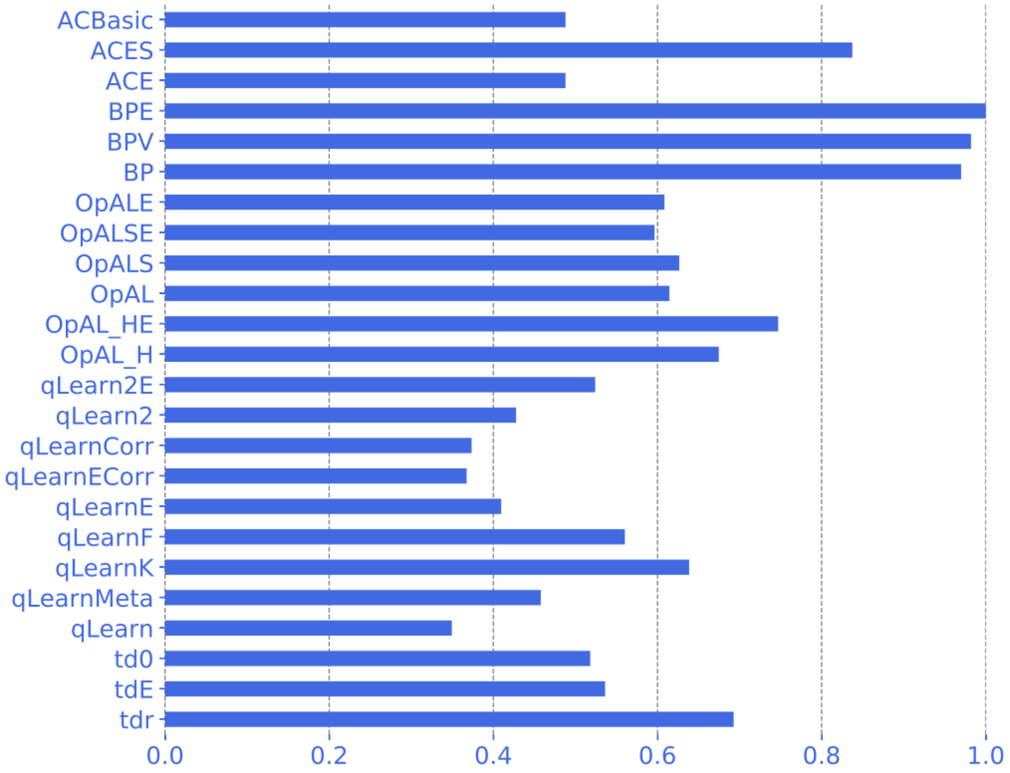


Figure 6-17 The proportion of the 166 fits where at least one recovered parameter was within 0.1% of its boundary and the Bayes factor of the fit was below 20 when compared to the biased random model.

6.2.3 Parameter correlations

If the recovered model parameters are identifying a feature of a participant's learning and decision-making process then we would expect that in different models the same parameter, performing the same task, would have very similar values. The correlations between parameters across models should therefore be high for parameters performing the same role in different models and low between parameters performing different roles, especially those in the same model. In this section, the correlations of the three most common parameters are examined: α, β and ε. A plot of the full comparison between model parameters can be seen in Appendix II. The correlations for the learning rate parameter, α, shown in Figure 6-18, need to be broken down further, as learning rates are used for learning various estimators.

One such group is the Q-learning class of models, with only one learning rate parameter and without separate actor and critic components, as described in chapters 3.2, 3.5, 0 and 4.8. Collectively, their correlations result in a Kendall's W, a measure of collective concordance, of 0.64, which suggests some correlation, but not a very strong one. Looking at the individual model parameter pair correlations, shown in Figure 6-19, it can be seen that all the correlations are positive, but there is significant variation in their strength. The correlations between the critic α parameters, also shown in Figure 6-19, show a similar pattern, albeit with slightly weaker correlations.

As with α, correlations could be expected for the α^+ and α^- parameters found in the OpAL models and qLearn2 variants. However, as can be seen in Figure 6-20, there are no correlations within these groups of parameters, with Kendall's W values of around 0.15. This is surprising, given the similarity of the OpAL models and equally surprising for the qLearn2 models. This may due to the difficulty in teasing apart the influence of the α^+ and α^- parameters. However, the correlations between these parameters within each model are also not high in the case of the qLearn2 models (0.27 for qLearn2 and 0.16 for qLearn2E) nor for the OpAL models (values ranging from -0.03 and 0.2).

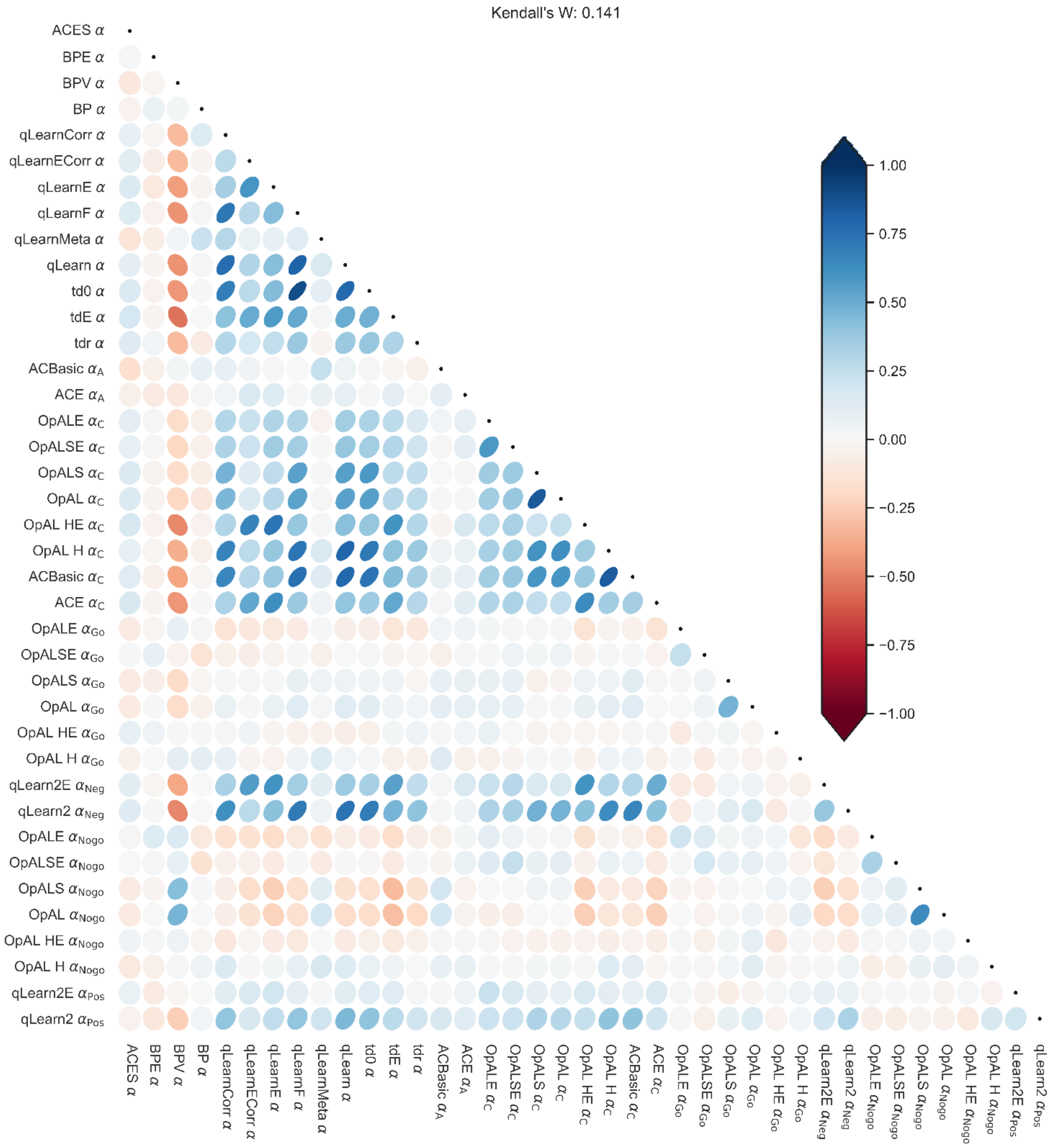


Figure 6-18 The correlations between recovered α parameter values from the Decks task participants. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation.

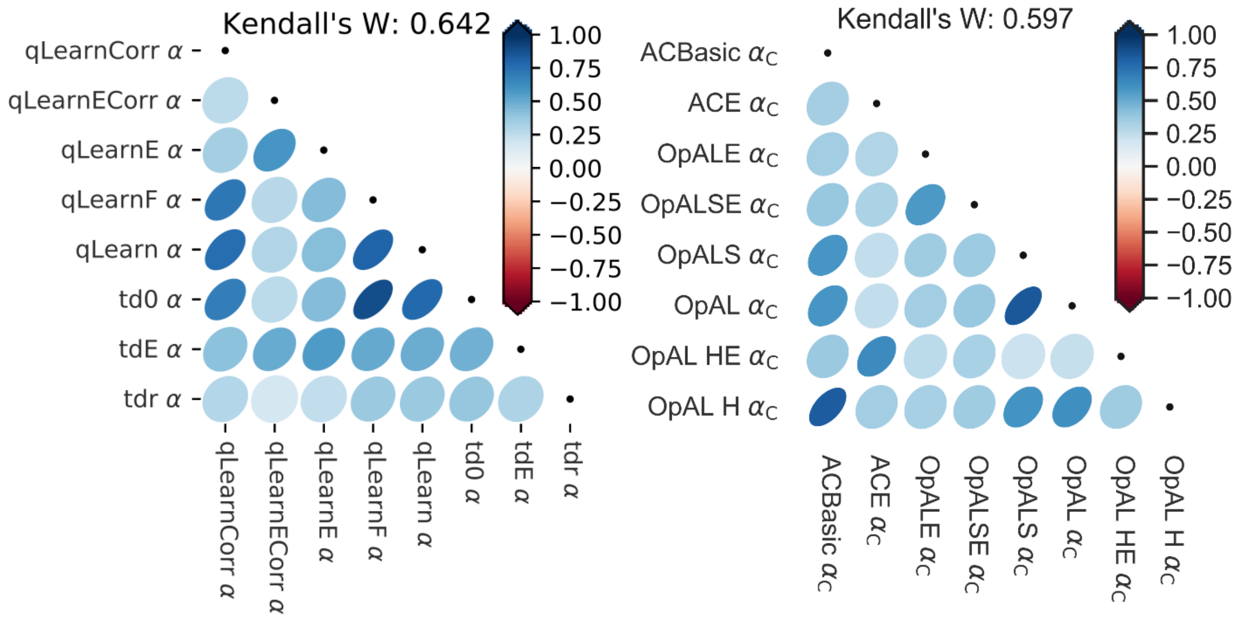


Figure 6-19 The correlations between recovered parameter values from the Decks task participants. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation. ***Left:*** *The critic learning rate parameters, α, in the critic only q-learning models* ***Right:*** *The critic learning rates from models with both actor and critic learning rates.*

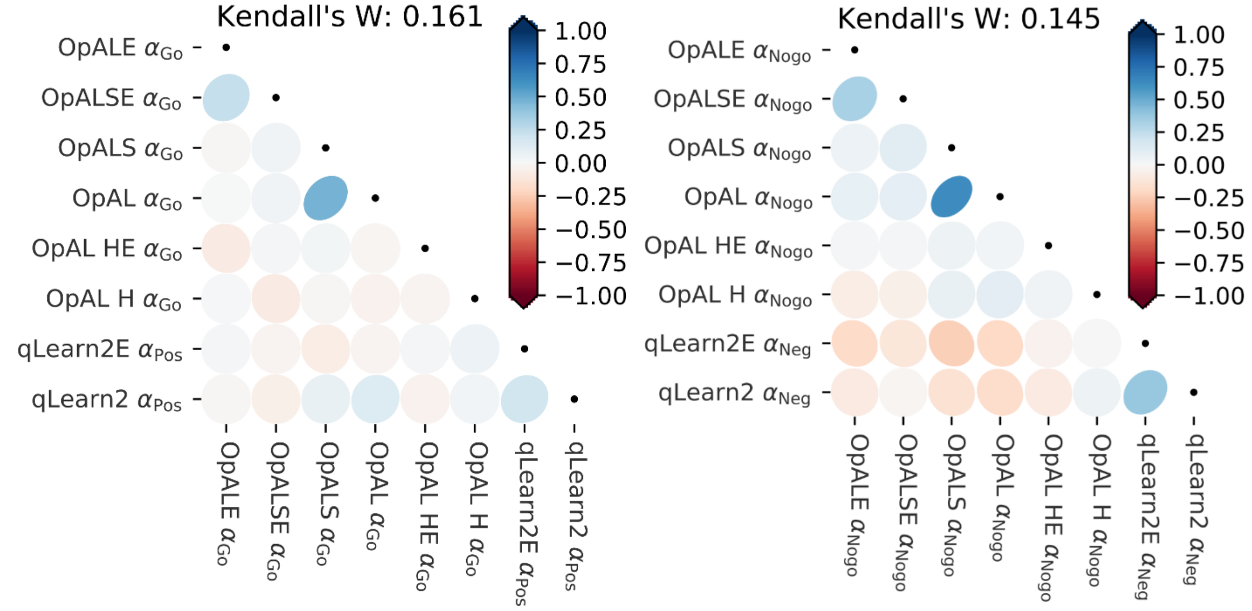


Figure 6-20 The correlations between recovered parameter values from the Decks task participants. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation. ***Left:*** *The learning rate for positive rewards* ***Right:*** *The learning rate for negative rewards.*

The same analysis was performed for the β and ε parameters, as shown in Figure 6-21 and Figure 6-22. The recovered ε parameters have strong correlations, as shown by the Kendall's W value across the models of 0.88. This contrasts with the large number of weak correlations for β, with a Kendall's W of 0.27, acting as further indications that the ε-greedy function provides more consistent parameter recovery than that of SoftMax. The only β parameters that are strongly correlated to each other are those for the Q-learning class of models shown in Figure 6-19.

The variations in similarity between recovered parameter values could be explained by differences in a model's capacity to accurately express the

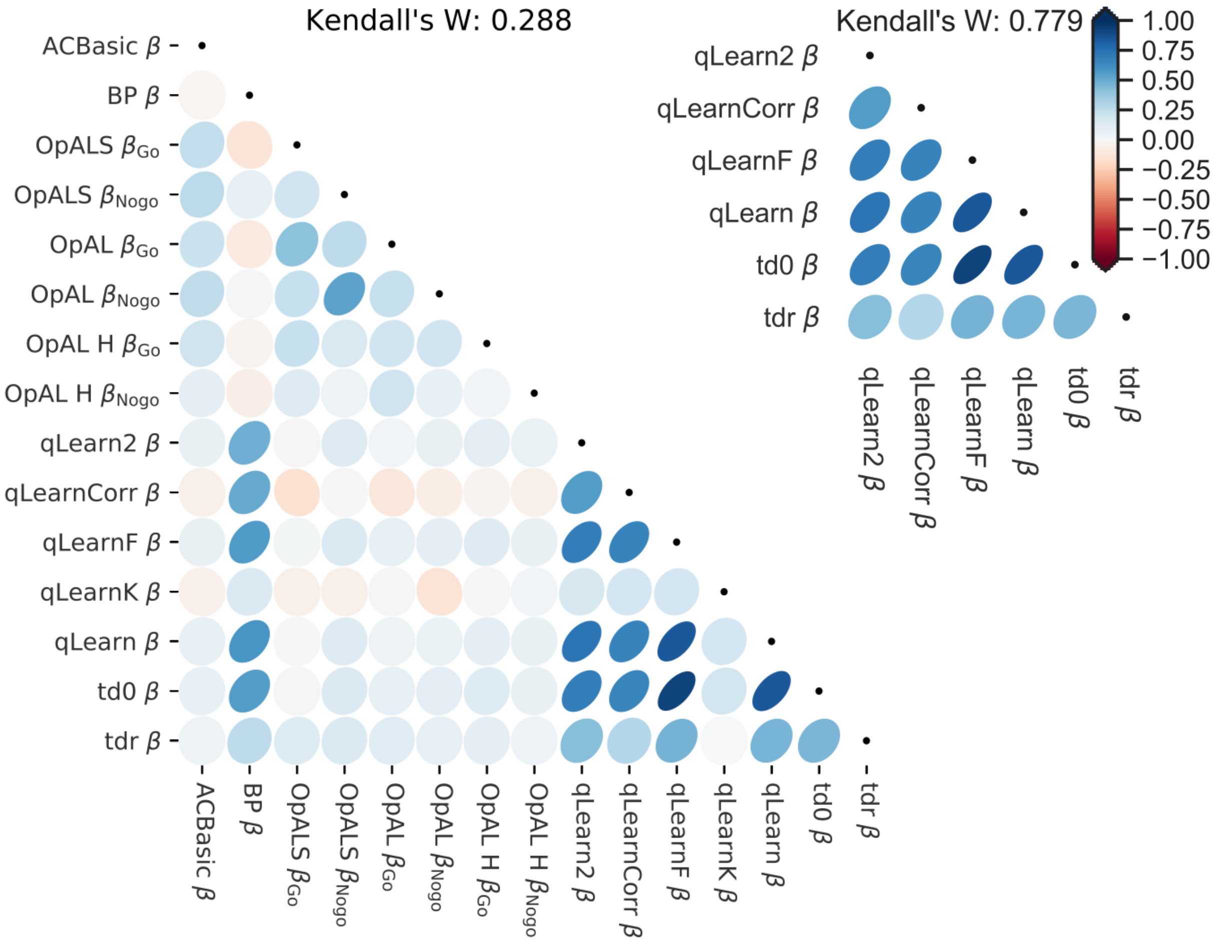


Figure 6-21 The correlations between recovered parameter values from the Decks task participants. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation. ***Left:*** *The β parameters for the Q-learning models* ***Right:*** *The β parameters in all models.*

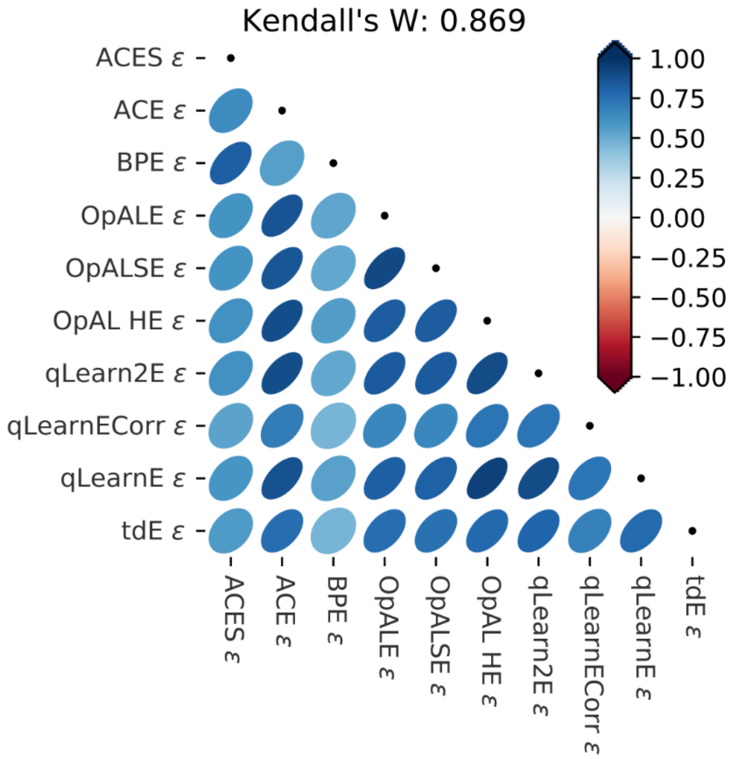


Figure 6-22 The correlations between recovered parameter values from the Decks task participants for models with an ε parameter. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation.

performance of each participant, i.e. how close they are to a 'true' model of how a participant is choosing their next action. Those models that express the participant's performance less well will have less difference between their fits of 'good' and 'bad' parameter values, resulting in there being greater noise in their final recovered parameters.

Another possible cause of variation in similarity is that some models are harder to fit participant action sequences to than others, due to differences in the complexity of their structure or the number of parameters they contain. Fitting model parameters to data is well known to result in correlated errors between fitted model parameters (Schmiedek, Oberauer, Wilhelm, Süss, & Wittmann, 2007). In chapter 4, the distribution of errors in parameter recovery for the qLearn model parameters α and β suggested that the recovery process resulted in the parameters being inversely correlated. For this Decks task dataset, the correlation was found to be -0.5, which matches with the previous results. By contrast, for qLearnE, the correlation between α and ε was found to be 0.22, which adds evidence to ε-greedy being more recoverable than the SoftMax.

From the assessment of recovered parameter correlations in this section, the number of parameters does not seem to have a high predictive power on the strength of correlations between parameters, as models with comparable number of parameters, such as the Q-learning, actor critic and Bayesian inspired models do not have similar parameter correlational strengths. However, the structure of the model does appear to have an influence on the likelihood of having correlated model parameter values, with the Q-learning class of models having more consistent recovery of parameters and those with ε-greedy having their ε parameter more consistently recovered than their β parameter counterparts. This provides further evidence that epsilon greedy performs better than softmax for consistently modelling participant actions.

6.3 Model parameter relationships to extraversion

Having established how successful these models are at reflecting the variations in actions of the different participants, it was possible to examine if there were a correlation between the recovered α parameter and a measure of the extraversion phenotype. As established in chapter 1.1, extraversion is likely to modulate the magnitude of the RPE. This would indicate a possible correlation could be found between extraversion and the learning rate parameter α (Pickering & Pesola, 2014). Pickering in unpublished analyses had shown a correlation for the first of the three datasets between α in the Q-learning model and the extraversion measure of EPQ-R. It was therefore of interest to compare the extraversion measures for the participants to the fitted parameter values for Q-learning class of models, with only one learning rate parameter and without separate actor and critic components, shown in Figure 6-19 (left). These are: qLearnCorr, qLearnECorr, qLearnE, qLearnF, qLearn, td0, tdE and tdr.

In chapter 4, the model parameter recovery was shown to be noisy for α under ideal conditions. To minimise the model recovery noise, the mean of the α values was calculated for each participant from those recovered from the Q-learning class of models. A Spearman's rank correlation was performed between the mean α value for each participant and the participant’s measured extraversion value of EPQ-R.

As in the previous section it was noted that there were, in many of the models, correlations between model parameters from the same model, the correlations between the recovered α parameter and the extraversion measure were also performed with the β or ε parameter variation accounted for. As the models either had a β or ε parameter, means of α were also calculated for those models with a β parameter and for those with an ε parameter. Estimates of those models mean β and ε parameters were also calculated for each participant. Partial correlations were performed between the extraversion measure and each of these α parameter estimates.

As seen in chapter 6.2.2, there was only strong evidence for some of the recovered model parameter combinations being a better explanation of the participant actions than those of the biased random model. Equally, in chapter 6.2.1, it was seen that some of the recovered model parameter combinations were very close to parameter boundaries. A subset of recovered α parameters, with both strong evidence for a recovered model parameter combination and a lack of boundary collisions, were used to calculate a mean α value for each participant. In the same way, mean values of α parameter estimates were also calculated separately for those models using softmax or ε-greedy. Equally, estimates were calculated for both the β parameter and ε parameters of those models. These were then used to calculate the partial correlations between the extraversion measure and the α parameter estimates.

In total this resulted in six correlations between participant estimates of α and the extraversion measure of EPQ-R, shown in Table 6-6. It is notable that all of these show a negative correlation between α and EPQ-R extraversion and that the correlations all become stronger when the parameter estimates were only calculated using parameter sets that both had strong evidence and a lack of boundary values. This correlation strengthened slightly when limited to models using softmax and with the corresponding β values partialled out. This strengthening was seen both when using all parameter sets and when using only the parameter sets that both had strong evidence and a lack of boundary values.

Participant model fit parameters used	*All*		*Good edge & Bayes*	
	ρ (DF)	*p*	*ρ (DF)*	*p*
Q-learning models	-0.091 (161)	0.251	-0.204 (123)	0.023
Q-learning models with β	-0.176 (158)	0.026	-0.283 (116)	0.002
Q-learning models with ε	-0.021 (158)	0.795	-0.150 (109)	0.115

Table 6-6 The correlations between each participant's averaged Q-learning model parameter α and the extraversion measure of EPQ-R. The α, β and ε values in these correlations are averaged, for each participant, across the relevant subset of models. Partial correlations were used for the β and ε subsets whereas the others were simple correlations.

6.4 Discussion

The Decks task tested participant's capacity to learn about changing payoffs. On average, the 166 participants were found to have adequately identified both the deck with the highest initial payoff and the need to switch decks as the task progressed.

The models were evaluated for their performance in producing the same action choices as those of the participants, with the baseline for their fit quality being set as the performance of the biased random model. Models were fitted on all the action choice trials performed by each participant.

The model fit accuracy varied by type of model, with those based on Q-learning providing some of the best fits, notably qLearn, qLearnCorr and qLearnCorrE, and the OpAL and Bayesian inspired models providing some of the worst.

Participant fits were found to be highly correlated between models, with some participants consistently being well fitted and others consistently badly fitted. Goodness of model fits was found to be inversely correlated with points won in the task when compared to a pure random model, but not the biased random model, which was able to explain most of this correlation. While most models provided strong evidence, as defined in chapter 2.2, that they explained the majority of participant actions better than the biased random model, those participants who performed well in the task had weaker Bayes factors. This suggests that the models were good at representing the participants with the less successful strategies, but not those with the most successful strategies, suggesting that there are elements of the strategies of the most successful subjects that cannot be encapsulated by these models.

The ε-greedy parameters were recovered consistently across all models and more consistently than the β parameter from equivalent models using SoftMax. The only consistently recovered β parameters were those of Q-learning variants. The α parameters were inconsistently recovered, with the critic learning rates best recover. The strength of correlations between α parameters was found to be related to the type of model, as had been seen with the fit quality.

The most consistent α critic learning rates were found to negatively correlate with the extraversion measure. This could suggest that extraversion is correlated with decreasing sensitivity to errors in expected rewards, which would result in extraverts learning more slowly than introverts for the same RPE. If this were the case, in probabilistic rewarding tasks, such as this one, it would suggest that extraverts would be less sensitive to the reward variability and more able to identify changes in average reward values, as demonstrated in Figure 5-1.

Pickering & Pesola (2014) suggested that α could correspond to the density of some dopamine receptors controlling dopaminergic-mediated reinforcement learning. Given the negative correlation between extraversion and α and the positive correlation between extraversion and RPE magnitude, the impact on learning remains unclear. As the extraversion measure did not correlate with the points won nor the difference in the average action choices between the first and second 40 trials, this correlation with α would not be identifiable without modelling the participants learning process.

It is worth noting that these results reflect the performance of the models only as implemented. There may be other implementations that would perform differently with slight tweaks in their implementation.

7 Probabilistic Selection task

The second task used was a version of the Probabilistic Selection task, first used by Frank, Seeberger, & O'Reilly (2004). The dataset discussed in this chapter was collected using a version of the task taken from Frank et al. (2007). Participants are shown a series of pairs of Hiragana characters, from the possible set of six characters, and asked to pick which of the two is the 'correct symbol'. The participant was given four seconds to respond for each trial. If no response was made, the trial was marked as not completed and the task moved on to the next trial. In this model fitting and analysis, these incomplete trials have not been included. Participants were given ten practice trials before starting the learning phase with another set of two Kanji symbols, one of which was the correct choice 7 out of the ten trials. In the first part of the task, the *learning trials*, the six characters were shown in three pairs, with complementary reward probabilities: {A:80%, B: 20%}, {C:70%, D: 30%}, {E:60%, F: 40%}. They were then told if they were correct. The learning phase was broken into blocks of 60 trials, with 20 trials for each character pair and 10 trials for each special arrangement for each character pair (e.g. AB and BA). For each correct choice, the participant won 5 pence and their cumulative winnings was shown after each trial. At the end of a block of trials, the proportion of correct responses was calculated for each character pair. If the

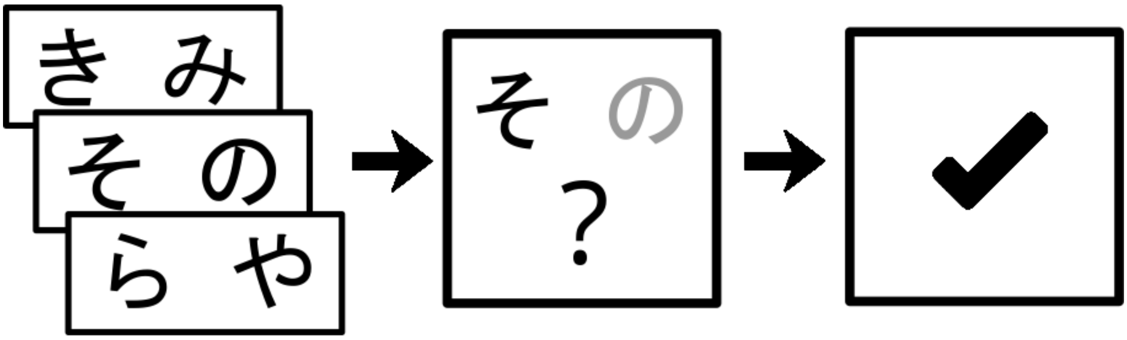


Figure 7-1 The Probabilistic Selection task. Participants are shown pairs of characters, from a set of six, and asked to pick the correct one. Each character has a different likelihood of being correct. During the initial learning phase, the characters are shown three pairs, with complementary reward likelihoods, multiple times and participants are given feedback. In the test phase, participants are presented with every combination of pairs of characters but are not given feedback.

participants success rate exceeded a specified threshold for one of the pairs (AB > 65% correct, CD > 60% correct and EF > 55% correct) then participants moved on to the test phase of the task. If not, they performed another block of 60 trials, with the character pairs presented in a different trial sequence. A maximum of six blocks of learning phase trials could be given to a participant before moving them on to the test phase. In the participant data examined in this chapter, participants received no more than four blocks of learning phase trials. In the second stage of the task, the *test trials*, the participants were again shown pairs of characters, but in this case all possible combinations of the six characters were shown. The participants were not given any feedback as to their performance. The test phase comprised of 60 trials, with 4 examples of each of the 15 character-pair combinations and 2 trials with each special arrangement for each character pair. The same sequence was used for each of the participants and they continued to be paid for each correct response, with their cumulative winnings shown to them at the end of the task. The sequence of character pairs and the order in which characters were shown on the screen varied quasi-randomly but was fixed across participants. An example sequence is shown in Figure 7-2 with one block of learning trials and the test trials. The characters used for A and B were exchanged for half the participants to eliminate the possibility that any association was due to their shape.

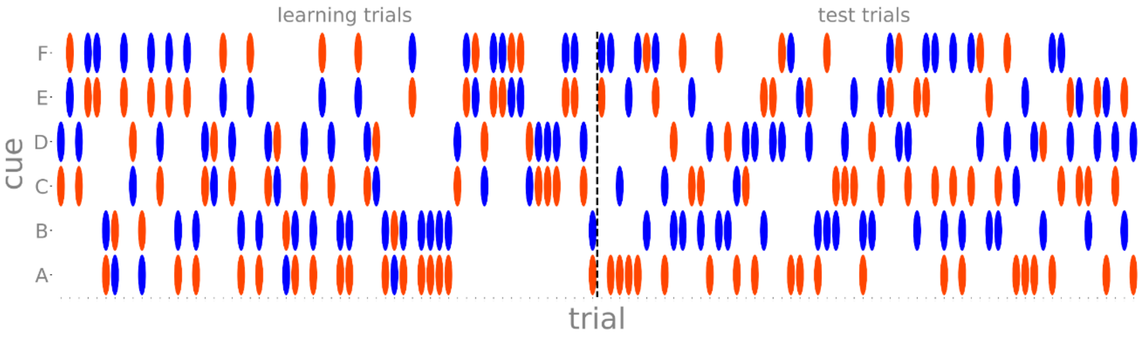


Figure 7-2 The characters (cues) displayed during each trial. The ovals indicate which cues were visible for each of the 60 learning trials and 60 test trials. The red oval denotes the correct, rewarded, cue and the blue oval the incorrect, unrewarded, trial. The black dotted line marks the transition from learning trials to test trials.

7.1 DATA PROPERTIES

The dataset contains 71 participants, of which 19 were men, collected at Goldsmiths, University of London as part of a masters project. The participant's ages ranged from 19 to 59 years, with a mean age of 26 years and a standard deviation of 7.5 years. No participants had past familiarity of Hiragana characters.

Before the data collection, the intention was that all three character pair success criteria would have to be met before participants could move on to the test phase. Instead, due to a coding error in the task program, participants could move on to the test phase if they met at least one of the criteria. As the criteria for the EF trials was barely above chance (EF > 55% correct), even if participants chose at random it was possible for them to only have one block of learning trials. Had the learning criteria been correctly used, only 5 of the 71 participants would have had only one block of learning trials. As it was, 56 participants had only one block of learning trials. However, only 5 of the 15 participants who had more learning trials improved their choices as they performed more trials. Comparing the performance of the participants with more blocks of learning trials, there was no indication that the extra trials resulted in an improvement in performance for AB t(31.77)= 0.21, p= 0.83, CD t(26.01)= 1.16, p= 0.25 or EF t(30.88)= 0.79, p= 0.43. It is likely that those participants who were selected to have more blocks of learning trials performed below average in their understanding of the task in the first trial block. The extra blocks of learning trials may therfore have brought their understanding of the task to the level of the other participants.

One measure of participant's performance is to see if they changed their choice behaviour when moving from the learning phase to the test phase, where there was a lack of feedback and trials with new character pairings. Their character choices for the same character pairs can be compared between the test and learning phases, where the learning phase performance is evaluated across all blocks of learning trials a participant performed. For all three character pairs there was no significant change in the distributions of choice behaviour when moving from the learning phase to the test phase t(70)= -1.78, p= 0.08 for AB, t(70)= -1.15, p= 0.25 for CD and t(70)= -0.24, p= 0.81 for EF. Looking at the proportion of character A, C and E choices in the test and learning phases for each participant, shown in Figure 7-3, correlations between the two phases were found for the characters C (r=0.31, p= 0.01) and E (r=0.62, p< 0.001), but not for character A (r=0.20, p= 0.10). Taken together, these results suggest that most participants were

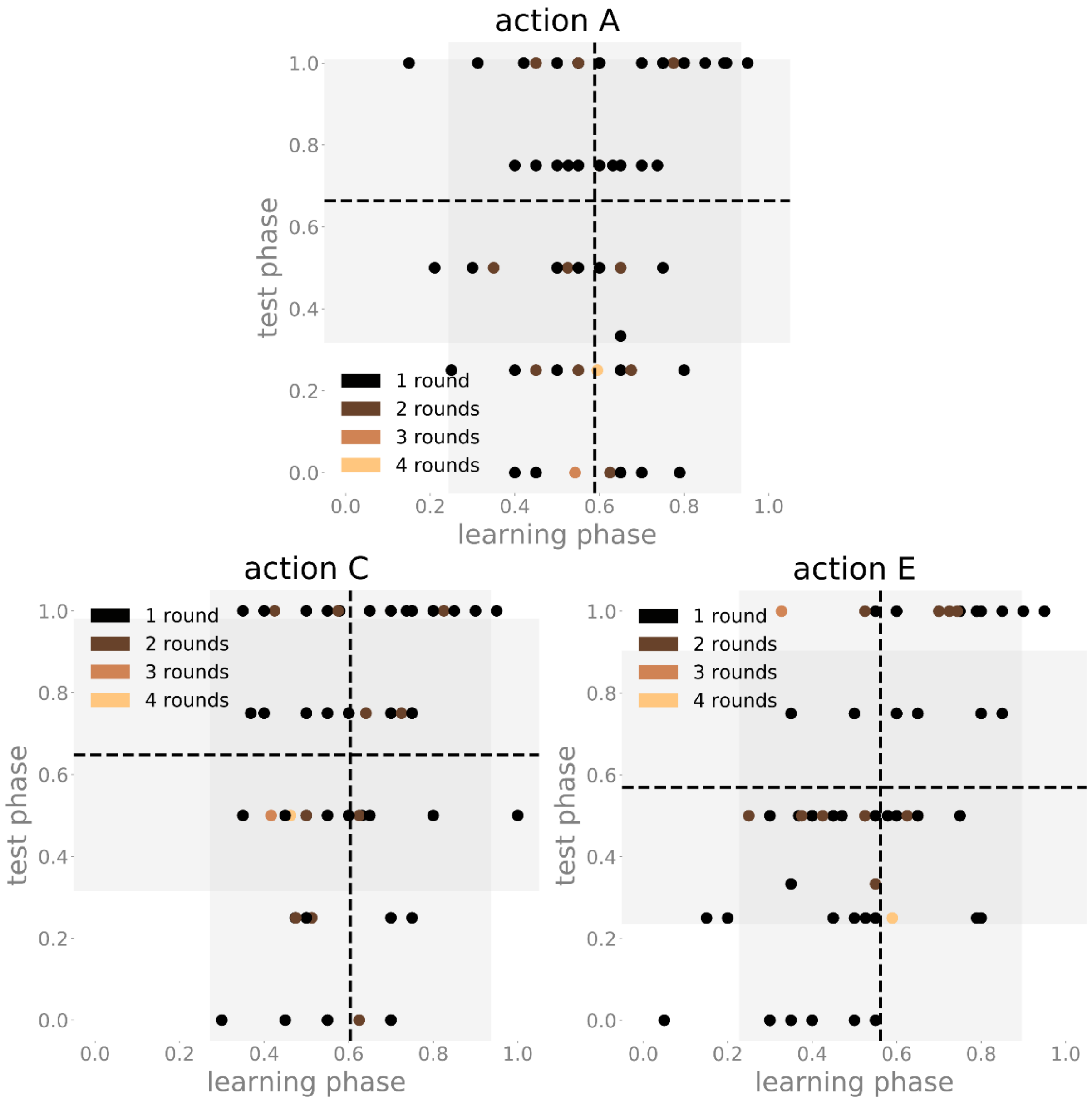


*Figure 7-3 A comparison of the choices in the learning and test phases for the AB character pair choice trials (**top**), the CD character pair choice trials (**bottom left**) and for the EF character pair choice trials (**bottom right**). As in each pair the choice of one character resulted in the other character not being chosen, so the proportions are those of the more rewarded character in each pair, namely A, C and E. For each phase, the mean is shown as a dotted line and the range one standard deviation from the mean is shown as a grey bar. The participant dots are coloured to show the number of 60 trial rounds of learning they were given.*

maintaining similar character choice proportions in the two phases, but that this became less the case the greater the difference in reward rates between the charaters in the pair. These figures also support the conclusion that the participants with more blocks of learning trials were not necessarily increasing their learning performance substancially from these extra training blocks.

A common measure of participant performance in past papers has been to look at the choices of characters A and B in the test phase new pairings (Frank et al., 2007, 2004; Simon, Howard, & Howard, 2010; Slagter, Georgopoulou, & Frank, 2015; Sojitra et al., 2018). As character A is the most likely of all the characters to return a reward, participants who have learnt this association would be expected to choose character A whenever it is presented in the test phase. The converse is true for character B. From this, *choose A* is calculated as the proportion of times character A was chosen when available in the test phase and when the other option was not character B. Equally, *avoid B* is calculated as the proportion of times character B was not chosen when available and when the other option was not character A.

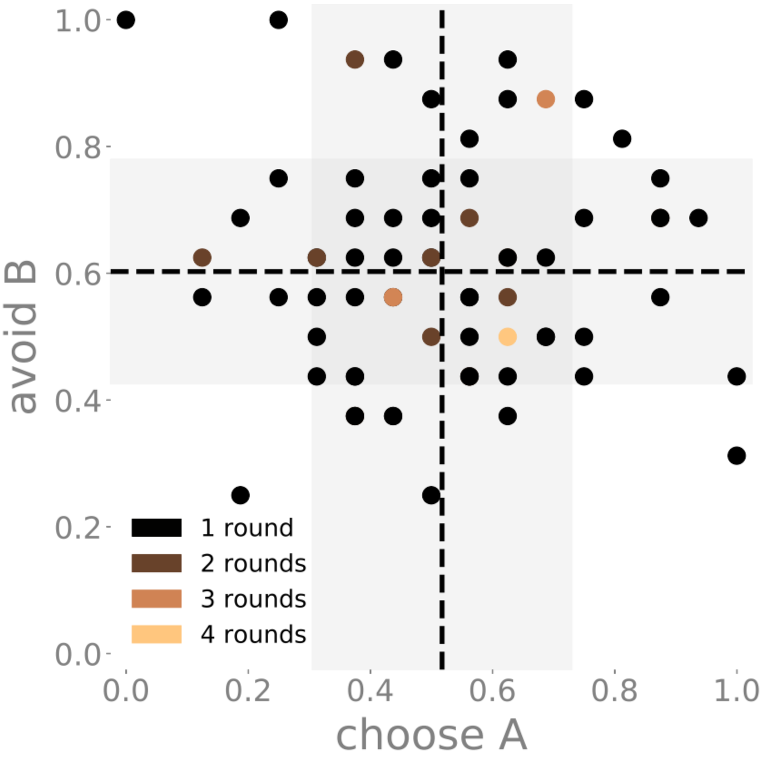


Figure 7-4 The relative distributions of participants performance with the measure of choose A and avoid B. The means are shown as black dashed lines and the grey bars around them denote one standard deviation. The colours of each point show the number of rounds each participant was given in the learning phase.

Participants on average chose A 52% of the time and avoided B 60% of the time Choose A was not found to be correlated with avoid B (r=0.03, p= 0.82), as can be seen in Figure 7-4. The number of block of learning trials did not affect participant's performance at learning choose A t(69)= 0.21, p= 0.83 or avoid B t(69)= -0.26, p= 0.79. Comparing participant's performance on these metrics with their EPQ-R extraversion scores found no correlation for either choose A (r=0.18, p= 0.19) or avoid B (r=0.01, p= 0.93). However, if a choose A outlier is removed, 2.44 standard deviations away from the mean and with a choose A value of 0, the correlation with extraversion greatly improves (r=0.23, p=0.056).

The participant's success at learning in this task can be compared to past studies with similar character reward rates, as shown in Table 7-1. Participants in this dataset performed slightly less well than those from other studies. However, they were given fewer learning trials than in the one study (Simon et al., 2010) that reported the number of learning trials performed by participants. The other published studies (Frank et al., 2007, 2004; Slagter et al., 2015) used the same procedure as Simon et al., while the current study used a weaker set of criteria for moving from the learning phase to the test phase. It is therefore very likely that the 3 other published studies also gave more learning phase trials than the current study.

Study	*Participant sample group*	*Average number of learning trials*	*Choose A*	*Avoid B*
Current	students	71 ± 35	0.52 ± 0.21	0.60 ± 0.18
(Simon et al., 2010)	college	139 ± 22	0.80 ± 0.30	0.64 ± 0.60
	old	169 ± 25	0.67 ± 0.55	0.71 ± 0.40
(Slagter et al., 2015)	low sEBR	-	0.63 + 0.60	0.86 + 0.30
	high sEBR		0.69 + 0.60	0.71 + 0.60
(Frank et al., 2007)	T/T	-	0.69 ± 0.05	0.78 ± 0.04
	C/C, C/T		0.73 ± 0.03	0.67 ± 0.03
	A/A		0.76 ± 0.03	0.67 ± 0.04
	G/G, G/A		0.67 ± 0.05	0.74 ± 0.04
	met/met, val/met		0.76 ± 0.06	0.74 ± 0.06
	val/val		0.71 ± 0.03	0.69 ± 0.03
(Frank et al., 2004)	PD on	-	0.79 ± 0.05	0.58 ± 0.11
	PD off		0.65 ± 0.07	0.82 ± 0.08
	seniors		0.68 ± 0.07	0.63 ± 0.07

Table 7-1 The mean participant choose A and avoid B values for each sample group in a series of studies, including this one. When available, the average number of trials each participant had in the learning phase is recorded.

Another measure of participant performance has been to examine how frequently participants in the learning phase stick with the same character choice after a rewarded trial, called *Win-stay*, or switch after an unrewarded trial, called *Lose-shift*. As can be seen for Win-stay in Figure 7-5 and for Lose-shift in Figure 7-6, the participants do on average learn to ignore rewarding trials of action B and ignore unrewarded trials of action A as the task progresses. This progression is clearest in the AB trials and becomes steadily less clear as the difference in rewards decreases between the pair of characters. The distributions of probabilities in

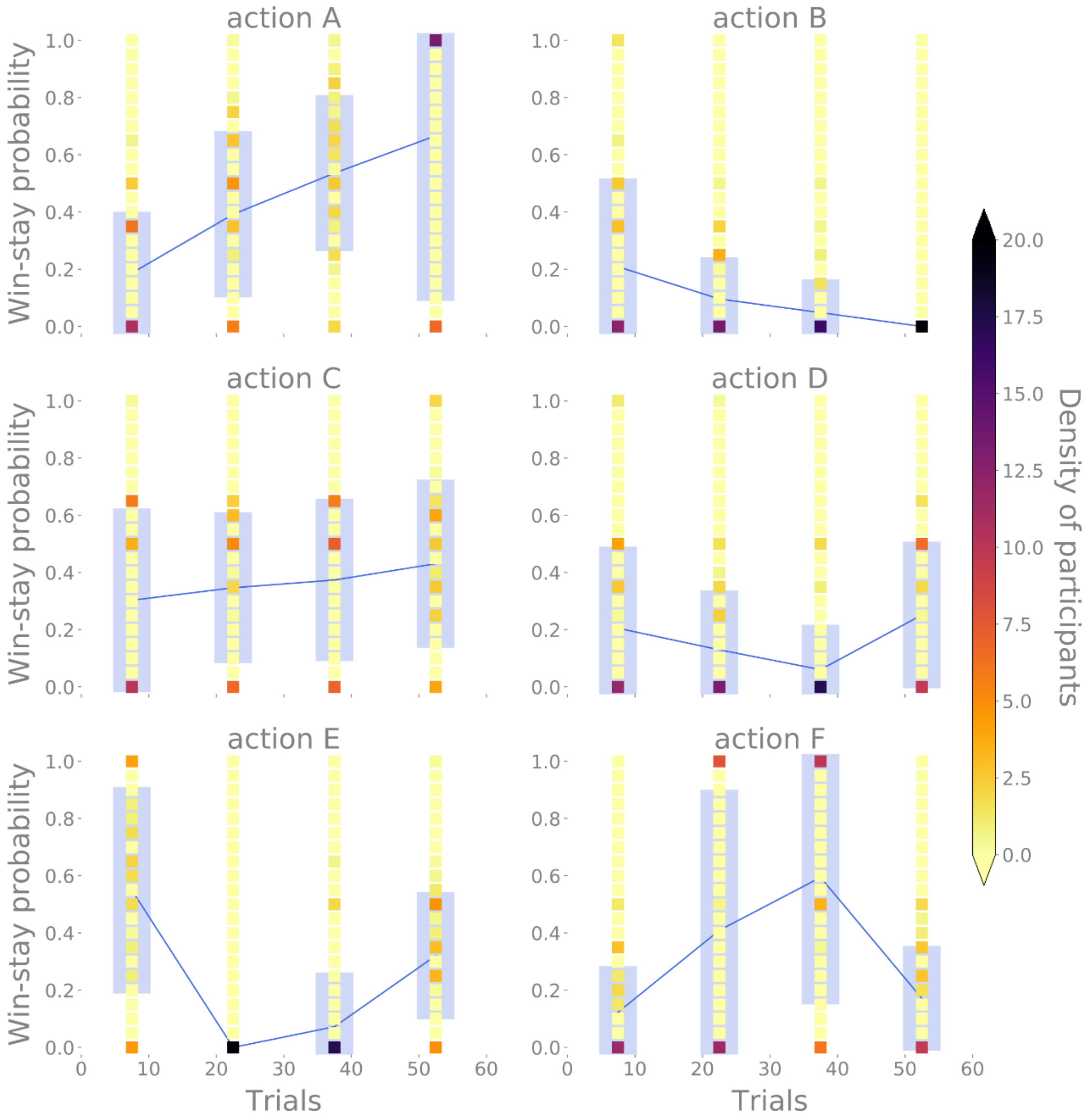


Figure 7-5 For the first four blocks of 15 trials in the learning phase, the distribution of participant Win-stay probabilities, i.e. the probability after receiving a reward of choosing the same action the next time it is presented. The mean for each block is marked by a thin blue line, one standard deviation around the mean is denoted by the pale blue bar.

these groups of 15 trials should be taken as being noisy as the number of trials with each character pair varied between each group of 15, as shown in Figure 7-2. For example, in the last 15 trials, there were only two trials with the AB character pair. As Win-stay or Lose-shift cannot be calculated for the final character pair, only one trial was used to calculate the probabilities for those characters in the last 15 trials.

Win-stay and Lose-shift can be calculated using the first 15 trials, as was done by Frank et al. (2007), who found that after the first 15 trials "individual negative

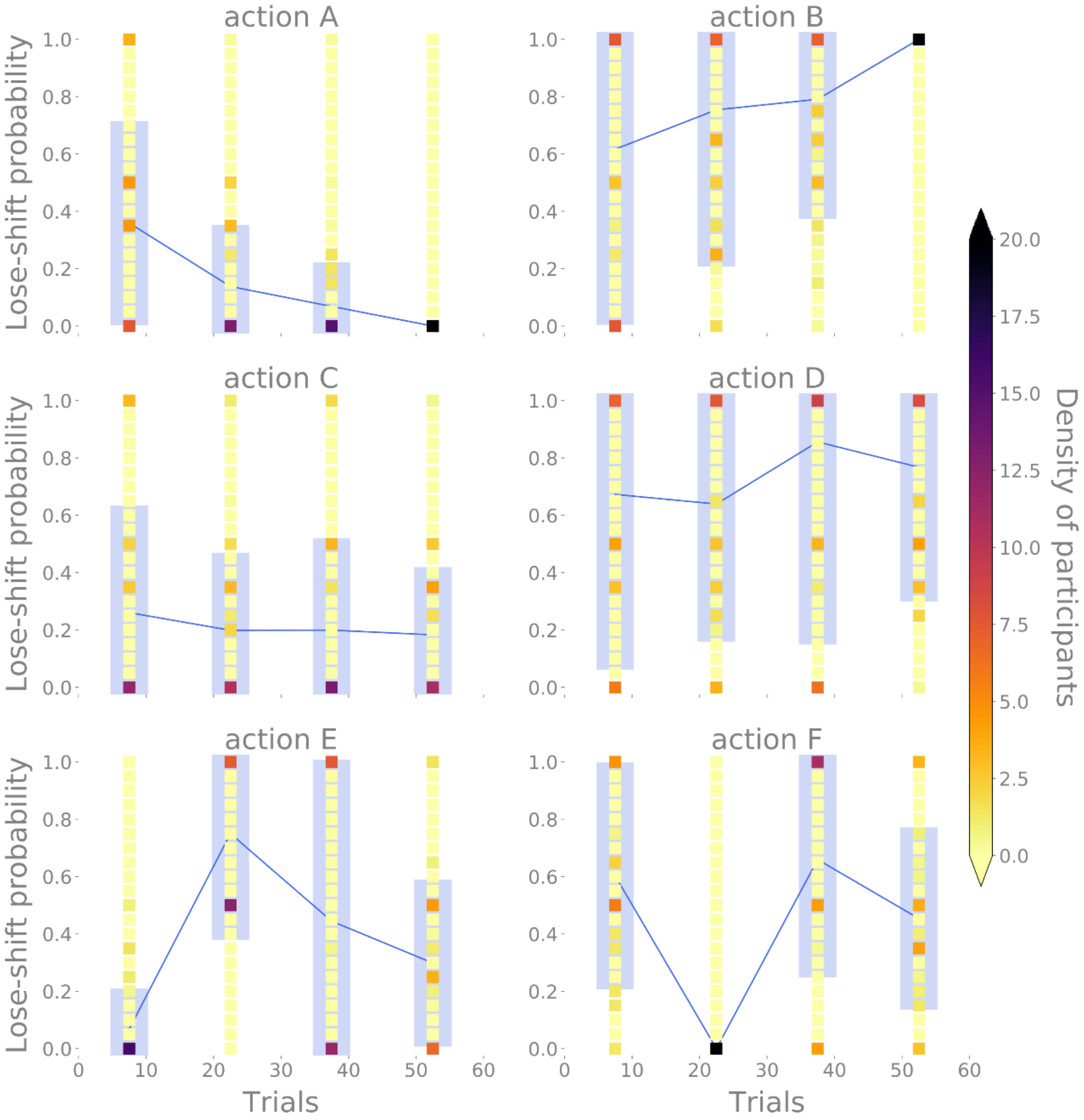


Figure 7-6 For the first four blocks of 15 trials in the learning phase, the distribution of participant Lose-shift probabilities, i.e. the probability after receiving no reward of not choosing the same action the next time it is presented. The mean for each block is marked by a thin blue line, one standard deviation around the mean is denoted by the pale blue bar.

feedback experiences became less informative”. Simon et al. (2010) used the Frank et al. study as the basis for their choice to use the first 60 trials, arguing that the “effects of feedback from individual trials can be assessed more clearly, before learning of the probabilities across trials has occurred”. That most of the learning has been completed, for an average participant, within the first 60 trials can be seen in Figure 7-5 and Figure 7-6. In Table 7-2 the Win-stay and Lose-shift values from these two studies are reported along with aggregate values from this study, calculated across the first 60 trials and averaging the results for all the characters. The values we calculated are lower than those presented in the other studies. However, it is not clear from the descriptions given in these papers if the reported figures were averaged over all characters, over all characters that could be expected to have the same trend, such as A, C and E, or if these are just calculated using character A. In all three cases the results from this study would not match those from the other studies.

Study	*Participant sample group*	*Calculation method*	*Win-stay*	*Lose-shift*
Current	students	First block (60)	0.29 ± 0.21	0.31 ± 0.26
(Simon et al., 2010)	college	First block (60)	0.83 ± 0.30	0.33 ± 0.5
	old		0.75 ± 0.20	0.36 ± 0.3
(Frank et al., 2007)	T/T	First 5 trials of each type, 15 trials total	0.68 + 0.06	0.49 + 0.05
	C/C, C/T		0.69 + 0.03	0.49 + 0.03
	A/A		0.69 ± 0.04	0.51 ± 0.03
	G/G, G/A		0.71 ± 0.04	0.49 ± 0.06
	met/met, val/met		0.68 ± 0.03	0.52 ± 0.02
	val/val		0.76 ± 0.05	0.41 ± 0.05

Table 7-2 The mean participant Win-stay and Lose-shift values for each sample group in a series of studies, including this one. The trials used to calculate these figures has also been recorded.

7.2 Fitting the models to the data

As the task rewards were either 0 or 1, within the models, the initial expected reward for both choices were set to 0.5. The upper bounds for infinitely bounded model parameters were truncated. The softmax β parameter was limited to 30, the Kalman model parameters σ_{α}^{2} and σ_{λ}^{2} were limited to 150 and the OpALS and OpALSE saturation parameters M were set to 10 and 49 respectively. The models were fitted to the participant's actions over both the learning and test phases.

During the fitting process, the models qLearnF, td0, tdr and OpAL experienced numerical overflows for certain parameter combinations, as discussed for OpAL in chapter 3.4. The parameter combinations where there has been an overflow are treated as worse than random fits, that have a probability of 0.5 for each action choice. For fits with overflows, a probability of 0.4 was returned for each action choice. If these provided a plateau of global minima, then the model could be discarded as being a worse representation of the participant's actions than a purely random model.

7.2.1 Boundary recovered parameters

The crudest measure of acceptable model fits is to measure the number of them that intersect with any of the parameter validity boundaries. In almost all cases a boundary parameter value is equivalent to removing an element of a model, thereby reducing it to a simpler model. To account for numerical errors in the fitting, a boundary, or edge, fit is considered to have occurred if the parameter is within 0.1% of either side of its range. This was chosen to be close enough to the boundaries to be unlikely to interfere with good parameter recovery, while still accounting for the approximate nature of numerical fitting.

The results from fitting the Probabilistic Selection dataset, in Figure 7-7, show the proportion of participants with at least one recovered model parameter on the edge of the parameter boundaries. Participants with boundary fits range from 1% to 37% of the sample, with no discernible pattern linking the number of edge fits and the type of model or number of parameters.

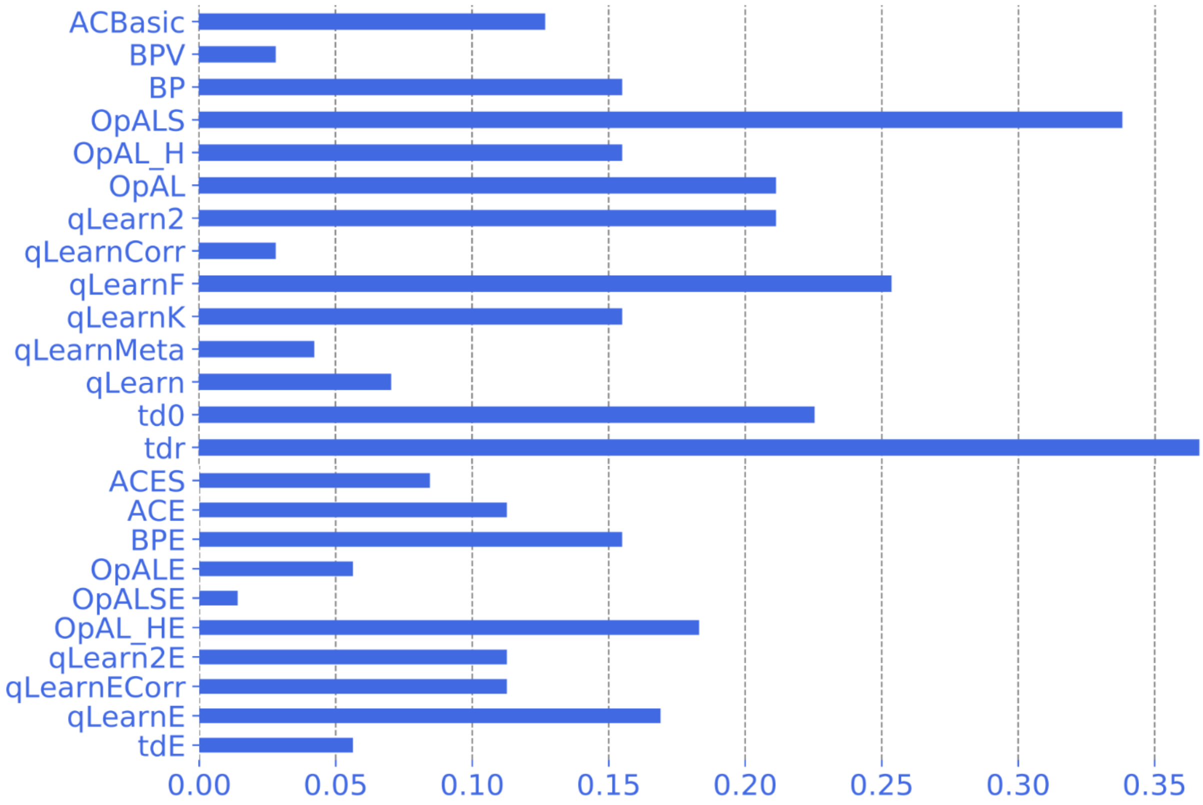


*Figure 7-7 The proportion of the 71 participants fitted to each model whose fits had at least one recovered parameter within 0.1% of its boundary. The models have been grouped into those using softmax (**top**) and those using epsilon-greedy (**bottom**).*

7.2.2 Goodness of model fits

A model's performance at fitting a participant's actions can be compared using a Bayes factor; comparing its performance to that of the random choice model, as described in chapter 2.2. For the Probabilistic Selection task dataset, using a Bayes factor of 20 as a criterion for strong evidence, as defined in chapter 2.2, for a model fit, 34 of the 71 participants had strong evidence for at least one model fitting their actions well. This can be seen in Figure 7-8, and using the normalised Bayes factor, in Figure 7-9. As with the Decks task in chapter 6.2.2, there does appear to be a similarity in Bayes factor value distributions within model classes, with the OpAL models having the weakest evidence that they are better than a purely random model, each having strong evidence for at most 17 participants. The Bayesian inspired models have some of the strongest evidence, with BPV having strong evidence for better

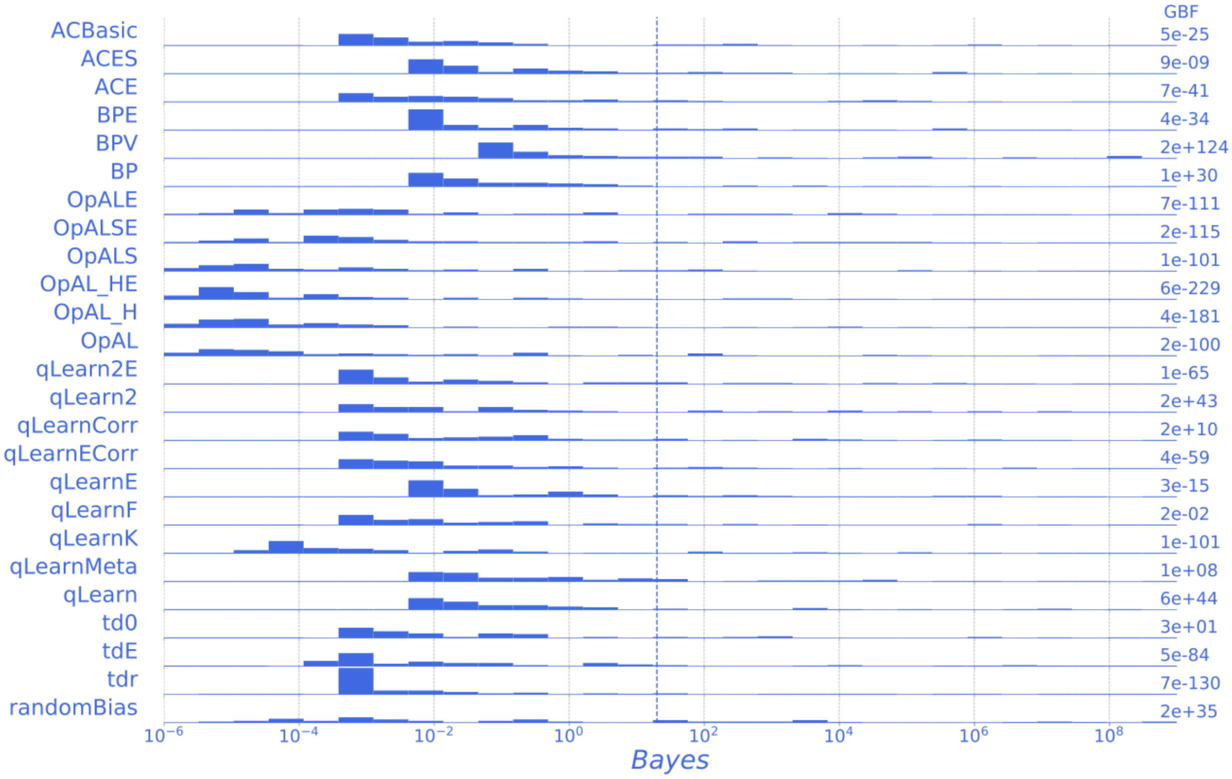


Figure 7-8 The distribution of the values of the fit quality Bayes factor from fitting the Probabilistic Selection dataset when compared to a pure random model. The dashed line marks a Bayes factor of 20, above which there is strong evidence for the model. On the right are the Group Bayes Factors for the model, defined in equation 2.9.

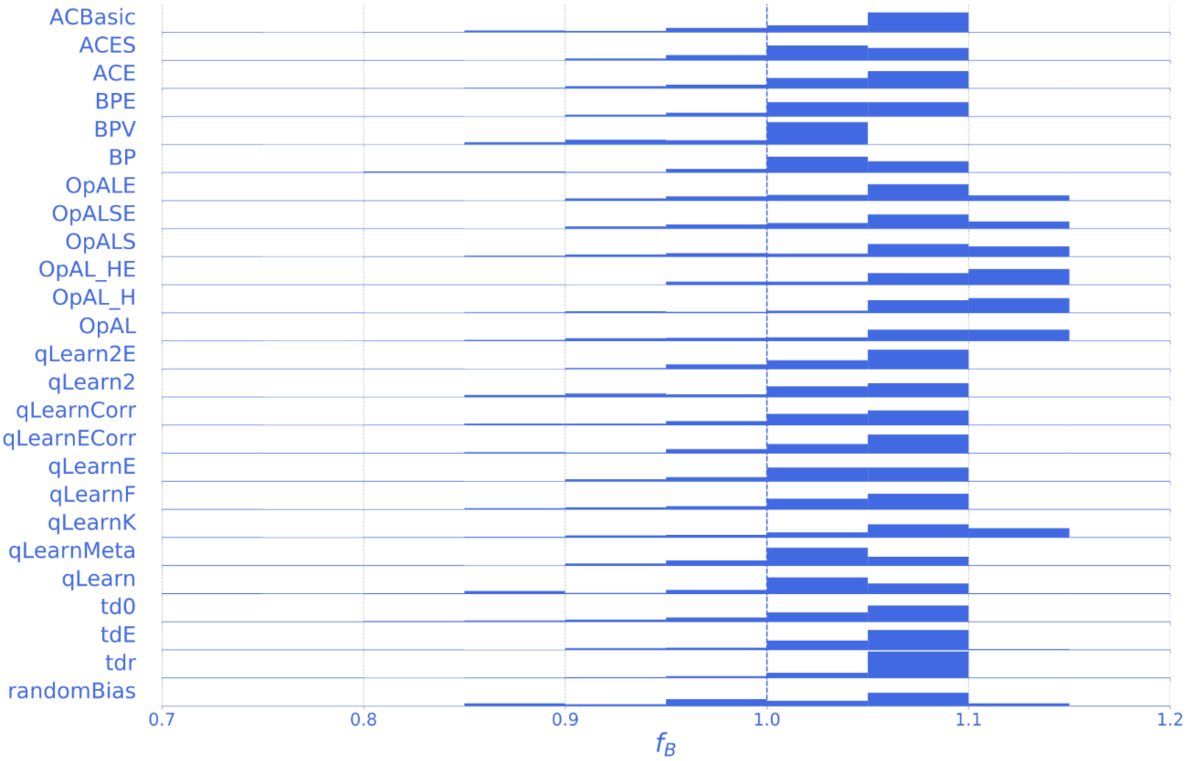


Figure 7-9 The distribution of the values of the fit quality normalised Bayes factor from fitting the Probabilistic Selection dataset when compared to a pure random model. Fits with values below 1 have a Bayes factor of over 20, so have strong evidence that the model can match the participant's actions better than a pure random model.

fitting 26 participants. As this task asks participants to learn a preference for some characters over others, the randomBias model could be expected to fit participant actions better than a pure random model. However, this does not appear to be the case for most participants, with only 27 participant fits having a Bayes factor above 20, the most for any model. Models on average had around 15 participant fits with a Bayes factor above 20, with six participants having strong evidence for at least 20 of the 25 models and two of the participants for all 25.

The Bayes factor values for each participant's model fits can be compared between models, as shown in Figure 7-11. In this figure, identical Bayes factors for the same participant are located along the diagonal line in each small plot. This is the case for almost all participants when comparing, for example, the OpAL and the OpALS models, as shown more clearly in Figure 7-10. If one model fits a participant consistently better than another model, then the participant will be shown away from the diagonal line. For example, the qLearn model fits have stronger evidence than those of OpAL_H, as they are on the qLearn side of the diagonal line. Nevertheless, they are still related, as most of the dots form a line that is parallel to the diagonal line of equal Bayes factor. The further away a dot is from the diagonal line, the greater the difference in the Bayes factors of the two models for that

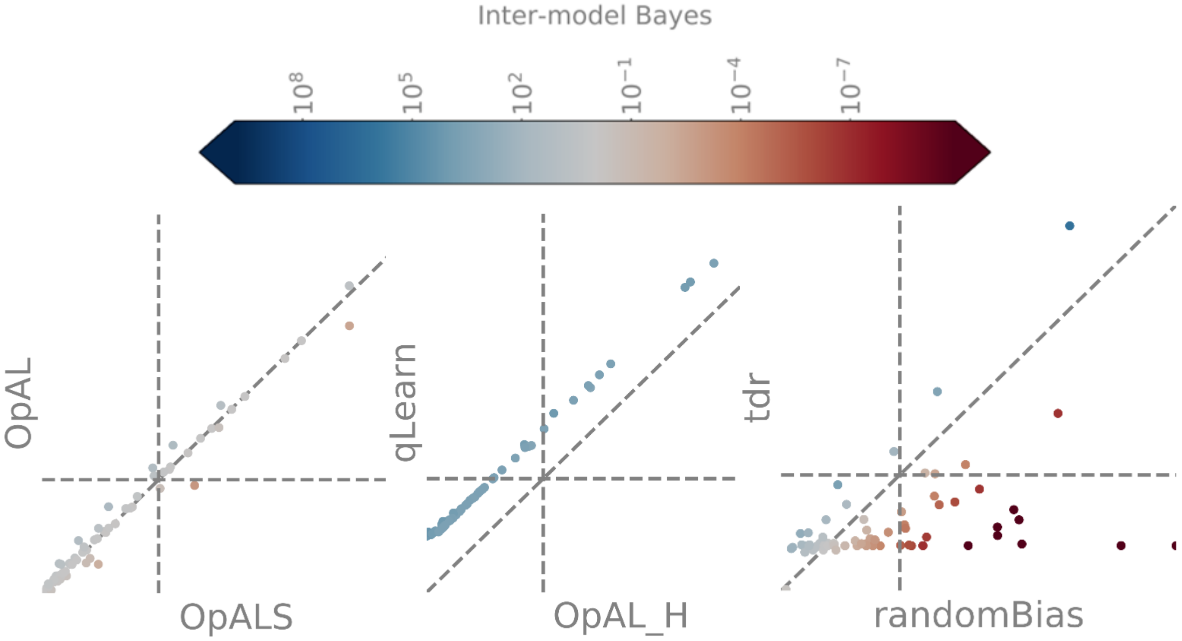


Figure 7-10 A selection of expanded miniplots from Figure 7-11

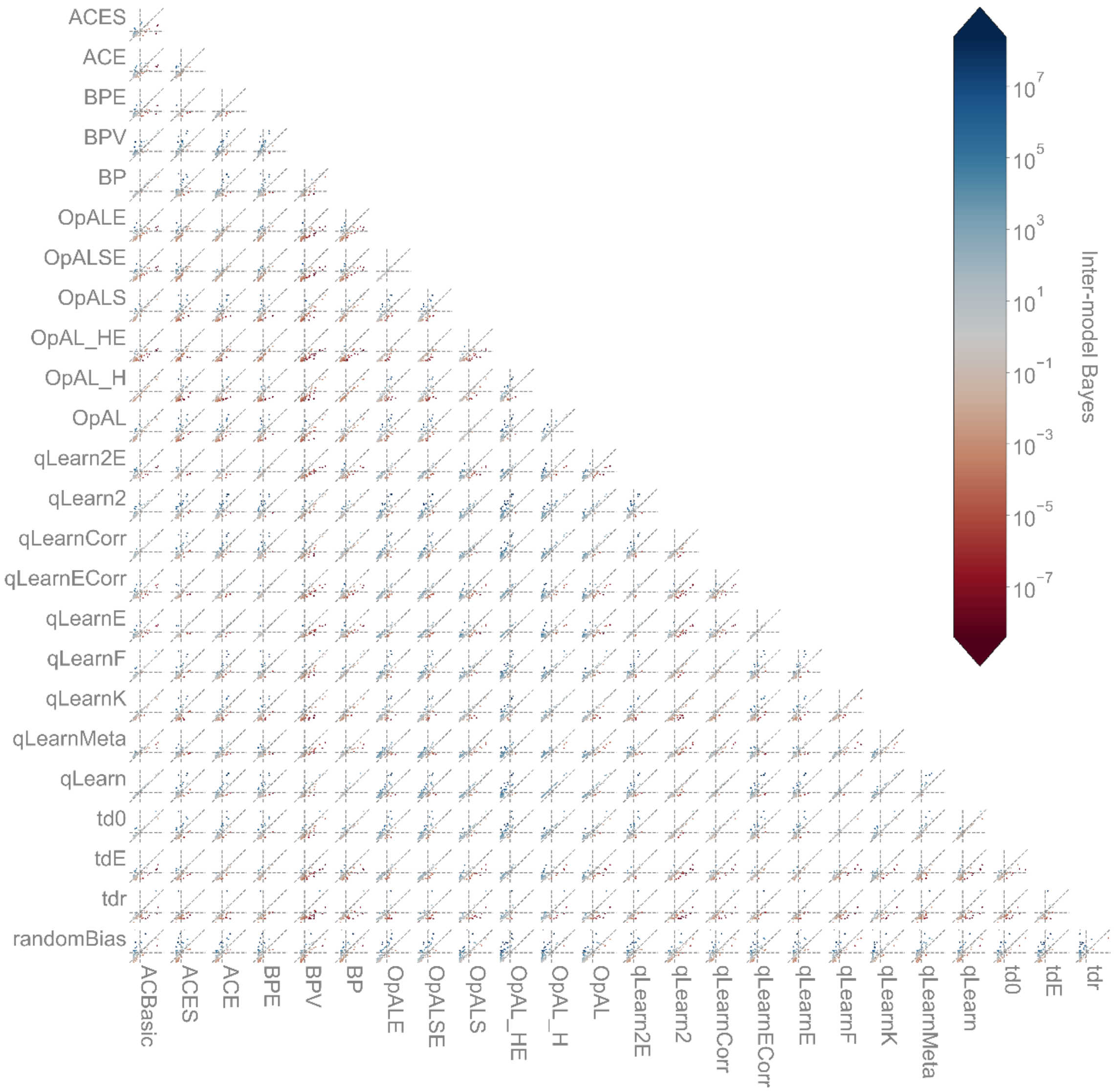


Figure 7-11 A comparison between models of the Bayes factor values for each participant when compared to a pure random model. Both the horizontal and vertical axes of each model comparison use a log scale ranging from 10^{-5} to 10^{18}. The horizontal and vertical lines denote a Bayes factor of 20 and the diagonal line follows the line of equal value for both axes. The dots are coloured with the inter-model Bayes factor, such that a Bayes factor of 20 signifies that there is strong evidence that the vertical model fits better the participant's actions than the horizontal model.

participant data. The horizontal and vertical lines mark a Bayes factor of 20 for the fit of the model axis they intersect with, i.e. the vertical axis model fit values are associated with the horizontal dashed line. Therefore, for most participants, the Bayes factor values for the tdr model relative to a pure random model are lower than 20, whereas the Bayes factor values of the randomBias model have a broad range of values relative to the pure random model. This results in the dots in their comparative plot not only being mostly on the randomBias side of the diagonal line, but also beyond the vertical dashed line.

The relative difference between model fits be expressed as a between-model Bayes value, calculated using the model fit BIC values by taking inspiration from equations 2.4 and 2.8 in chapter 2.2, so that:

$$\mathcal{B} = 2^{\frac{\mathrm{BIC}_{model\,1} - \mathrm{BIC}_{model\,2}}{2}}$$

These are shown in Figure 7-11 by the colour of each participant's dot, with the vertical model as model 2 and the horizontal model as model 1. Therefore, the participant dots with a Bayes factor of 20 or higher, coloured blue, have strong evidence that the vertical model is a better fit for the participant's actions than the horizontal model. Conversely, with a Bayes factor of 1/20 or lower, coloured red, there is strong evidence that the horizontal axis model is a better fit for the participant's actions than the vertical axis model. The median values for each inter-model comparison are shown in Figure 7-12. This highlights the poor performance of the OpAL models, especially OpAL_H and OpAL_HE, and qLearnK to fit participant data on this task. BPV can be seen to perform better than all the other models, with BP performing better than or equal to the remaining models. There are occasions where Figure 7-12 can be misleading, such as when it suggests that there is somewhat strong evidence for qLearnECorr to fit the participants better than OpAL_H, with a median inter-model Bayes factor of 113, whereas Figure 7-11 shows that there is strong evidence that qLearnECorr and OpAL_H each fit different participants well. Equally randomBias has strong evidence for fitting some participants better than BP or BPV, but for other participants, the opposite is true. These are shown in a larger form in Figure 7-13.

Another approach to comparing the model's performance is to examine the expectation of the model frequencies, EF, as discussed in chapter 2.4. This assess the relative frequencies with which two models could have generated participant data in the dataset. The probability that this relative frequency is above chance is estimated using the protected exceedance probability (Rigoux et al., 2014). Both the expectation of the model frequencies and the related protected exceedance probability were calculated using the VBA toolbox (Daunizeau et al., 2014). This used as inputs the BIC values calculated for each model's fit to each participant's task action sequence. These comparisons can be seen in Figure 7-14. This reinforces our previous conclusion that the BPV model has the strongest evidence, but also marks qLearn as being more successful than was previously apparent.

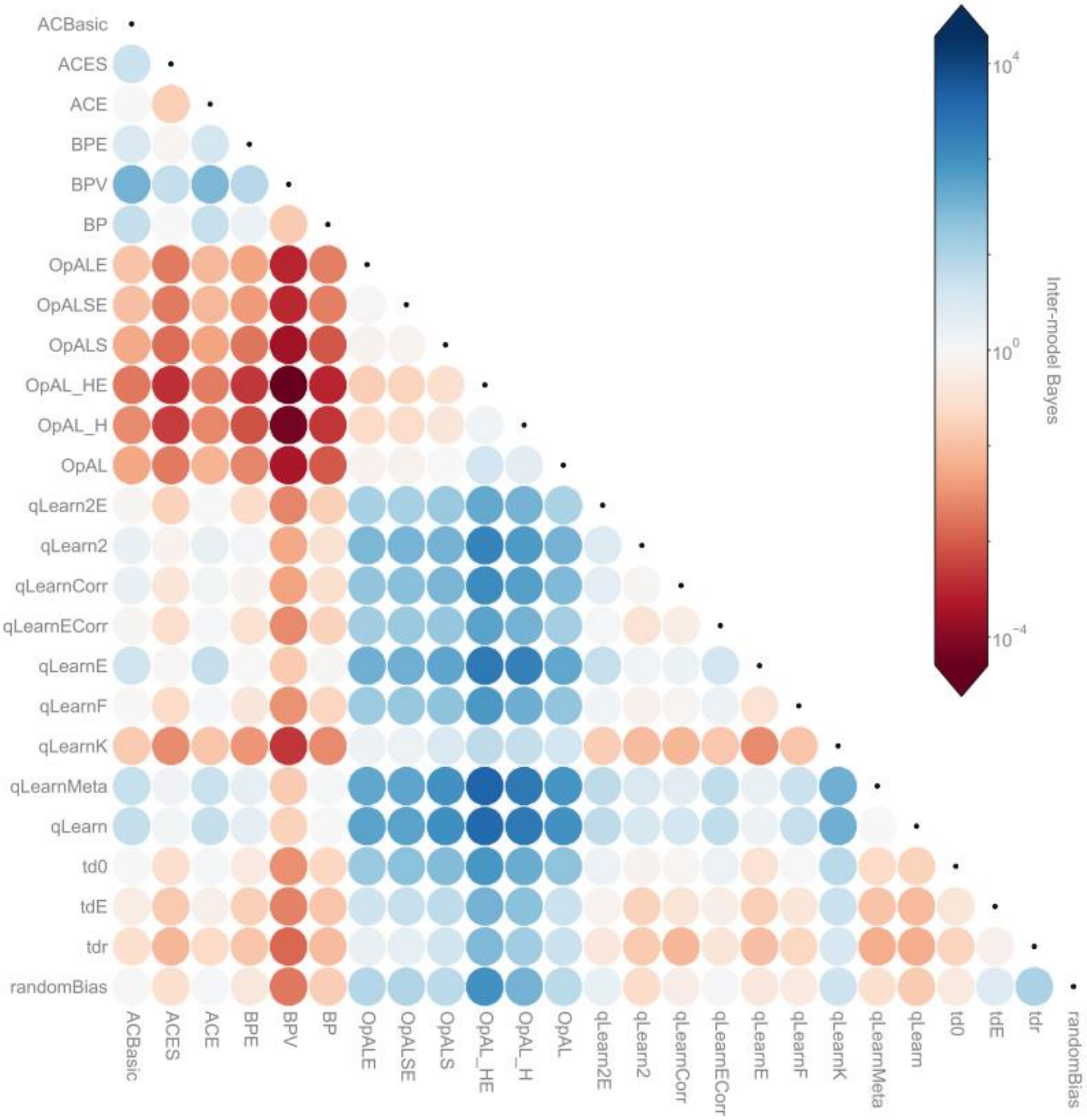


Figure 7-12 The median inter-model Bayes values for the Probabilistic Selection dataset participant fits. High values signify that the model on the vertical axis had a lower BIC value, and so a better match of the participant's actions, than the horizontal axis model.

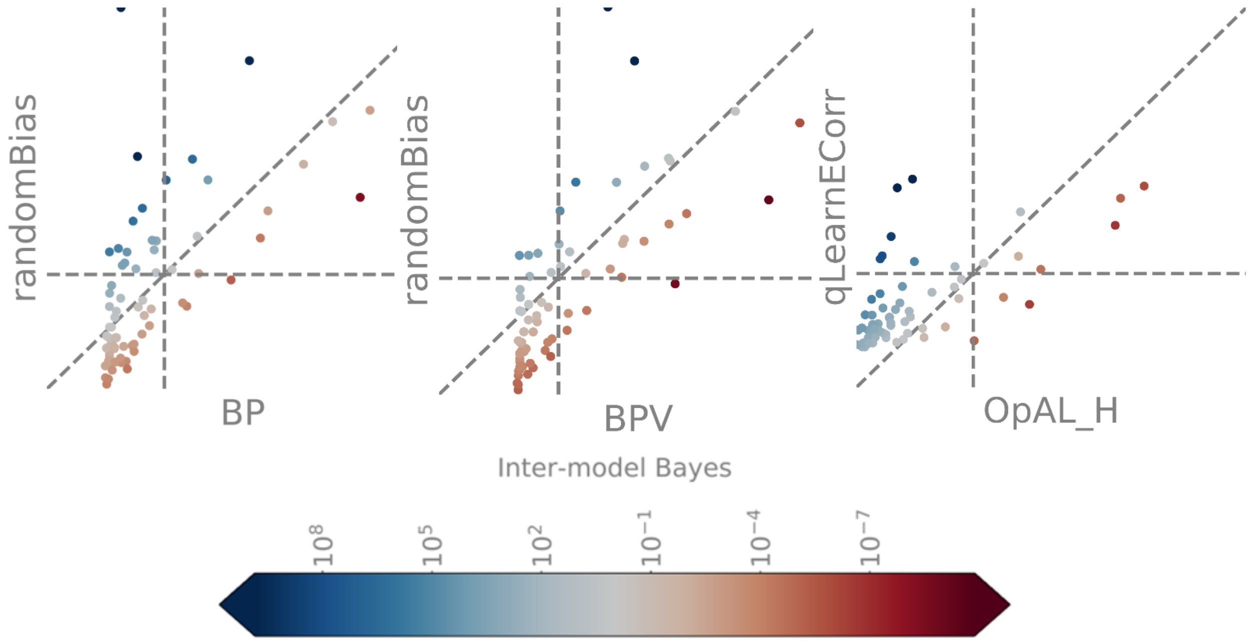


Figure 7-13 A selection of expanded miniplots from Figure 7-11

Comparing the test phase participant action choice measure, choose A and avoid B with the Bayes factors of model fits relative to a pure random model, no significant correlations were found with avoid B after Bonferroni correcting. However, many were found with choose A, show in Figure 7-15. This suggests that there are strong correlations between the participant's understanding of the information given to them during the task, measured through choose A, and the capacity of models to fit a participant's actions better than the pure random model. This, in spite of both poor participant performance at the task and poor model fits, indicates that participant actions can, to some degree, be captured by these models when the participant is learning during the task.

Similar correlations were performed between the model fit Bayes factors and extraversion, but no correlations were found, even when ignoring any Bonferroni corrections.

When removing all the edge fits found in chapter 7.2.1, the distributions of fit qualities do not change significantly, suggesting that model fits that result in parameters recovered at a boundary do not have particularly higher or lower fit quality values than others.

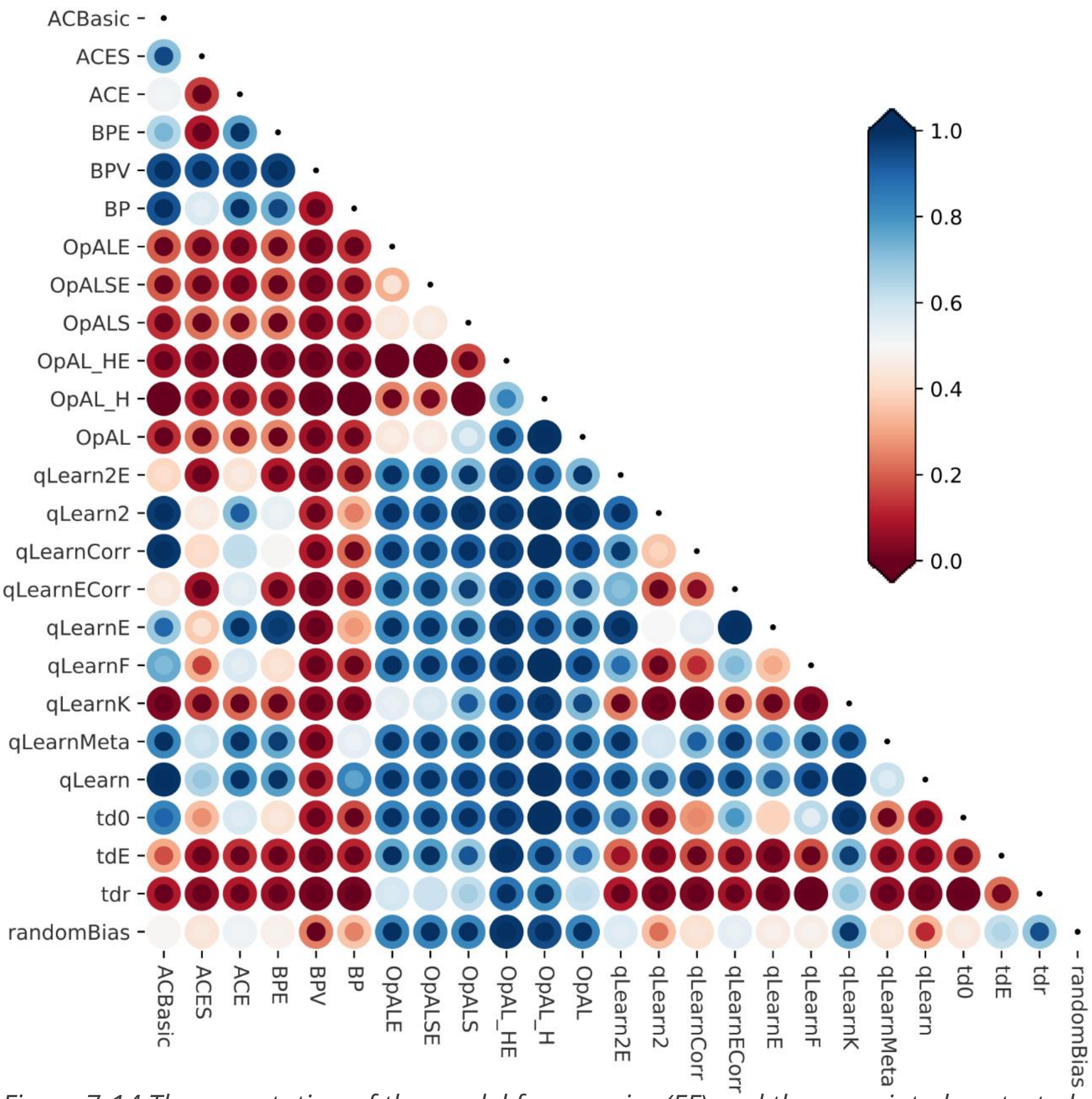


Figure 7-14 The expectation of the model frequencies (EF) and the associated protected exceedance probabilities (pEP) for model pairs. Each pair of circles shows the EF and pEP for the vertical axis model relative to that of the horizontal axis model. The outer, larger circle is the EF and the inner circle is the pEP. Both are scaled between [0,1].

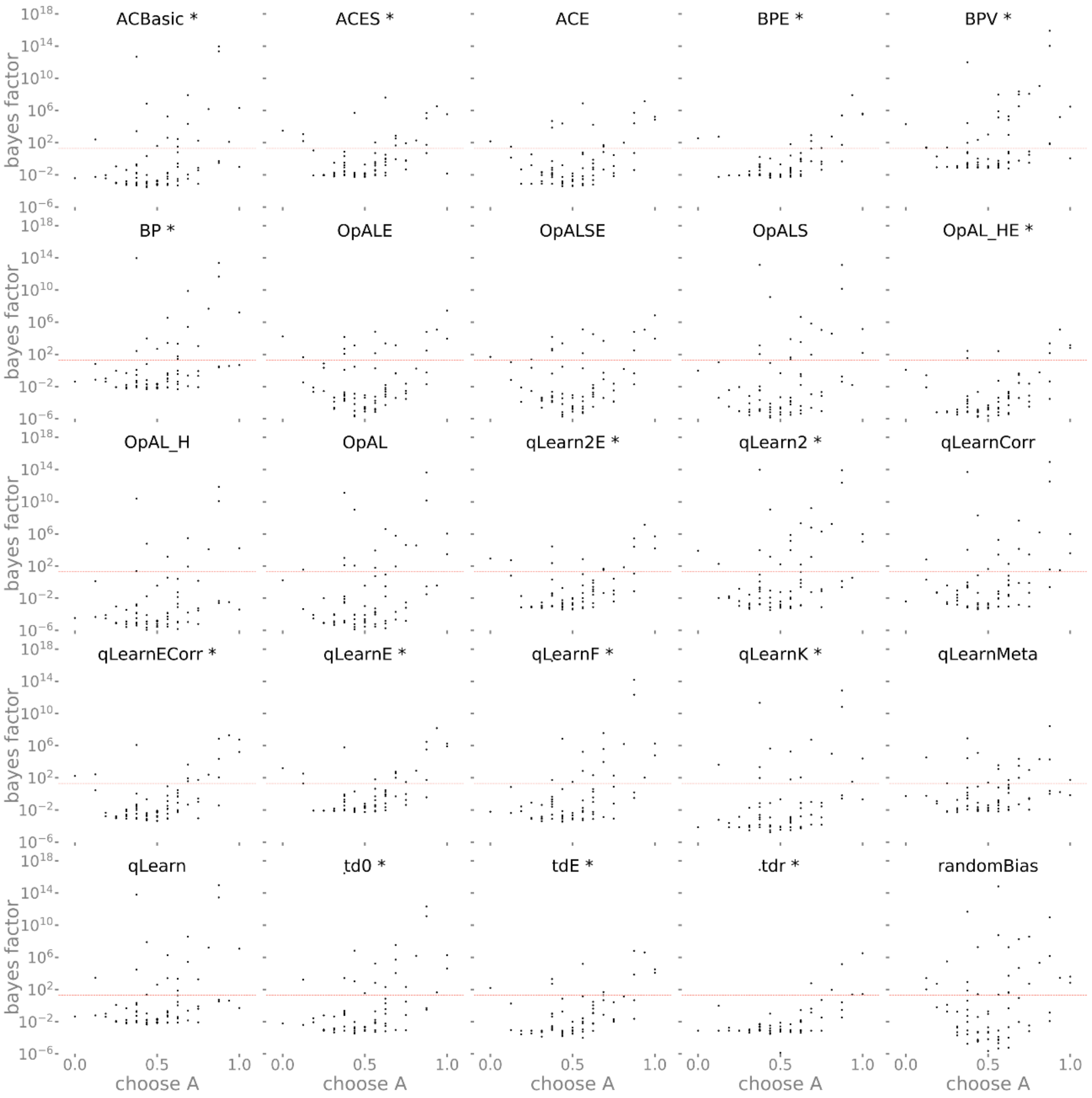


Figure 7-15 Participant test phase proportion of choosing character A compared to the model fit Bayes factor when compared to a pure random model. The star denotes a Bonferroni corrected Spearman's rank correlation with a p <0.05

To provide an estimate of the number of participant fits that may be considered to have good fits, we can use the proportion of participants with a fit that has a Bayes factor of 20 or more, and with recovered parameters not on the edge of the parameter boundaries. The proportion of participants with not good fits, for each model, can be seen in Figure 7-16. While all of these models had less than half of their participants with good fits, the best performing models are randomBias and BPV, which contrasts with the Decks task, where the BP models performed the worst.

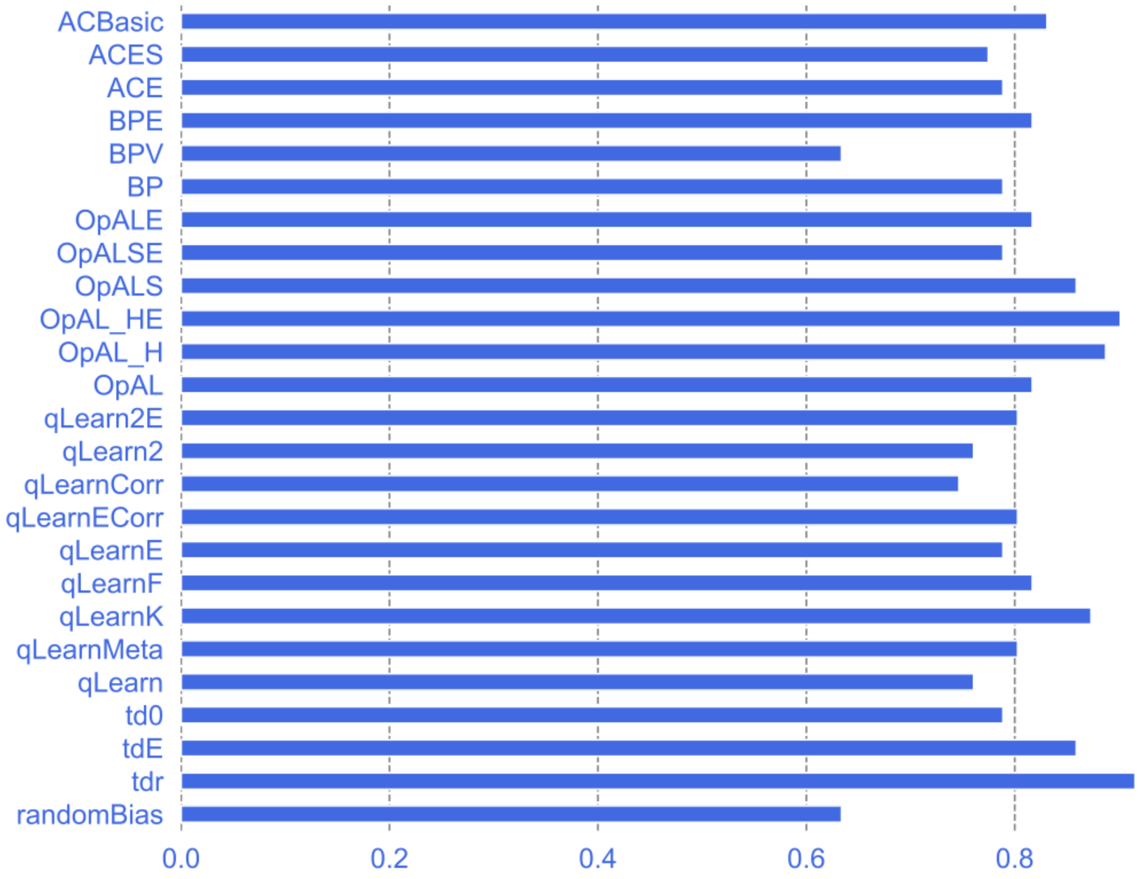


Figure 7-16 The proportion of the 71 fits where at least one recovered parameter was within 0.1% of its boundary and the Bayes factor of the fit was below 20 when compared to the pure random model.

7.2.3 Parameter correlations

If the recovered model parameters are identifying a feature of a participant's learning and decision-making process then we would expect that in different models the same parameter, performing the same task, would have very similar values. The correlations between parameters across models should therefore be high for parameters performing the same role in different models and low between parameters performing different roles, especially those in the same model. In this section, the correlations of the three most common parameters are examined: α, β and ε. A plot of the full comparison between model parameters can be seen in Appendix II. The correlations for the learning rate parameter, α, shown in Figure 7-17, need to be broken down further, as learning rates are used for learning various estimators. Contrasting these with those in the Decks task, Figure 6-18, the correlations are overall weaker.

One subgroup of α parameters is those of the Q-learning class of models, with only one learning rate parameter and without separate actor and critic components, as described in chapters 3.2, 3.5, 0 and 4.8. Collectively, their correlations result in a Kendall's W, a measure of collective concordance of 0.31, which suggests a weak correlation, half the strength of that found in the Decks task, 0.642 (Figure 6-19). Looking at the individual model parameter pair correlations, shown in Figure 7-18, it can be seen that the correlations are mostly positive, with some weakly negative correlations. However, the strong correlations are all positive. The correlations between the critic α parameters, also shown in Figure 7-18, show a similar pattern, albeit with more strong correlations.

As with α, correlations could be expected for the α^+ and α^- parameters found in the OpAL models and qLearn2 variants. However, as can be seen in Figure 7-19, there are no correlations within these groups parameters, with Kendall's W values of 0.20 for the α^+ and 0.16 for the α^-. This is surprising, given the similarity of the OpAL models and equally surprising for the qLearn2 models. However, it is notable that the same strong correlations have been found between the OpAL and OpALS α^+ and α^- in both this task dataset and that of the Decks task. Once again, the correlations between these parameters within each model are also quite low in the case of the qLearn2 models (0.07 for qLearn2 and 0.03 for qLearn2E) and for the OpAL models (values ranging from -0.2 and 0.1), suggesting that these weak correlations are not due to the difficulty in teasing apart the influence of the α^+ and α^- parameters.

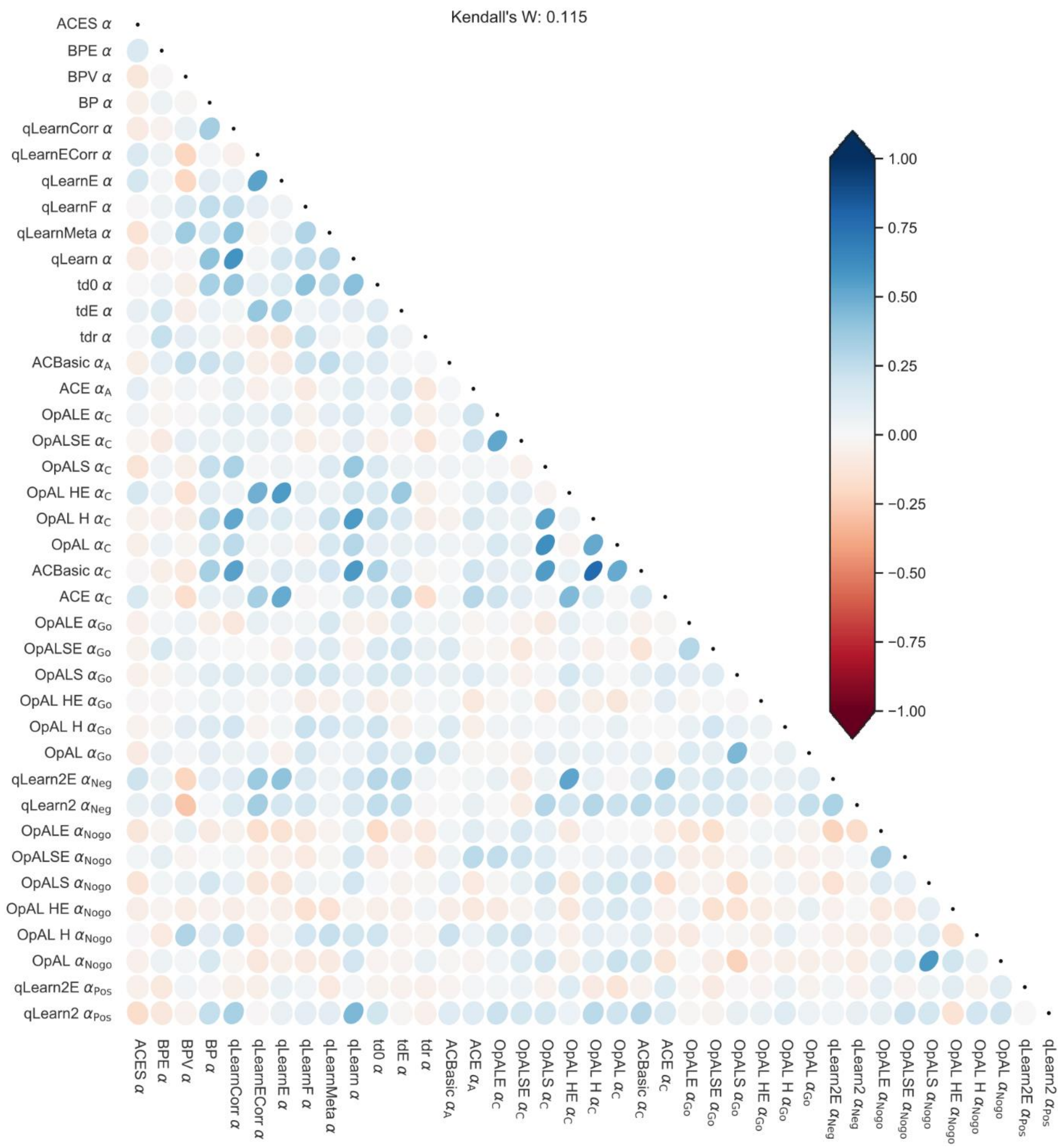


Figure 7-17 The correlations between recovered α parameter values from the Probabilistic stimulus task participants. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation.

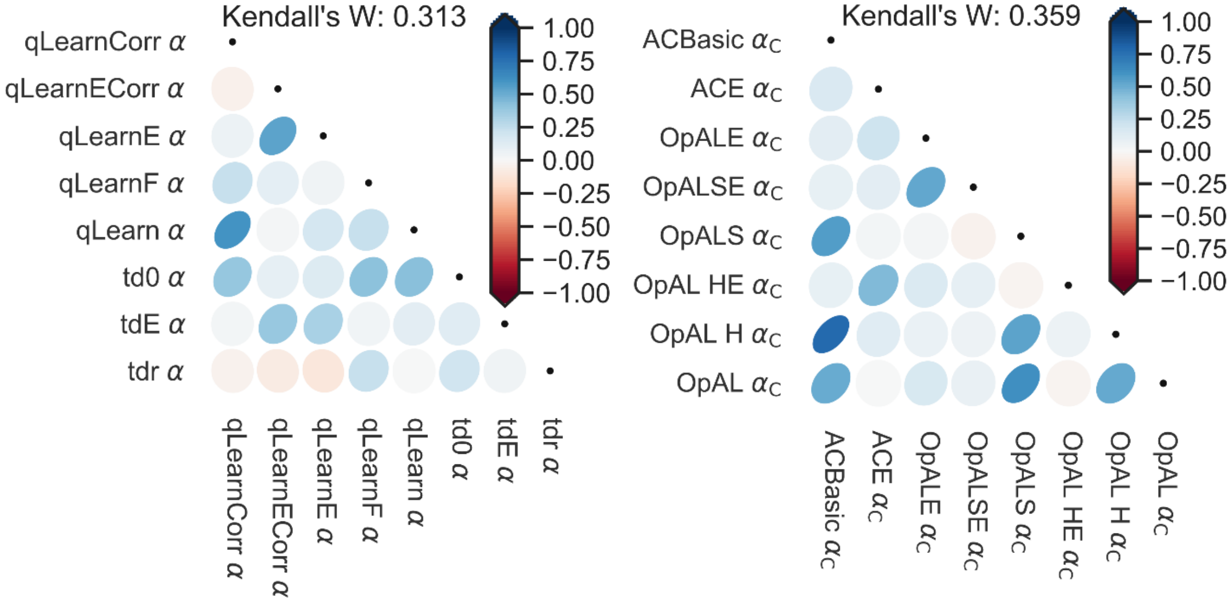


Figure 7-18 The correlations between recovered parameter values from the Probabilistic stimulus task participants. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation. ***Left:*** *The critic learning rate parameters, α, in the critic only q-learning models* ***Right:*** *The critic learning rates from models with both actor and critic learning rates.*

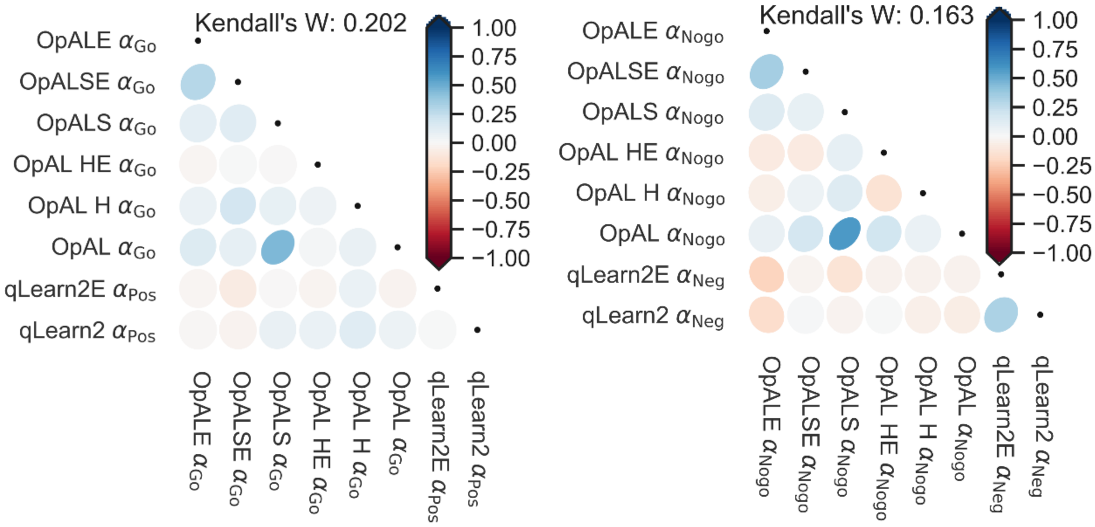


Figure 7-19 The correlations between recovered parameter values from the Probabilistic stimulus task participants. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation. ***Left:*** *The learning rate for positive rewards* ***Right:*** *The learning rate for negative rewards.*

The same analysis was performed for the β and ε parameters, as shown in Figure 7-20 and Figure 7-21. The recovered ε parameters have strong correlations, as shown by the Kendall's W value across the models of 0.78. This contrasts with the large number of weak correlations for β, with a Kendall's W of 0.33, acting as further indications that the ε-greedy function provides more consistent parameter recovery than that of SoftMax. The only β parameters that are strongly correlated to each other are those for the Q-learning class of models shown in Figure 7-20, with a combined Kendall's W of 0.61, dragged down by the consistently weak correlations of the β parameter in tdr with the other model β parameters.

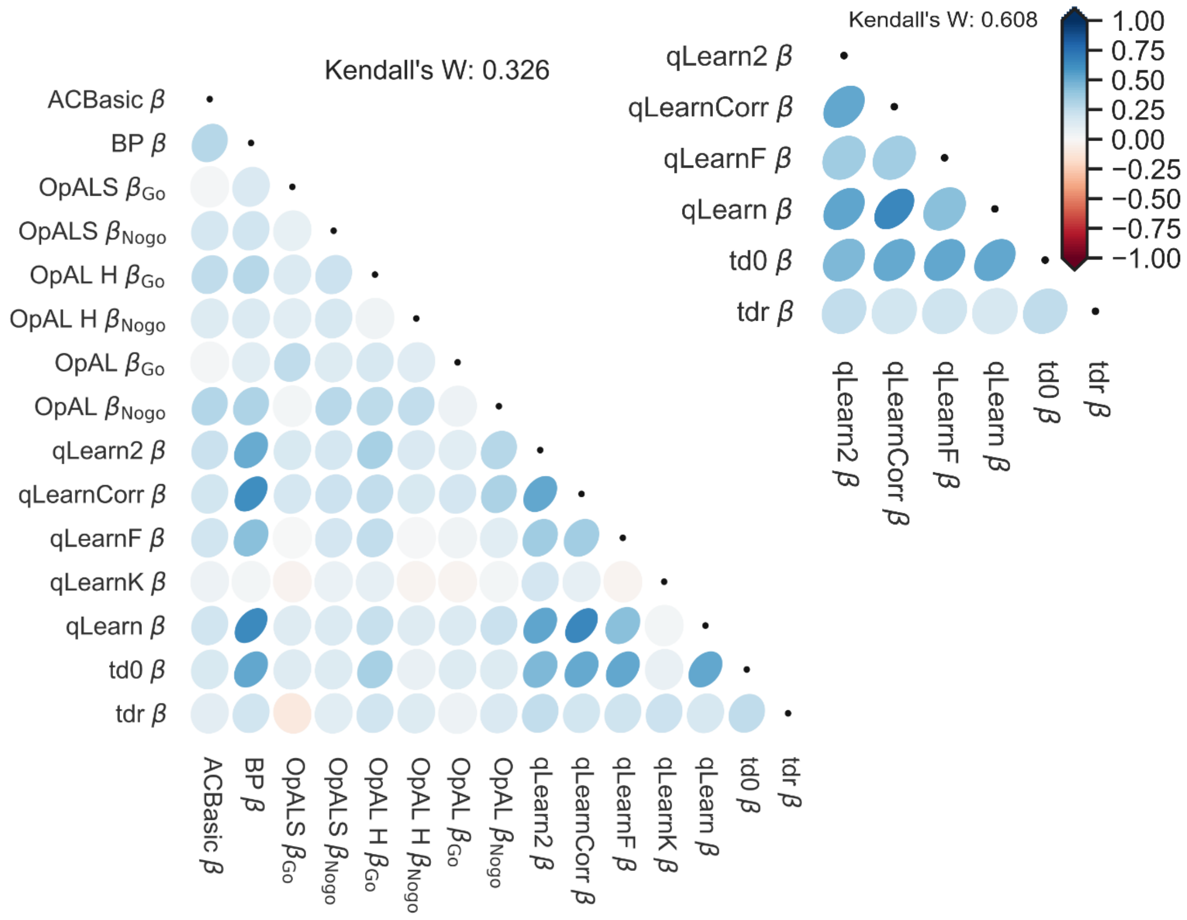


Figure 7-20 The correlations between recovered parameter values from the Probabilistic stimulus task participants. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation. ***Top right:*** *The β parameters for the Q-learning models* ***Bottom left:*** *The β parameters in all models.*

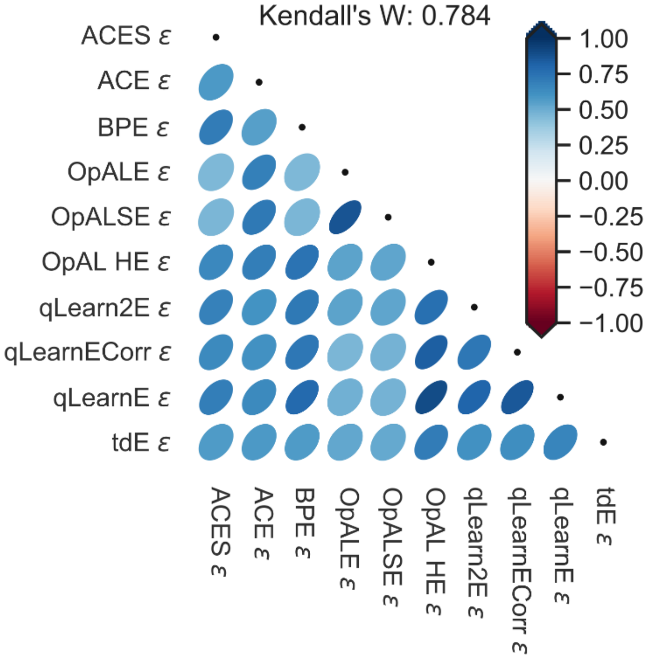


Figure 7-21 The correlations between recovered parameter values from the Probabilistic stimulus task participants for models with an ε parameter. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation.

The weakness in correlations between parameters that perform essentially the same function in different models might be explained by differences in the capacity of each model to match the performance of each participant. A model that struggles to explain the actions of most participants will have less difference between fit quality measure of different parameter combinations, resulting in more noise in the final fits.

Another potential source of noise may stem from differences in model complexities resulting in varying difficulties in fitting models. Models with more parameters or more layers, i.e. more degrees of freedom, will require more information to fit as accurately as simpler models. This could explain the poor performance of the OpAL and qLearnK models that have the most parameters. It could also explain the surprisingly poor performance of the biased random model, relative to the pure random model, as for this task the biased random model had six parameters, one for each character.

One confounding factor in describing a models degrees of freedom is that recovered model parameters are known to be correlated (Schmiedek et al., 2007).

In chapter 4, the distribution of errors in parameter recovery for the qLearn model parameters α and β suggested that the recovery process resulted in the parameters being inversely correlated. For this task dataset, the correlation was found to be -0.42, matching previous results. By contrast, for qLearnE, the correlation between α and ε was found to be 0.22, which adds evidence to ε-greedy being more recoverable than the SoftMax parameter β. However, models with large numbers of parameters. Such as randomBias, shown in Figure 7-22, do not show strong correlations between any of the parameters, suggesting that these correlations have the potential to be more pronounced in models with only two or three parameters.

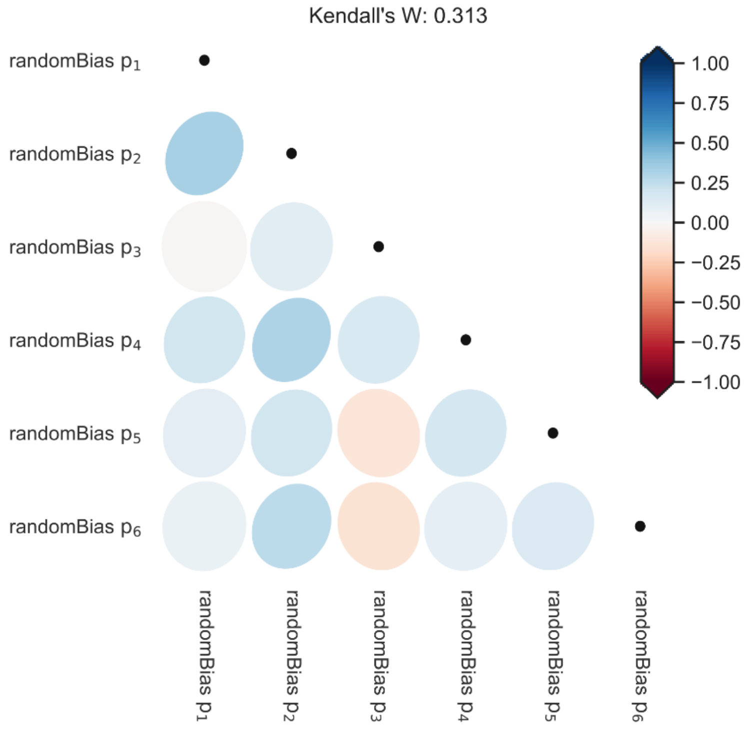


Figure 7-22 The correlations between recovered parameter values from the Probabilistic stimulus task participants for the model randomBias. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation.

7.3 Model parameter relationships to extraversion

Having found correlations between model fits and choose A, but not between model fits and extraversion, an examination was made of a possible relationship between the recovered α parameter and a measure of the extraversion phenotype. Such a correlation was found with the Decks task datasets, in chapter 6.3. As the extraversion measure of EPQ-R was also recorded for this dataset, the same analysis could be performed here as had been for the Decks dataset, comparing the extraversion measures for the participants to the fitted parameter values for Q-learning class of models, with only one learning rate parameter and without separate actor and critic components, shown in Figure 7-18 (left). These are: qLearnCorr, qLearnECorr, qLearnE, qLearnF, qLearn, td0, tdE and tdr. The mean of the α values was calculated for each participant from those recovered from the Q-learning class of models. A Spearman's rank correlation was performed between the mean α value for each participant and the participant's measured extraversion value of EPQ-R.

To remove some of the noise from this correlation, those fits that did not have strong evidence of being better than a pure random model or who had one or more parameter close to their parameter boundaries were removed before calculating the mean α values for each participant.

To reduce the potential of any intra-model correlations between α and β or ε affecting the correlations, separate means were also calculated for the subset of models with a β parameter and for those with an ε parameter. This split is especially important as the correlations between α and β, and α and ε are in the opposite directions, negative for α and β and positive for α and ε. Estimates of those models mean β and ε parameters were also calculated for each participant. Partial correlations were performed between the extraversion measure and each of these α parameter estimates.

In total this resulted in six correlations between participant estimates of α and the extraversion measure of EPQ-R, shown in Table 7-3. These show a negative correlation between α and extraversion for the well recovered α values and positive correlations when all recovered values are used. The α values calculated from models with β and only good model fits was the only one to have a significant Bonferroni-corrected correlation, as was the case for the Decks task dataset.

Participant model fit parameters used	*All*		*Good edge & Bayes*	
	ρ (DF)	*p*	*ρ (DF)*	*p*
Q-learning models	0.071 (71)	0.558	-0.060 (24)	0.781
Q-learning models with β	0.143 (68)	0.237	-0.703 (18)	0.001
Q-learning models with ε	0.024 (68)	0.845	-0.145 (14)	0.592

Table 7-3 The correlations between each participant's averaged Q-learning model parameter α and the extraversion measure of EPQ-R. The α, β and ε values in these correlations are averaged, for each participant, across the relevant subset of models. The β and ε subset correlations were partial correlations whereas the others were simple correlations.

7.4 Discussion

The Probabilistic Selection task tests participant's capacity to apply an understanding of reward likelihoods from pairs of characters to novel pairs of those characters. The 71 students were found to have not been given sufficient trials to fully learn the reward likelihoods in the initial pairings before being shown the new pairings. This can be seen in participant's weaker performance at choose A and avoid B than in published studies with this task, where participants performed more learning phase trials. Extraversion, as measured by the EPQ-R, was not found to correlate with choose A, avoid B.

The models were evaluated for their performance in producing the same action choices as those of the participants, with the baseline for their fit quality being set as the performance of the pure random model. The model fits were performed on all the action choice trials performed in the learning and test phases by each participant. None of the models provided significantly better fits for all the participants than the pure random model. The model fit accuracy varied by type of model, with the randomBias and Bayesian inspired models providing some of the best fits, notably BPV. The OpAL models provided the worst fits.

Correlations were found for most models between fit quality and choose A, suggesting that were the participants to be given more trials, which would be expected to improve choose A performance, it is likely that many of these models would fit the participant’s actions better than the pure random model.

α parameters were inconsistently recovered with most being weakly correlated. The ε-greedy parameters were found to be recovered quite consistently across all the models and much more consistently than the β parameter from equivalent models using softmax. The strength of correlations between β parameters was found to be related to the type of model. The most consistently recovered β parameters were those of the Bayesian and Q-learning variants.

A strong correlation was found for the participants who were well fitted by at least one Q-learning model using softmax and their EPQ-R extraversion measure. Although this correlation was with a very small sample of participants, N=18, this was the same correlation and direction as was found in the larger sample of participants with the Decks task, N=116, further suggesting that this correlation between α and extraversion might be a real effect.

It is worth noting that these results reflect the performance of the models only as implemented. There may be other implementations of the same models that perform better or worse with slight tweaks to their implementation, to their starting parameter values or their parameter upper bounds.

8 WEATHER TASK

The Weather task is a category learning task based on one described by Gluck & Bower (1988) and later adapted by Knowlton, Squire, & Gluck (1994). It asks participants to associate a series of cues with one of two outcomes. One to three cue cards, from a set of four cards, are presented to the participant in each trial. The participant must decide which one of the two possible outcomes the displayed cards are most likely associated with. Once the participant decides, they are then told if they were correct or not. The cues each have a probabilistic relationship with the two outcomes, with this this version of the task having novel probabilistic relationship, with the probability of an outcome varying depending on the combination of cues displayed, as described in Table 8-1. For example, if the first two cues are displayed, then the first outcome is guaranteed. If only one of them is displayed, then the first outcome will be the correct one 75% of the time. Across the whole task, the first two cues having a 64% chance of being associated with the first outcome and the second two having the inverse.

In the first phase of the task, the *learning phase*, participants are given feedback on if their choice was correct. In the second phase, the *testing phase*, participants are not given any feedback. The sequence of cues and the outcomes were fixed beforehand and are shown in Figure 8-2. The learning phase contains four examples of each of 14 possible cue pairs, totalling 56 trials. The test phase contains one example of each of the 14 possible cue pairs.

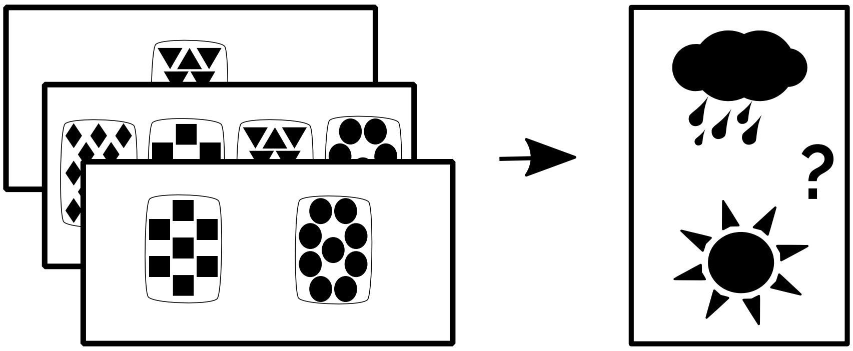


Figure 8-1 The Weather task consists of a series of trials where one to three cue cards, from a set of four cards, are presented to the participant. The participant must decide which of the two outcomes the cues are more likely to predict.

Cue group type	*Most likely outcome for each active cue combination*	*Probability of most likely outcome*
Same pair	1100 -> action 1 0011 -> action 2	1
Single	1000 -> action 1 0100 -> action 1 0010 -> action 2 0001 -> action 2	0.75
Triple cues	1110 -> action 1 1101 -> action 1 1011 -> action 2 0111 -> action 2	0.75
Opposing pair	1010 -> either action 1001 -> either action 0110 -> either action 0101 -> either action	0.5

Table 8-1 The probabilities of most likely outcome for each possible combination of cues. These are grouped by type of combination.

The cues and outcomes were presented in two different forms: in its traditional form as a Weather prediction task, shown in Figure 8-1, and as a disease prediction task. The Weather prediction task used abstract symbol cards for the prediction of sunshine and rain. The disease prediction form had participants predicting incidences of two fictitious diseases, Merlitis and Calditis, based on four symptoms or cues: skin rash, vomiting, fever and dry cough. There was no difference in cue sequence between the two forms of the study. Participants did not have time limits on trial responses.

The datasets were collected using a computer program created using Microsoft DOS for the pilot and then recreated in MATLAB for the other datasets. Participant’s pressed the “c” or “m” keys to express their choice of outcome in trial. During the learning phase participants were rewarded for correct predictions with “Well Done!” printed on the screen and a monetary reward, which varied between studies. During the test phase participants did not receive feedback but did continue to receive the monetary reward without it being displayed to them. It was then delivered to them at the end of the task.

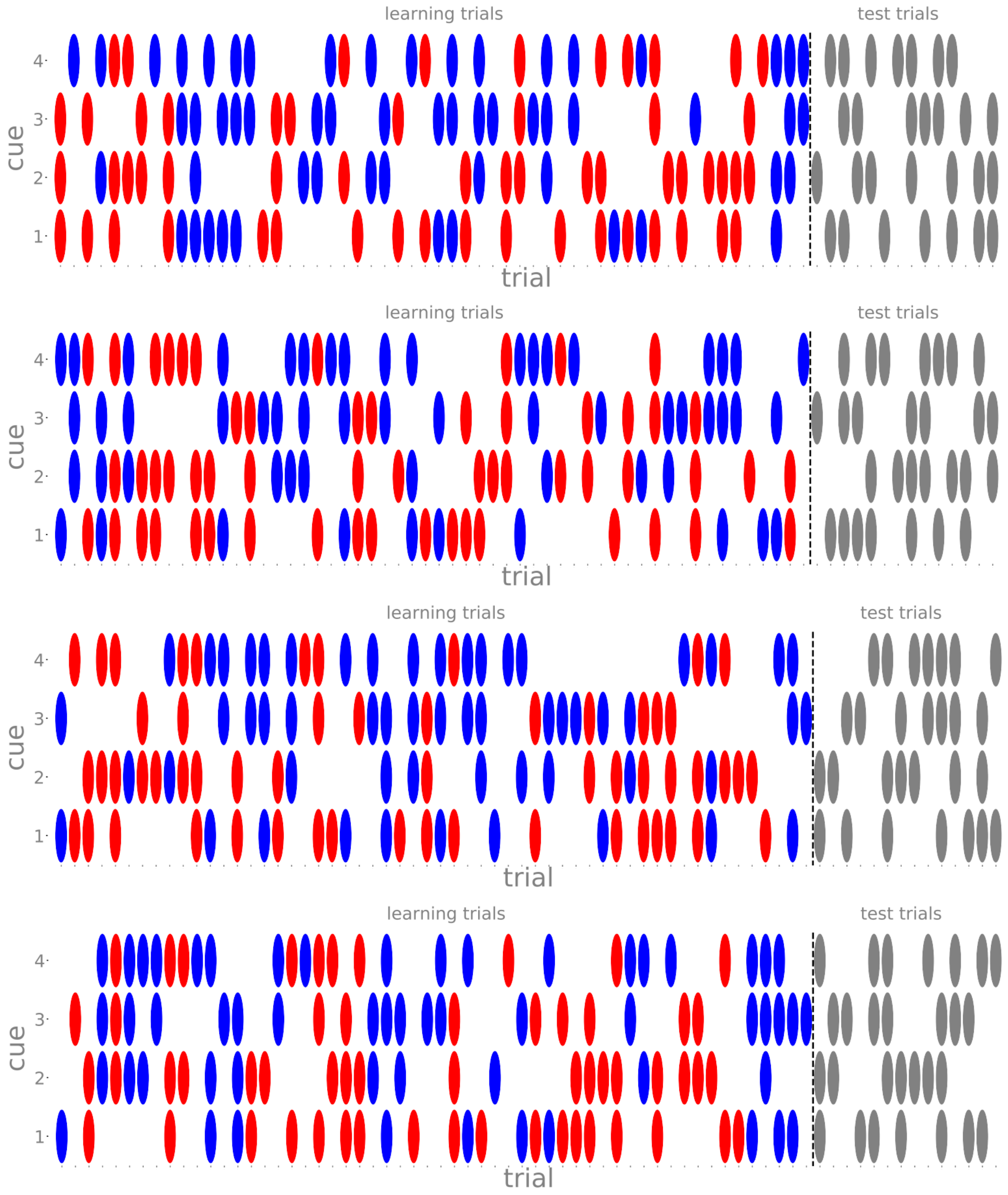


Figure 8-2 The sequence of cues shown to the participants during each trial. From top to bottom these were used in the Australian dataset, the Goldsmiths dataset and the two sequences used in the pilot. The ovals indicate which cues were visible for each of the 56 learning trials and 14 test trials. The colour of the cue ovals in the feedback block denote which of the two weathers/diseases were predicted by the cues in that trial. The black dotted line marks the transition from learning trials to test trials.

8.1 Data properties

For this task, three sets of participant data were available. The first dataset was a pilot study performed in 1999 at St. George's Hospital Medical School with 40 students performing an active version of the task, where participants are explicitly asked to provide their prediction for each trialstep. The results of this study are discussed further in (Pickering, 2004). For the subsequent two datasets, the participants performed both the active version of the task and a passive, observational version, where they observe during the learning phase and only respond during the testing phase. The first dataset, collected at Goldsmiths, university of London, recruited 71 participants from the undergraduate psychology students. The final dataset, collected at the University of Melbourne in Australia, recruited 124 participants from the local general public through social media platforms and notice boards. These were paid AU$15 for their participation. Only the active form of the task will be examined here.

Each correct response was rewarded with the feedback 'Well Done!' and with a monetary incentive of AU$0.10 in the Australian study or £0.1 in the Goldsmiths study and £0.02 in the pilot study. This was displayed on the trial feedback screen. While feedback was not provided in the testing phase, participants did continue to receive the same renumeration for the correct responses without it being displayed to them until the end of the test phase.

As the datasets were collected using the same version of the task, they were analysed together. In total there were 233 participants with ages from 18 to 59, with a distribution shown in Figure 8-3. 131 identified as female, 100 male and 2 other.

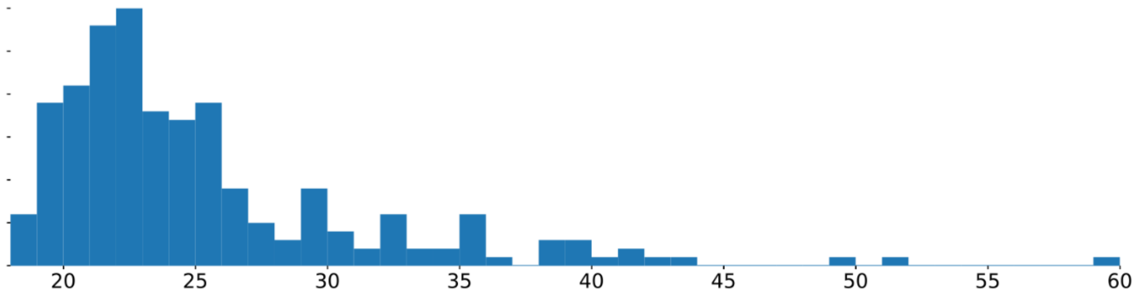


Figure 8-3 The distribution of participant ages for all three datasets.

One participant chose the same action for all but one of their action choices. This participant will therefore be excluded from the rest of this analysis.

The participants did not favour one option over another, with the distribution of average action choices across the datasets having a mean of 0.51, standard deviation of 0.08 and skew of 0.33. When grouped by type of cue group, as described in Table 8-1, participants can be seen, in Figure 8-4, to have mostly learnt the relationship between the cue combinations and the outcomes, as the same pair cue stimuli have more frequent best responses than the single cue stimuli or the triple cue stimuli, and the opposing pair cues were treated as unbiased random. Significant positive correlations were found between participant’s performance at learning the optimal responses for one type of cue group and another. Unsurprisingly, there were no correlations between the opposing cue pair group and any of the others. However, differences were found between the distributions of all these cue groups. Those of same pair cue group were higher than those of the single cue group t(413)=3.47, p=6.26e-4, the single cue group were higher than those of triple cue group t(464)=2.44, p=1.54e-2 and triple cue group were higher than those of opposing pair cue group t(446)=8.81, p=2.84e-16. It is notable that there was a difference between the single and triple cue groups, as their reward probabilities were identical. These learnt outcome relationships will be revisited when discussing the model fit qualities in chapter 8.2.2.

No significant correlations were found between extraversion as measured by the EPQ-R and a participant’s frequency of best responses for a given cue group.

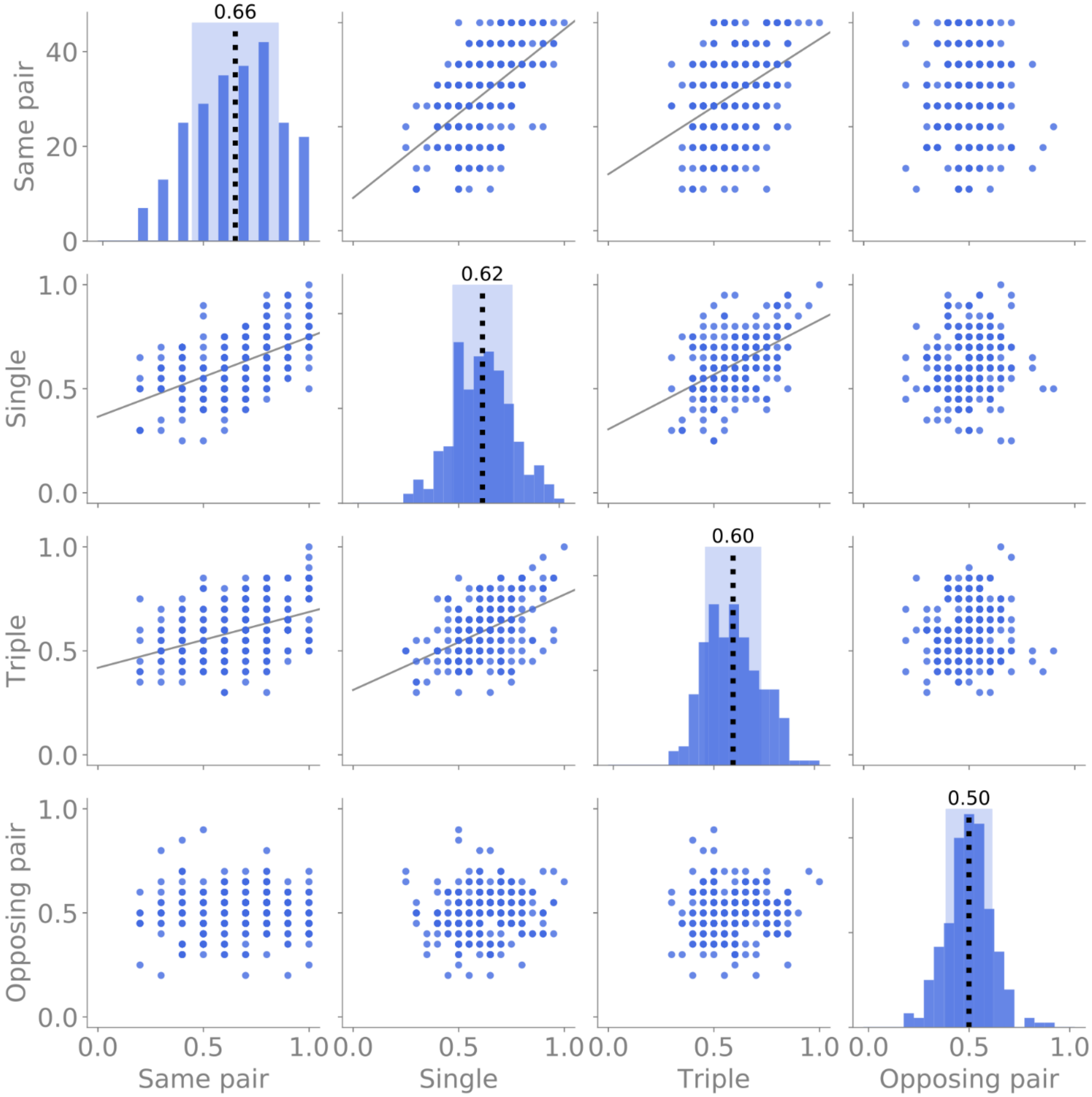


Figure 8-4 A pair plot of the proportion choices made by each participant that matches with the most likely outcome, for each type of cue combination, described in Table 8-1, in the trials with such a combination. The black dotted vertical line in the histograms denotes the mean and the pale blue bar surrounding it covers one standard deviation around the mean. The grey lines through the scatterplots are the linear regression lines, shown only for significant correlations (p < 0.001).

8.2 Fitting the models to the data

As the task rewards were either 0 or 1, within the models, the initial expected reward for both choices were set to 0.5. The upper bounds for infinitely bounded model parameters were truncated. The softmax β parameter was limited to 30, the Kalman model parameters σ_{α}^{2} and σ_{λ}^{2} were limited to 150 and the OpALS and OpALSE saturation parameters M were set to 49. The models were fitted to the participant's actions over both the learning and test phases.

During the fitting process, the models qLearnF, td0, tdr and OpAL and qLearnMeta experienced numerical overflows for certain parameter combinations, as discussed for OpAL in chapter 3.4. The parameter combinations where there has been an overflow are treated as worse than random fits, that have a probability of 0.5 for each action choice. For fits with overflows, a probability of 0.4 was returned for each action choice. If these provided a plateau of global minima, then the model could be discarded as being a worse representation of the participant's actions than a purely random model. tdE took too long to fit for certain participants and exceeded the maximum number of fitting iterations allowed. In these cases, the recovered parameter set is the best fitting one found so far.

8.2.1 Boundary recovered parameters

The crudest measure of acceptable model fits is to measure the number of them that intersect with any of the parameter validity boundaries. In almost all cases a boundary parameter value is equivalent to removing an element of a model, thereby reducing it to a simpler model. To account for numerical errors in the fitting, a boundary, or edge, fit is considered to have occurred if the parameter is within 0.1% of either side of its range. This was chosen to be close enough to the boundaries to be unlikely to interfere with good parameter recovery, while still accounting for the approximate nature of numerical fitting.

The results from fitting the Weather task datasets, in Figure 8-5, show the proportion of participants with recovered parameters on the edge of the parameter boundaries. The number of participants whose fits contained boundary parameter values varied across the different models from 1% to 33% of the sample, with the epsilon-greedy models having fewer, no more than 9%. There was no discernible pattern linking the number of edge fits and the type of learning in the model or the number of parameters.

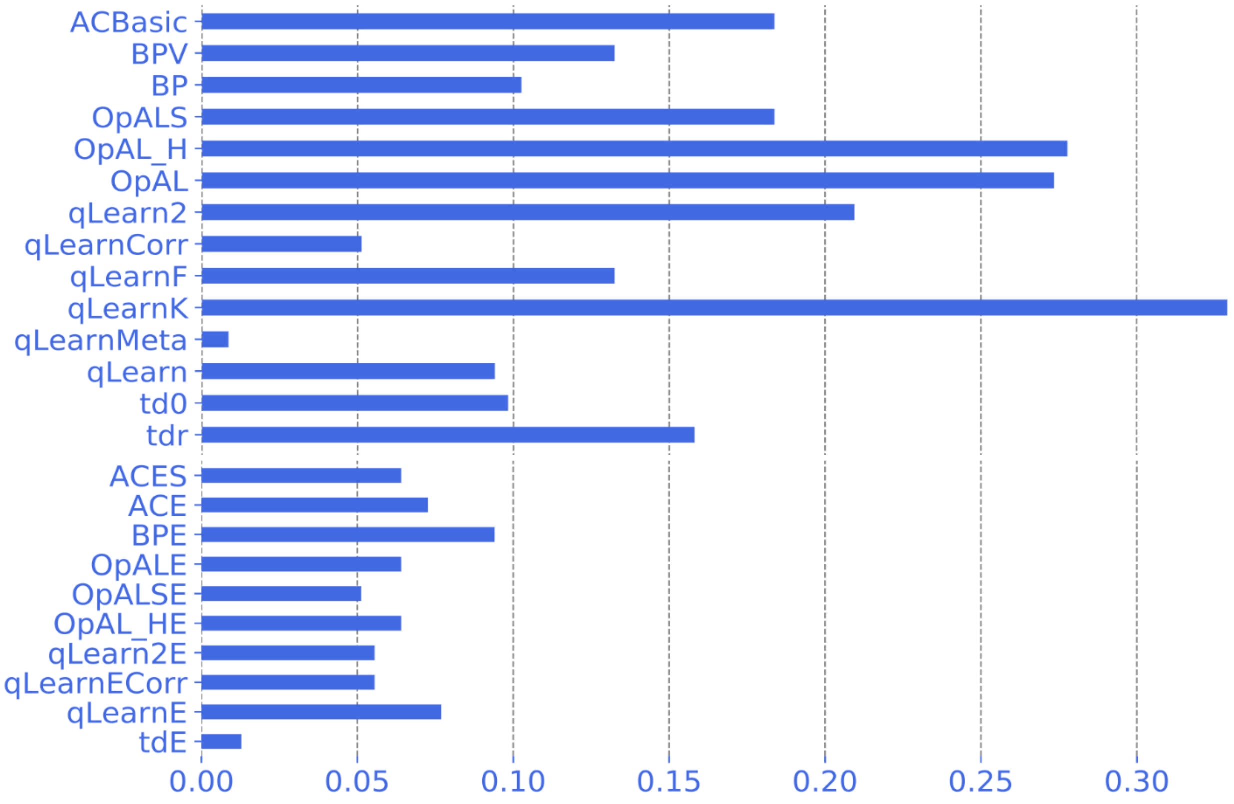


*Figure 8-5 The proportion of the 234 participants fitted to each model whose fits had at least one recovered parameter within 0.1% of its boundary. The models have been grouped into those using softmax (**top**) and those using epsilon-greedy (**bottom**).*

8.2.2 Goodness of model fits

A model's performance at fitting a participant's actions can be compared using a Bayes factor; comparing its performance to that of the random choice model, as described in chapter 2.2. For the Weather task dataset, using a Bayes factor of 20 as a criterion for strong evidence for a model fit, 100 of the 234 participants had strong evidence for at least one model fitting their actions well. This can be seen in Figure 8-7, and using the normalised Bayes factor, in Figure 8-6. As with the Decks task in chapter 6.2.2 and the Probabilistic Selection task in 7.2.2, there does appear to be a similarity in Bayes factor value distributions within model classes, with the OpAL models having the weakest evidence that they are better than a purely random model, each having strong evidence for at most 31 participants. The Bayesian inspired models have some of the strongest evidence, with BPV having strong evidence for better fitting 77 participants. As the stimulus cues in this task are evenly balanced in their associated correct action choices, the randomBias

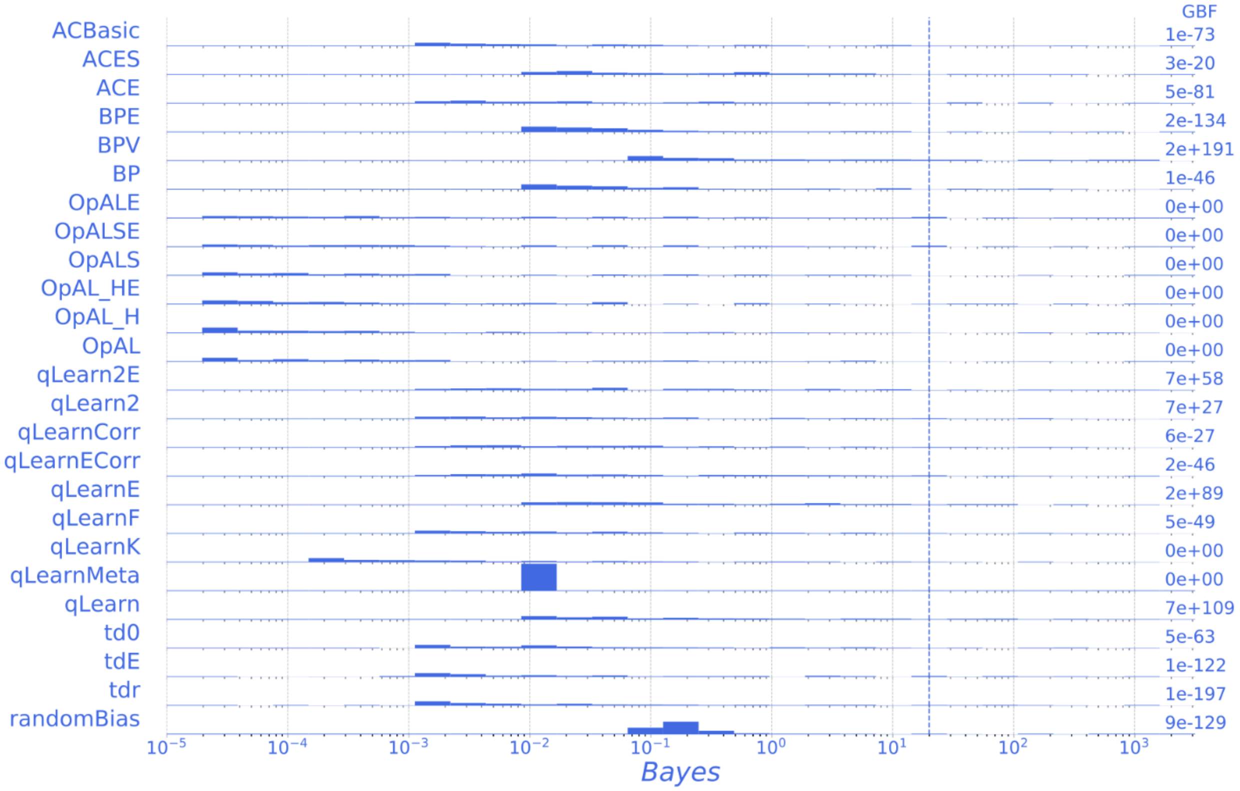


Figure 8-6 The distribution of the values of the fit quality Bayes factor from fitting the Probabilistic Selection dataset when compared to a pure random model. The dashed line marks a Bayes factor of 20, above which there is strong evidence for the model. On the right are the Group Bayes Factors for the model, defined in equation 2.9.

model would be expected to perform not much better than a pure random model and in fact, only 6 participants fits had a Bayes factor above 20. Models on average had around 40 participant fits with a Bayes factor above 20, with 18 participants having strong evidence for at least 20 of the 25 models.

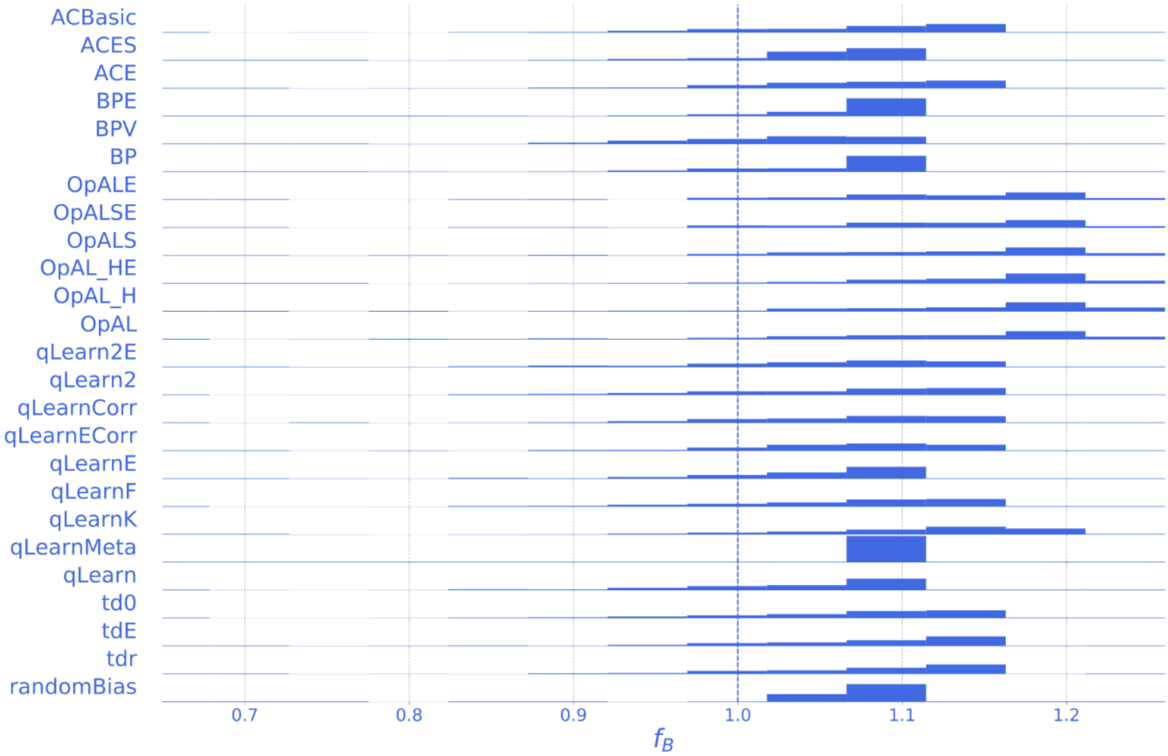


Figure 8-7 The distribution of the values of the fit quality normalised Bayes factor from fitting the Probabilistic Selection dataset when compared to a pure random model. Fits with values below 1 have a Bayes factor of over 20, so have strong evidence that the model can match the participant's actions better than a pure random model.

The Bayes factor values for each participant's model fits can be compared between models, as shown in Figure 8-9. In this figure, identical Bayes factors for the same participant are located along the diagonal line in each small plot. This is the case for almost all participants when comparing, for example, the OpAL and the OpALS models, as shown more clearly in Figure 8-8. If one model fits a participant consistently better than another model, then the participant will be shown away from the diagonal line. For example, the qLearn model fits have stronger evidence than those of OpAL_H, as they are on the qLearn side of the diagonal line. Nevertheless, they are still related, as most of the dots form a line that is parallel to the diagonal line of equal Bayes factor. The further away a dot is from the diagonal line, the greater the difference in the Bayes factors of the two models for that participant data. The horizontal and vertical lines mark a Bayes factor of 20 for the fit of the model axis they intersect with, i.e. the vertical axis model fit values are

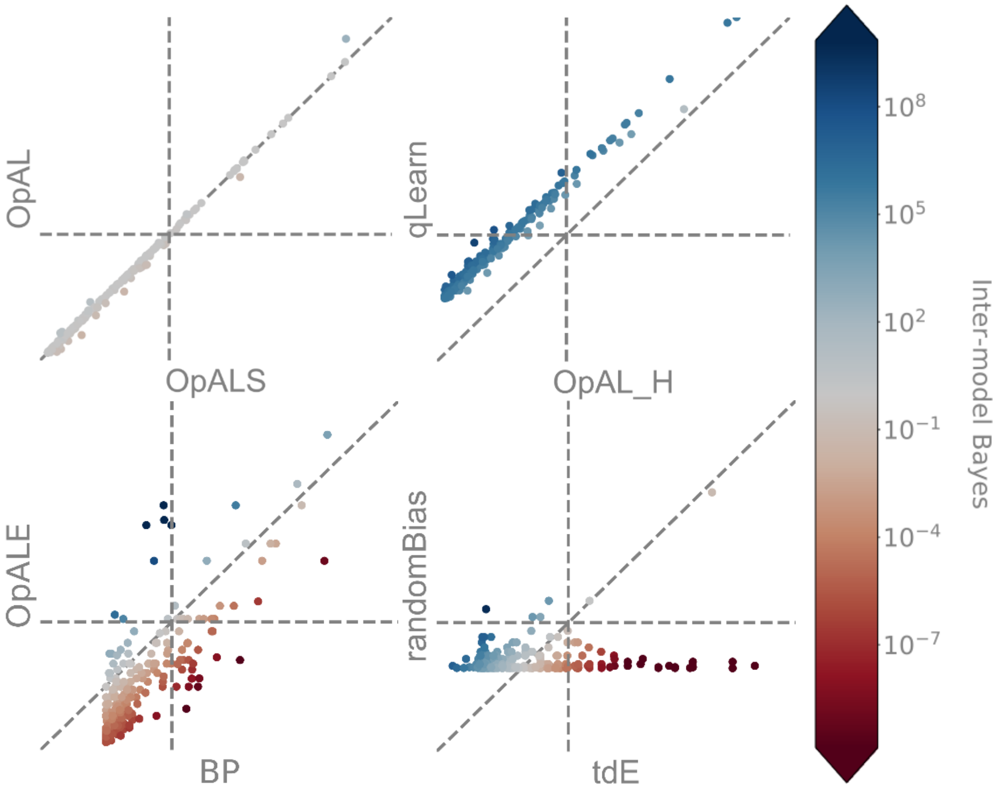


Figure 8-8 A selection of expanded miniplots from Figure 8-9

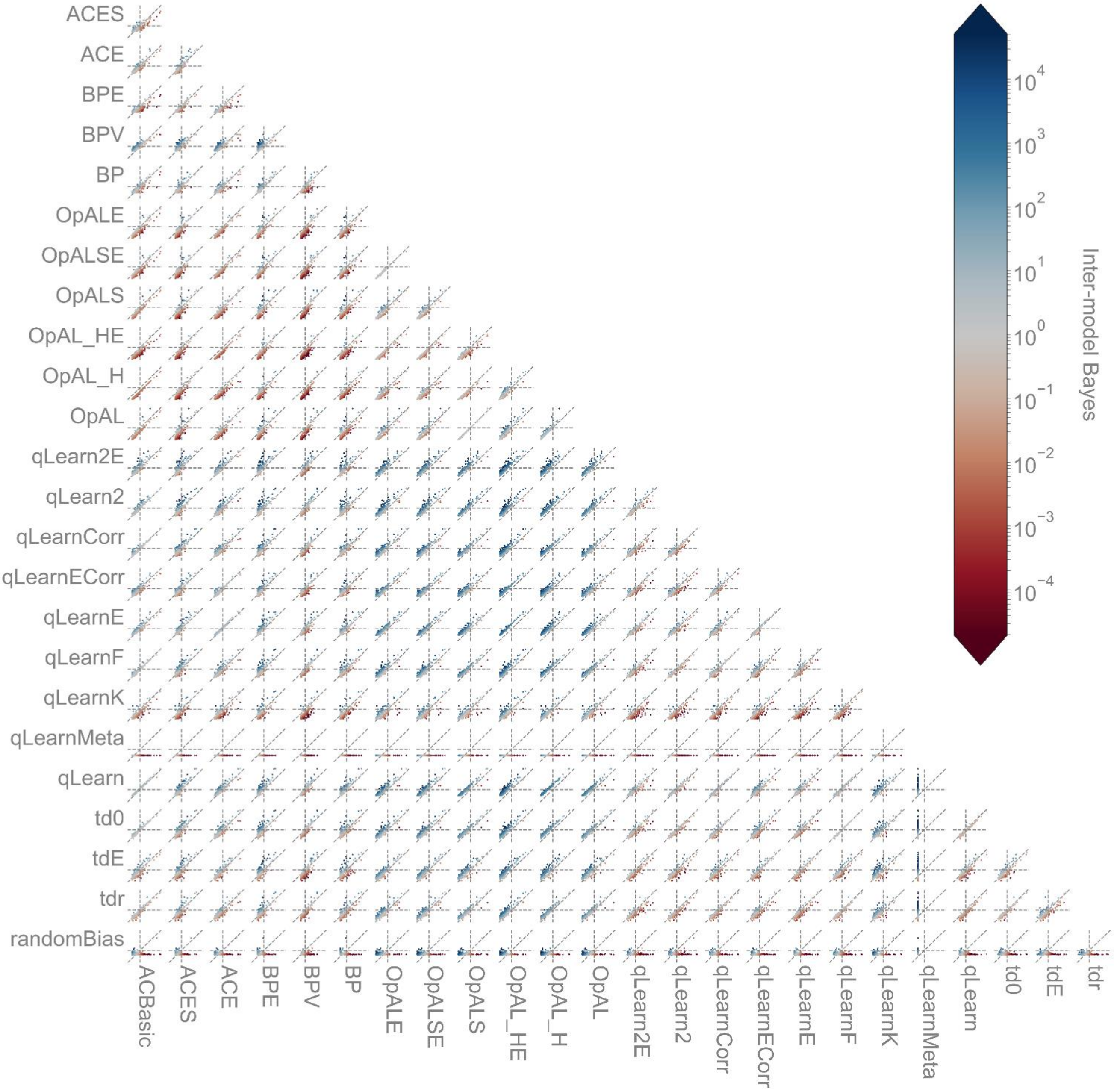


Figure 8-9 A comparison between models of the Bayes factor values for each participant when compared to a pure random model. Both the horizontal and vertical axes of each model comparison use a log scale ranging from 10^{-5} to 10^{18}. The horizontal and vertical lines denote a Bayes factor of 20 and the diagonal line follows the line of equal value for both axes. The dots are coloured with the inter-model Bayes factor, such that a Bayes factor of 20 signifies that there is strong evidence that the vertical model fits better the participant's actions than the horizontal model.

associated with the horizontal dashed line. Therefore, for most participants, the Bayes factor values for the randomBias model relative to a pure random model are lower than 20, whereas the Bayes factor values of the tdE model have a broad range of values relative to the pure random model. This results in the dots in their comparative plot not only being mostly on the tdE side of the diagonal line, but also beyond the horizontal dashed line.

The relative difference between model fits be expressed as a between-model Bayes value, calculated using the model fit BIC values by taking inspiration from equations 2.4 and 2.8 in chapter 2.2, so that:

$$\mathcal{B} = 2^{\frac{\mathrm{BIC}_{model\,1} - \mathrm{BIC}_{model\,2}}{2}}$$

These are shown in Figure 8-9 by the colour of each participant's dot, with the vertical model as model 2 and the horizontal model as model 1. Therefore, the participant dots with a Bayes factor of 20 or higher, coloured blue, have strong evidence that the vertical model is a better fit for the participant's actions than the horizontal model. Conversely, with a Bayes factor of 1/20 or lower, coloured red, there is strong evidence that the horizontal axis model is a better fit for the participant's actions than the vertical axis model. The median values for each inter-model comparison are shown in Figure 8-10. This highlights the poor performance of the OpAL models, especially OpAL_H and OpAL_HE, and qLearnK to fit participant data on this task. BPV can be seen to perform better than all the other models. There are occasions where Figure 8-10 can be misleading, such as when it suggests that there is somewhat strong evidence for BP to fit the participants better than OpALE, with a median inter-model Bayes factor of 64, whereas Figure 8-9 shows that there is strong evidence that OpALE and BP each fit different participants well.

Another approach to comparing the model's performance is to examine the expectation of the model frequencies, EF, as discussed in chapter 2.4. This assess the relative frequencies with which two models could have generated participant data in the dataset. The probability that this relative frequency is above chance is estimated using the protected exceedance probability (Rigoux et al., 2014). Both the expectation of the model frequencies and the related protected exceedance probability were calculated using the VBA toolbox (Daunizeau et al., 2014). This used as inputs the BIC values calculated for each model's fit to each participant's task action sequence. These comparisons can be seen in Figure 8-11. This reinforces our previous conclusion that the BPV model has the strongest evidence, followed by qLearn and qLearnE.

When removing all the edge fits found in chapter 8.2.1, the distributions of fit qualities do not change significantly, suggesting that model fits that result in parameters recovered at a boundary do not have particularly higher or lower fit quality values than others.

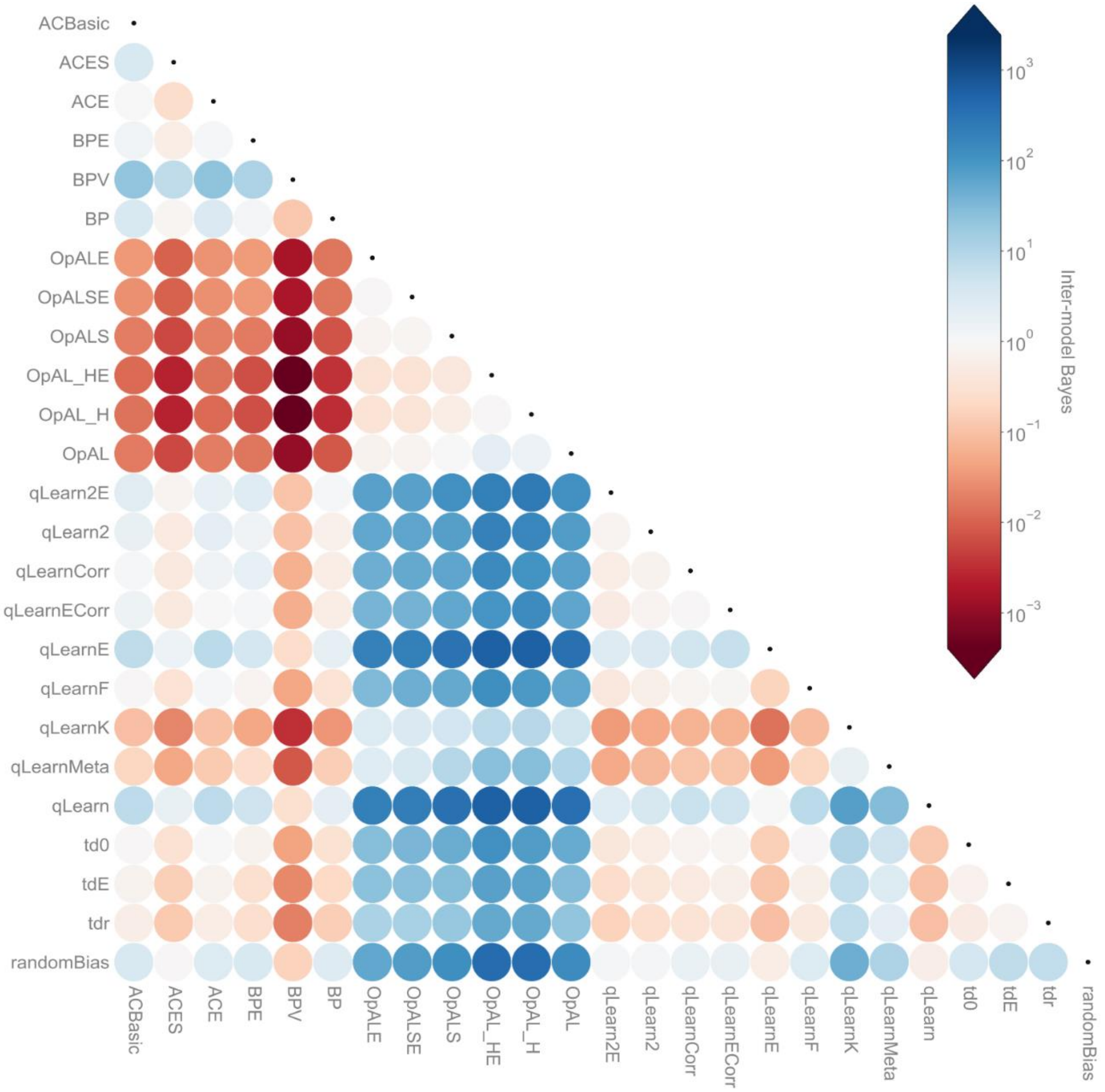


Figure 8-10 The median inter-model Bayes values for the Probabilistic Selection dataset participant fits. High values signify that the model on the vertical axis had a lower BIC value, and so a better match of the participant's actions, than the horizontal axis model.

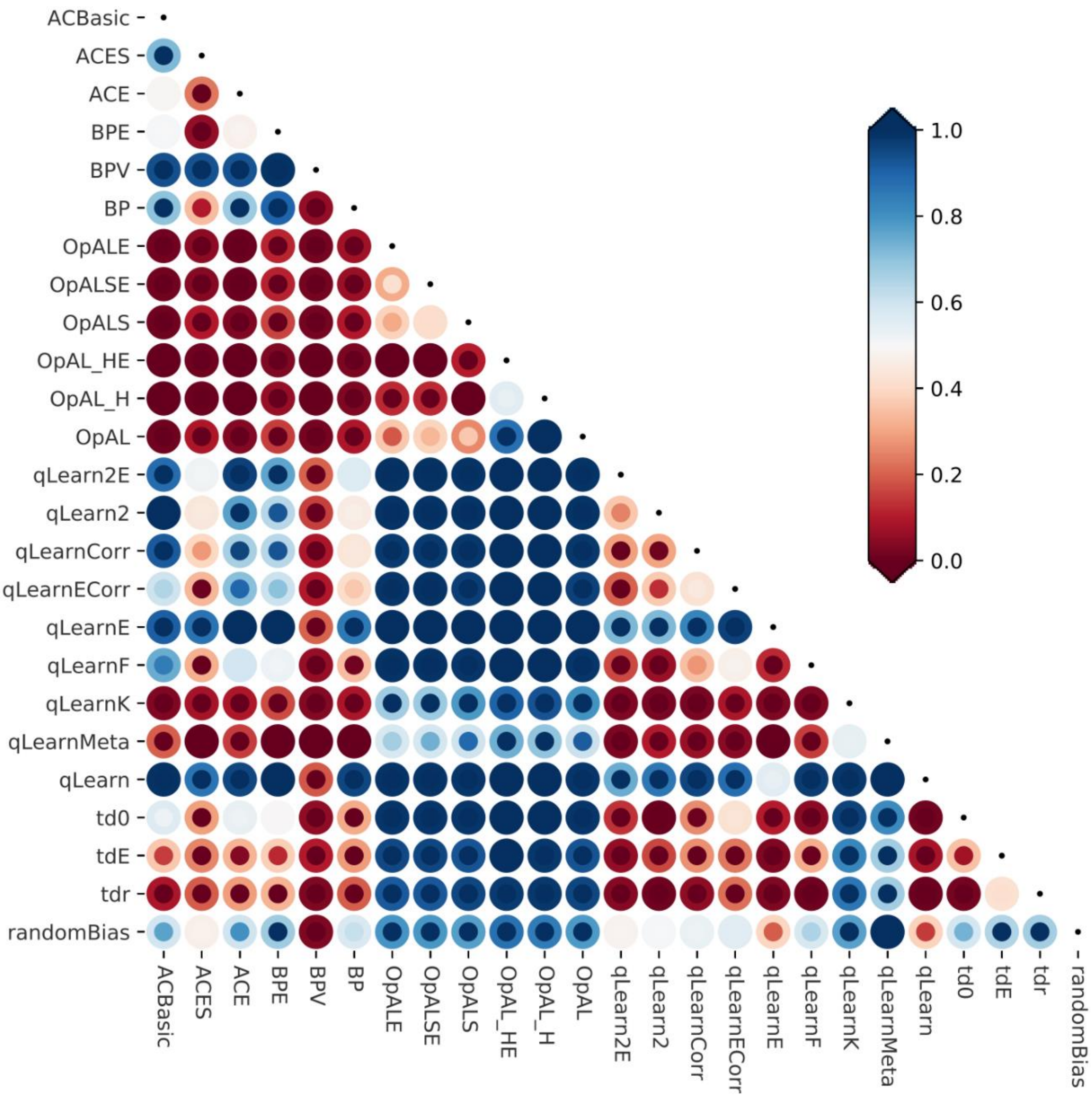


Figure 8-11 The expectation of the model frequencies (EF) and the associated protected exceedance probabilities (pEP) for model pairs. Each pair of circles shows the EF and pEP for the vertical axis model relative to that of the horizontal axis model. The outer, larger circle is the EF and the inner circle is the pEP. Both are scaled between [0,1].

To provide an estimate of the number of participant fits that may be considered to have good fits, we can use the proportion of participants with a fit that has a Bayes factor of 20 or more, and with recovered parameters not on the edge of the parameter boundaries. The proportion of participants with not good fits, for each model, can be seen in Figure 8-12. While all of these models had fewer than half of their participants with good fits, the best performing models are the qLearn variants and BPV. The success of the qLearn variants was found with both the Decks task, in chapter 6.2.2, and the Probabilistic Selection task, in chapter 7.2.2

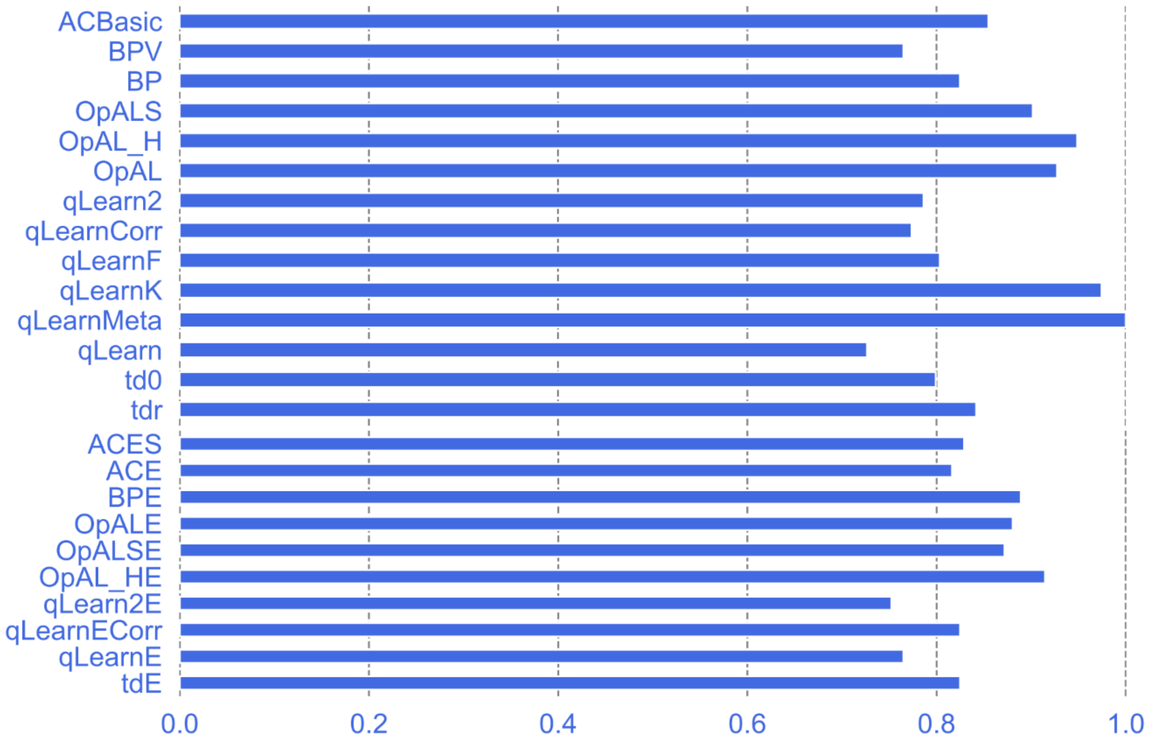


*Figure 8-12 The proportion of the 234 participants fitted to each model whose fits had at least one recovered parameter within 0.1% of its boundary and the Bayes factor of the fit was below 20 when compared to the pure random model. The models have been grouped into those using softmax (**top**) and those using epsilon-greedy (**bottom**).*

and the success of the BPV model was also seen in the Probabilistic Selection task, but not the Decks task.

Participant's model fit quality was compared to the frequency with which they predicted the correct outcome when presented with a same pair cue stimuli, in Figure 8-13 finding that there is a strong correlation across almost all models. As correlations had been found in chapter 8.1 between participant's frequency of good responses to different cue groups, by transitivity this correlation can be extended to the other cue groups. This suggests that only those participants who learnt the relationships between cues and outcomes were well fitted by the majority of models. Accurate parameter recovery could therefore be expected for versions of the task where more participant actions were well fitted to models.

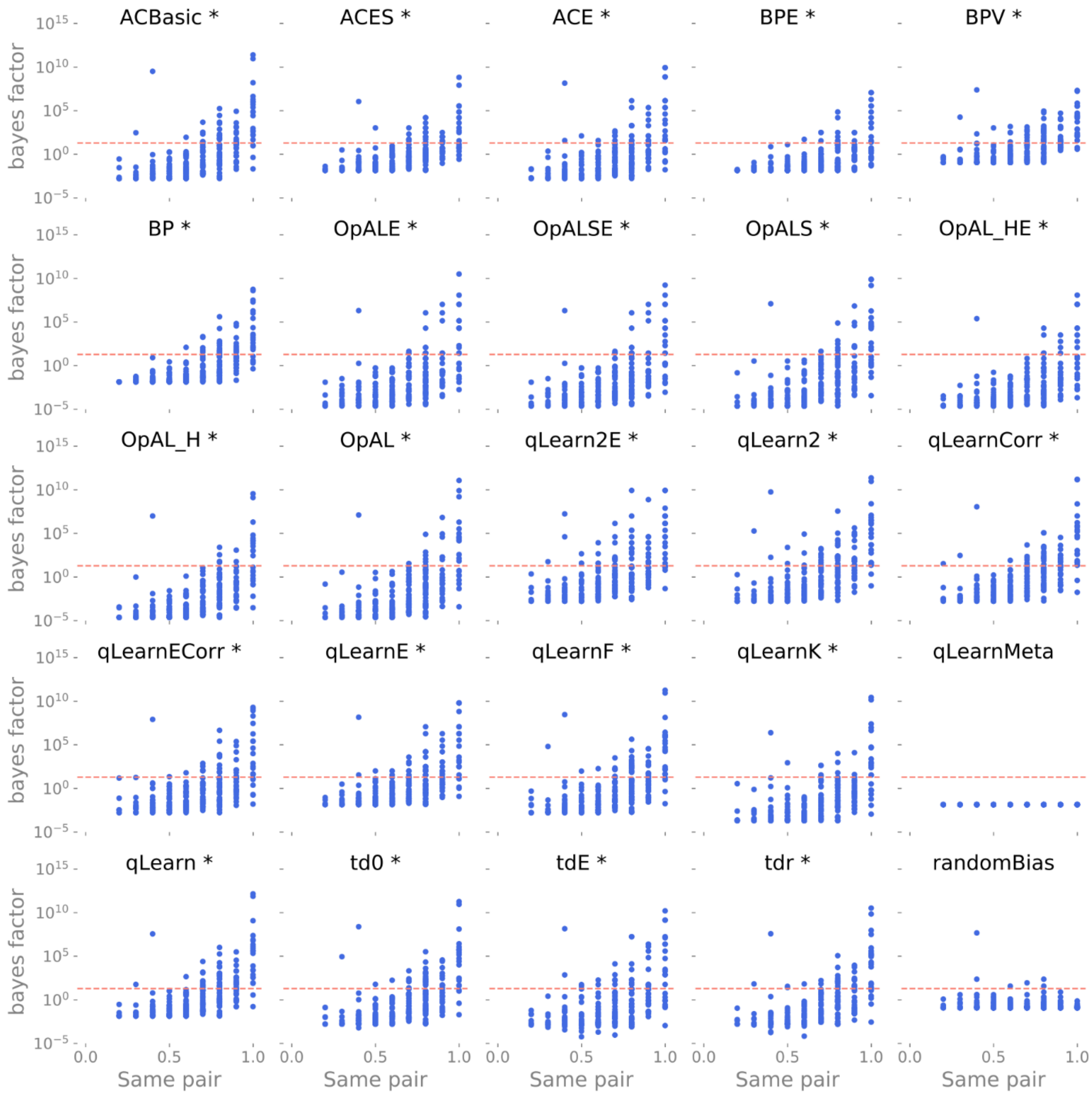


Figure 8-13 Proportion of same pair cues stimuli choices made by each participant that matches with the most likely outcome, as described in Table 8-1, compared to the model fit Bayes factor when compared to a pure random model. The star denotes a Bonferroni corrected Spearman's rank correlation with a p <0.05. The pink dashed line denotes a Bayes factor of 20.

8.2.3 Parameter correlations

If the recovered model parameters are identifying a feature of a participant's learning and decision-making process then we would expect that in different models the same parameter, performing the same task, would have very similar values. The correlations between parameters across models should therefore be high for parameters performing the same role in different models and low between parameters performing different roles, especially those in the same model. In this section, the correlations of the three most common parameters are examined: α, β and ε. A plot of the full comparison between model parameters can be seen in Appendix II. The correlations for the learning rate parameter, α, shown in Figure 8-14, need to be broken down further, as learning rates are used for learning various estimators. Overall, these are of similar strength to those found for the Probabilistic Selection task, Figure 7-17, and weaker than those found for the Decks task, Figure 6-18.

One subgroup of α parameters is those of the Q-learning class of models, with only one learning rate parameter and without separate actor and critic components, as described in chapters 3.2, 3.5, 0 and 4.8. Collectively, their correlations result in a Kendall's W, a measure of collective concordance, of 0.47, which suggests a weak correlation, but stronger than that of the Probabilistic Selection task, 0.31 and weaker than that of the Decks task, 0.68. Looking at the individual model parameter pair correlations, shown in Figure 8-15 they have a similar arrangement of strong and weak correlations to those found for both of the previous datasets, Figure 6-19 and Figure 7-18. The correlations between the critic α parameters, also shown in Figure 8-15, show a similar strength distribution, albeit with a higher lower bound of correlation strength.

As with α, correlations could be expected for the α^+ and α^- parameters found in the OpAL models and qLearn2 variants. However, as can be seen in Figure 8-16, there are only weak correlations within these groups parameters, with Kendall's W values of 0.20 for the α^+ and 0.15 for the α^-. This is surprising, given the similarity of the OpAL models and equally surprising for the qLearn2 models, but has been consistent across the three tasks, with the Decks Kendall's W values of 0.16 for α^+ and 0.15 for α^- and the Probabilistic Selection task Kendall's W values of 0.20 for the α^+ and 0.16 for the α^-. This consistency continues with the parameter pairs that have strong and weak correlations, such as OpAL and OpALS α^+ and α^-. Once again, the correlations between parameters within each model are also quite low for both qLearn2 models (-0.02 for qLearn2 and 0.03 for qLearn2E) and for the OpAL models (values ranging from -0.25 and 0.1). It is therefore unlikely that these weak correlations are due to a difficulty in teasing apart the influence of the α^+ and α^- parameters.

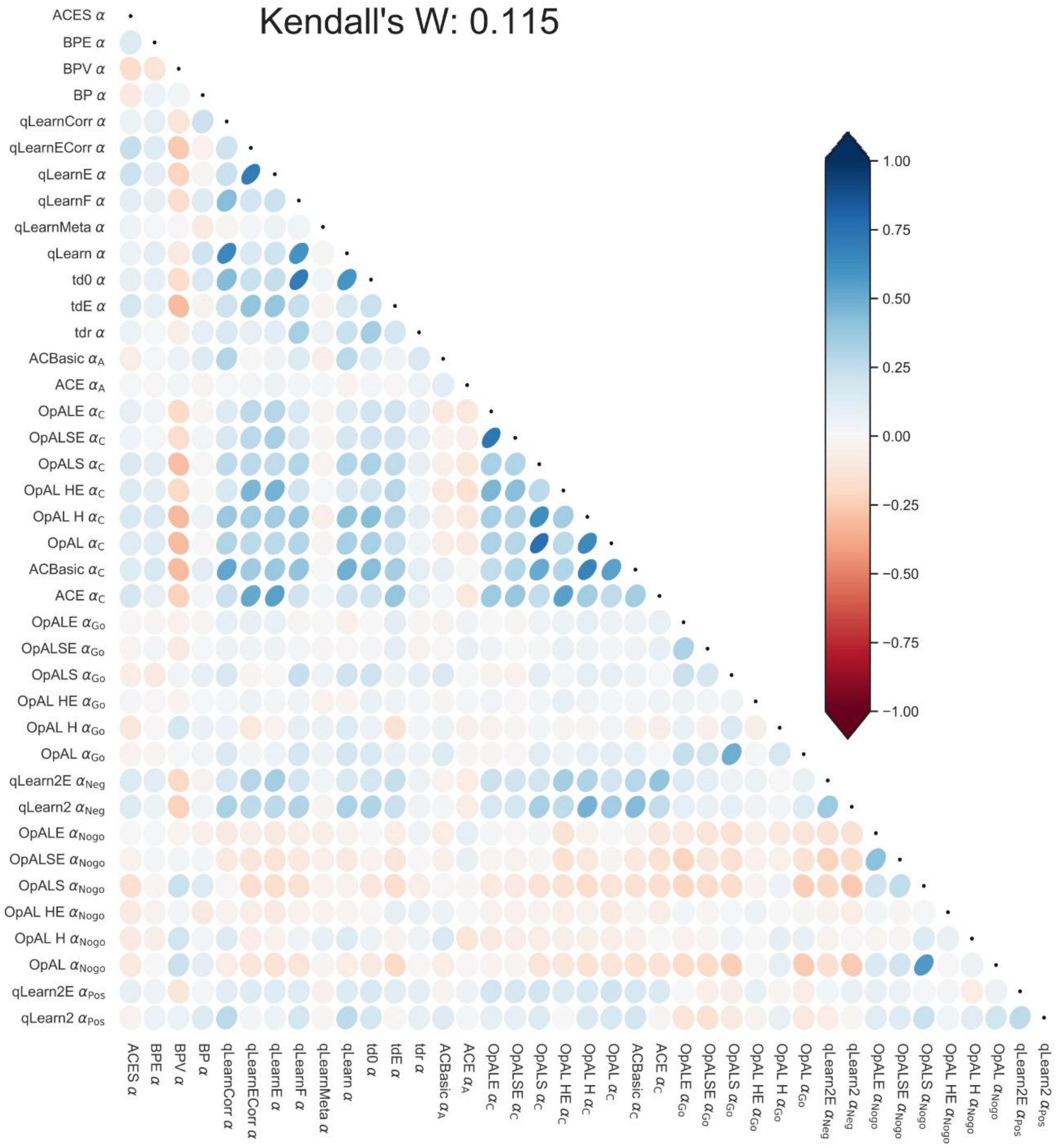


Figure 8-14 The correlations between recovered α parameter values from the Weather task participants. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation.

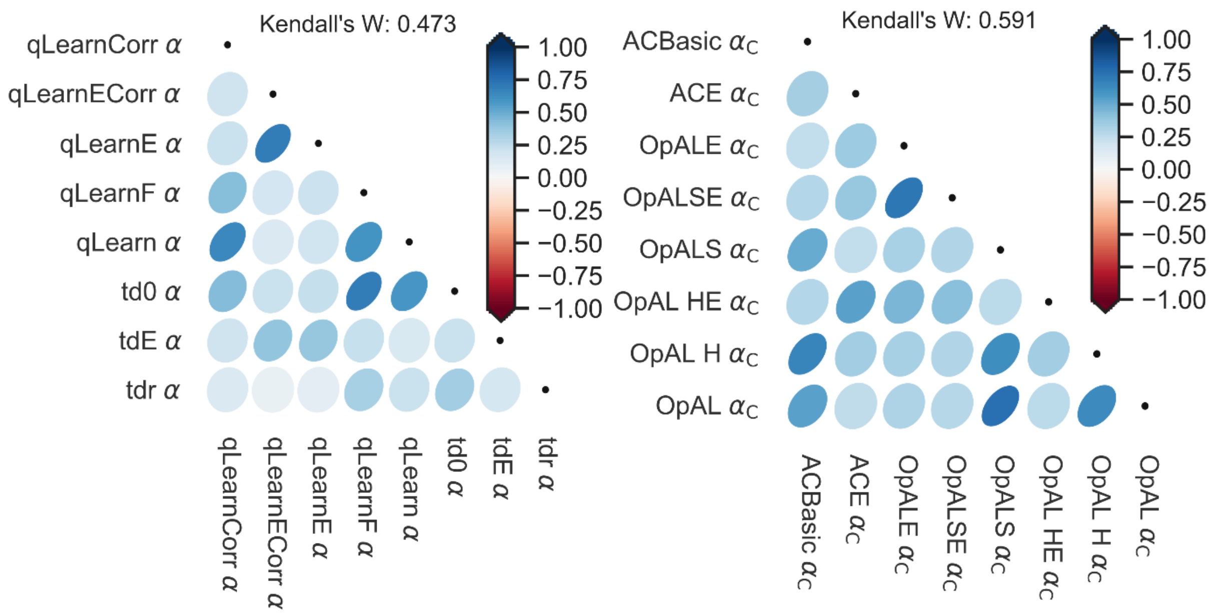


Figure 8-15 The correlations between recovered parameter values from the Weather task participants. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation. ***Left:*** *The critic learning rate parameters, α, in the critic only q-learning models* ***Right:*** *The critic learning rates from models with both actor and critic learning rates.*

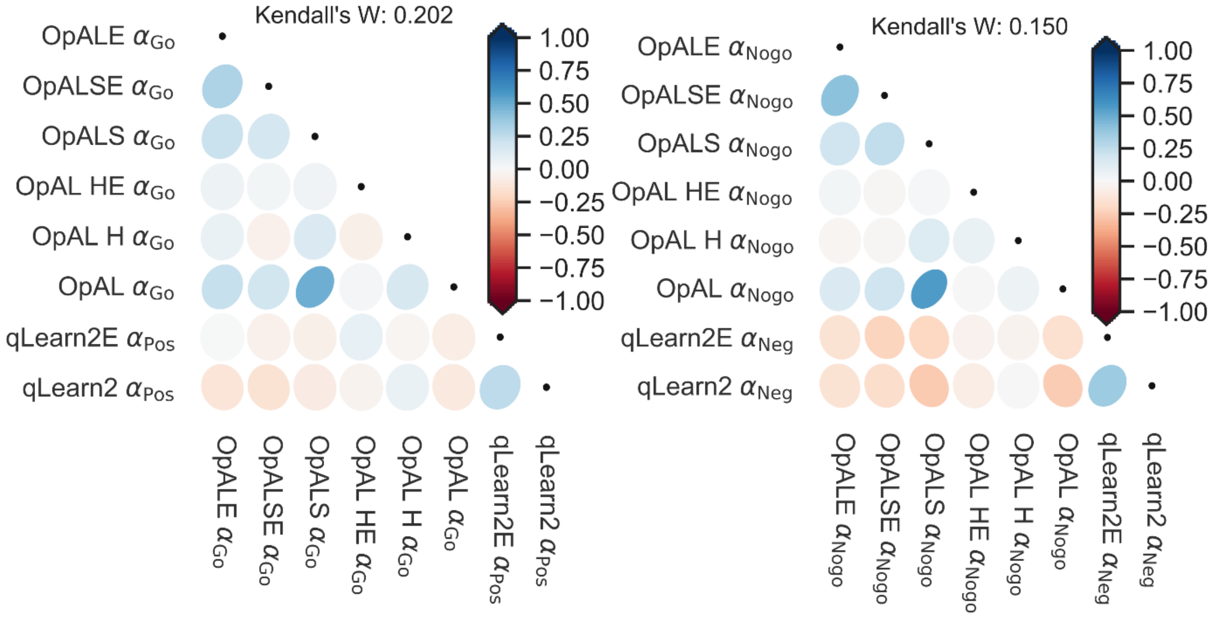


Figure 8-16 The correlations between recovered parameter values from the Weather task participants. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation. ***Left:*** *The learning rate for positive rewards* ***Right:*** *The learning rate for negative rewards.*

The same analysis was performed for the β and ε parameters, as shown in Figure 8-17 and Figure 8-18. The recovered ε parameters have strong correlations, as shown by the Kendall's W value across the models of 0.90. This contrasts with the large number of weak correlations for β, with a Kendall's W of 0.37, acting as further indications that the ε-greedy function provides more consistent parameter recovery than that of SoftMax. The only β parameters that are strongly correlated to each other are those for the Q-learning class of models shown in Figure 8-17, with a combined Kendall's W of 0.74

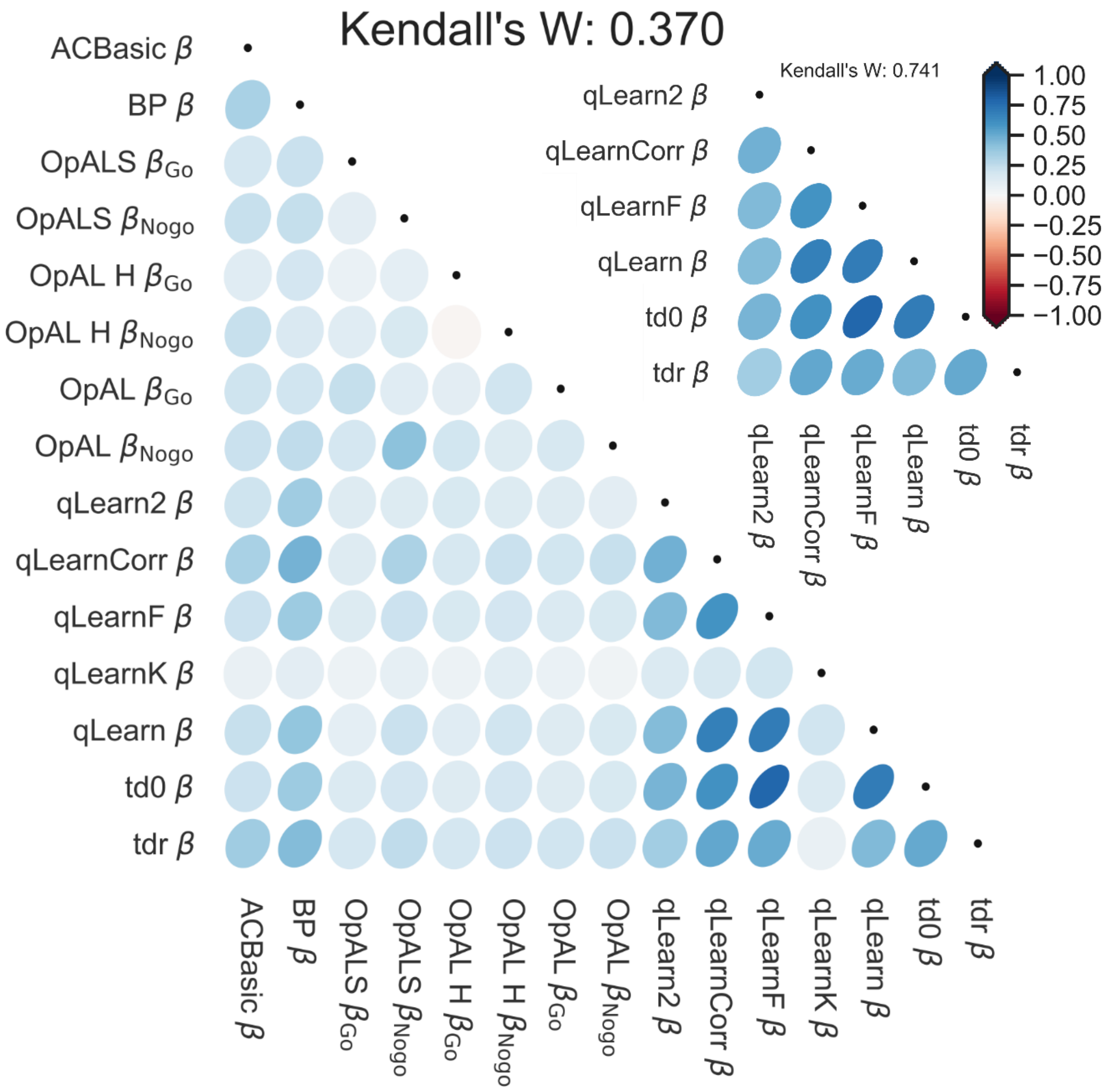


Figure 8-17 The correlations between recovered parameter values from the Weather task participants. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation. ***Top right:*** *The β parameters for the Q-learning models* ***Bottom left:*** *The β parameters in all models.*

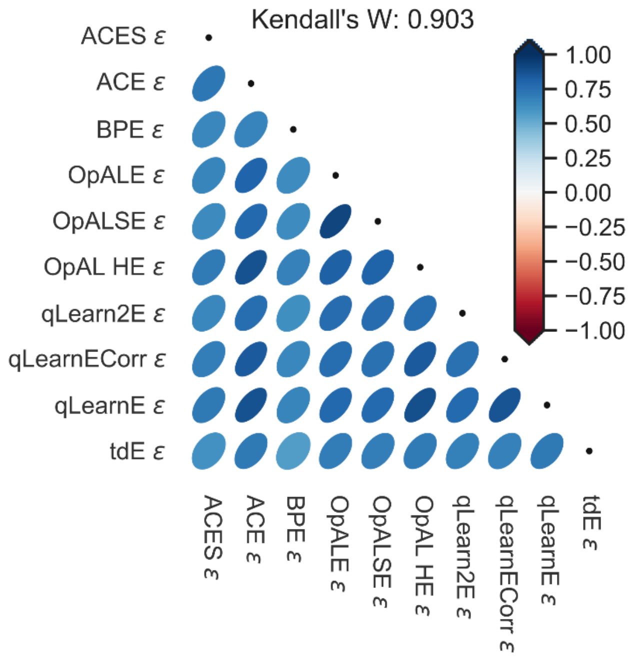


Figure 8-18 The correlations between recovered parameter values from the Weather task participants for models with an ε parameter. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation.

The degree of variability in correlation strengths between parameters performing similar tasks in different models may be explained by the differences in the capacity of models to fit the performance of the participants. A model that struggles to adequately explain the actions of most participants will have similar log likelihood values for different parameter combinations, resulting in more noise in the final recovered parameter values.

Another potential source of noise may stem from differences in model complexities resulting in varying difficulties in fitting models. Models with more parameters or more layers, i.e. more degrees of freedom, will require more information to fit as accurately as simpler models. This could explain the poor performance of the OpAL and qLearnK models that have the most parameters.

One confounding factor in describing a models degrees of freedom is that recovered model parameters are known to be correlated (Schmiedek et al., 2007).

In chapter 4, the distribution of errors in parameter recovery for the qLearn model parameters α and β suggested that the recovery process resulted in the parameters being inversely correlated. For this task dataset, the correlation was found to be -0.45, matching previous results. By contrast, for qLearnE, the correlation between α and ε was found to be 0.22, which adds evidence to ε-greedy being more recoverable than the SoftMax parameter β. However, models with large numbers of parameters. Such as OpALS, shown in Figure 7-22, do not show strong correlations between any of the parameters, suggesting that these correlations have the potential to be more pronounced in models with only two or three parameters.

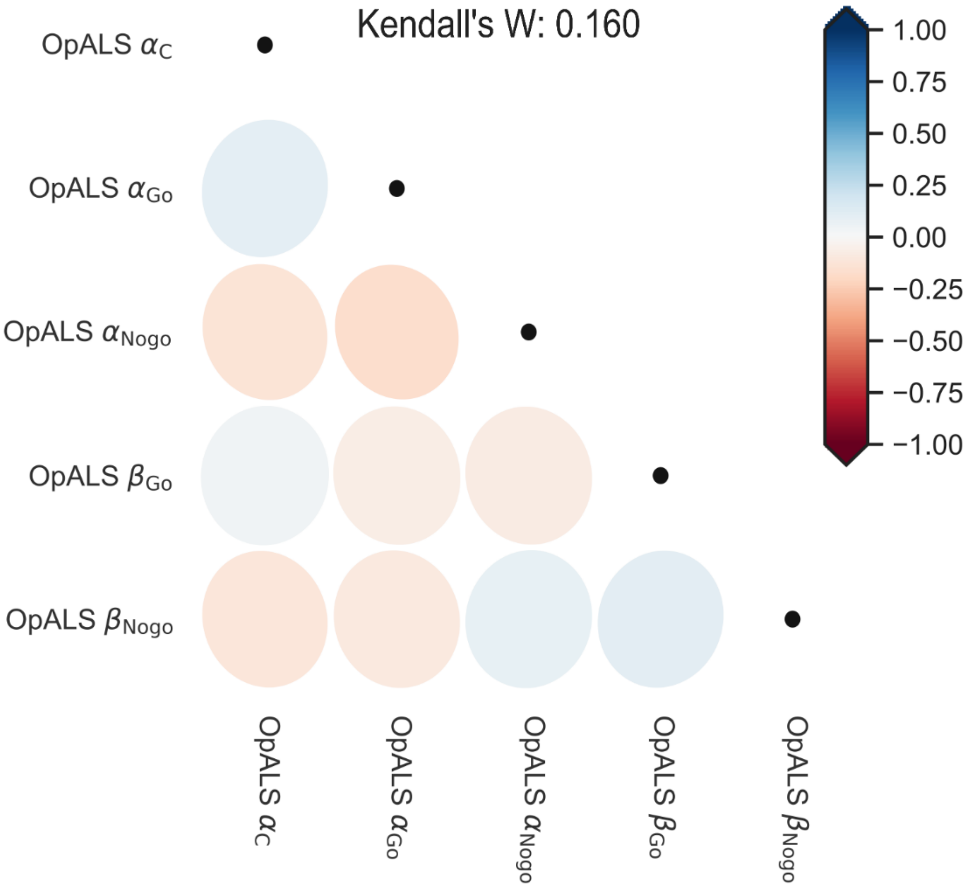


Figure 8-19 The correlations between recovered parameter values from the Weather task participants for the model OpALS. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation.

8.3 Model parameter relationships to extraversion

Having found correlations between the recovered α parameter and a measure of the extraversion phenotype with the Decks task datasets, in chapter 6.3, and with the Probabilistic Selection task, in chapter 7.3, this was also examined for the Weather task. As the extraversion measure of EPQ-R was also recorded for this dataset, the same analysis could be performed here as had been for the others: comparing the extraversion measures for each of the participants to the combined fitted parameter values for Q-learning class of models, with only one learning rate parameter and without separate actor and critic components, shown in Figure 8-15 (left). These are: qLearnCorr, qLearnECorr, qLearnE, qLearnF, qLearn, td0, tdE and tdr. The mean of the α values was calculated for each participant from those recovered from the Q-learning class of models. A Spearman's rank correlation was performed between the mean α value for each participant and the participant's measured extraversion value of EPQ-R.

To remove some of the noise from this correlation, those fits that did not have strong evidence of being better than a pure random model or who had one or more parameters close to their parameter boundaries were removed before calculating the mean α values for each participant.

To reduce the potential of any intra-model correlations between α and β or ε affecting the correlations, separate means were also calculated for the subset of models with a β parameter and for those with an ε parameter. This split is especially important as the correlations between α and β, and α and ε are in the opposite directions, negative for α and β and positive for α and ε. Estimates of those models mean β and ε parameters were also calculated for each participant. Partial correlations were performed between the extraversion measure and each of these α parameter estimates.

In total this resulted in six correlations between participant estimates of α and the extraversion measure of EPQ-R, shown in Table 8-2. These show some weak positive and negative correlations between α and extraversion. The α values calculated from models with β had the strongest correlations and this improved when only good model fits were included, as was found with the previous two task datasets. However, unlike those task datasets, none of the correlations were significant.

Participant model fit parameters used	*All*		*Good edge & Bayes*	
	ρ (DF)	*p*	*ρ (DF)*	*p*
Q-learning models	0.032 (235)	0.622	0.050 (97)	0.625
Q-learning models with β	-0.088 (232)	0.177	-0.171 (82)	0.120
Q-learning models with ε	0.072 (232)	0.275	-0.069 (78)	0.543

Table 8-2 The correlations between each participant's averaged Q-learning model parameter α and the extraversion measure of EPQ-R. The α, β and ε values in these correlations are averaged, for each participant, across the relevant subset of models. The β and ε subset correlations were partial correlations whereas the others were simple correlations.

8.4 Parameter recovery across tasks

One of the key assumptions underlying this thesis was that model parameters represent stable properties of participant's probabilistic learning and decision-making processes, as discussed in chapter 1.2. This can be tested by assessing the consistency of recovered parameters for the same participants across different tasks. The participants who performed the Probabilistic Selection task also performed the Weather task, allowing their recovered parameters to be compared, as shown in Figure 8-20. If these recovered parameter values are similar, a plot of the recovered parameter values would be expected to form a diagonal line from the lowest parameter values (bottom left) to the highest (top right) of a subfigure. For these datasets, parameter recovery does not seem to be consistent across tasks for any of the model parameters examined. The strongest correlation between recovered parameters across tasks is found for ε in OpALSE, with spearman rank 0.34 (p=0.004). This can be seen in an expanded subfigure in Figure 8-21. As in both task datasets strong correlations were found between the same parameters in different models, notably those of ε, seen in Figure 7-21 and Figure 8-18, it could be argued that the issue is not poor parameter recovery.

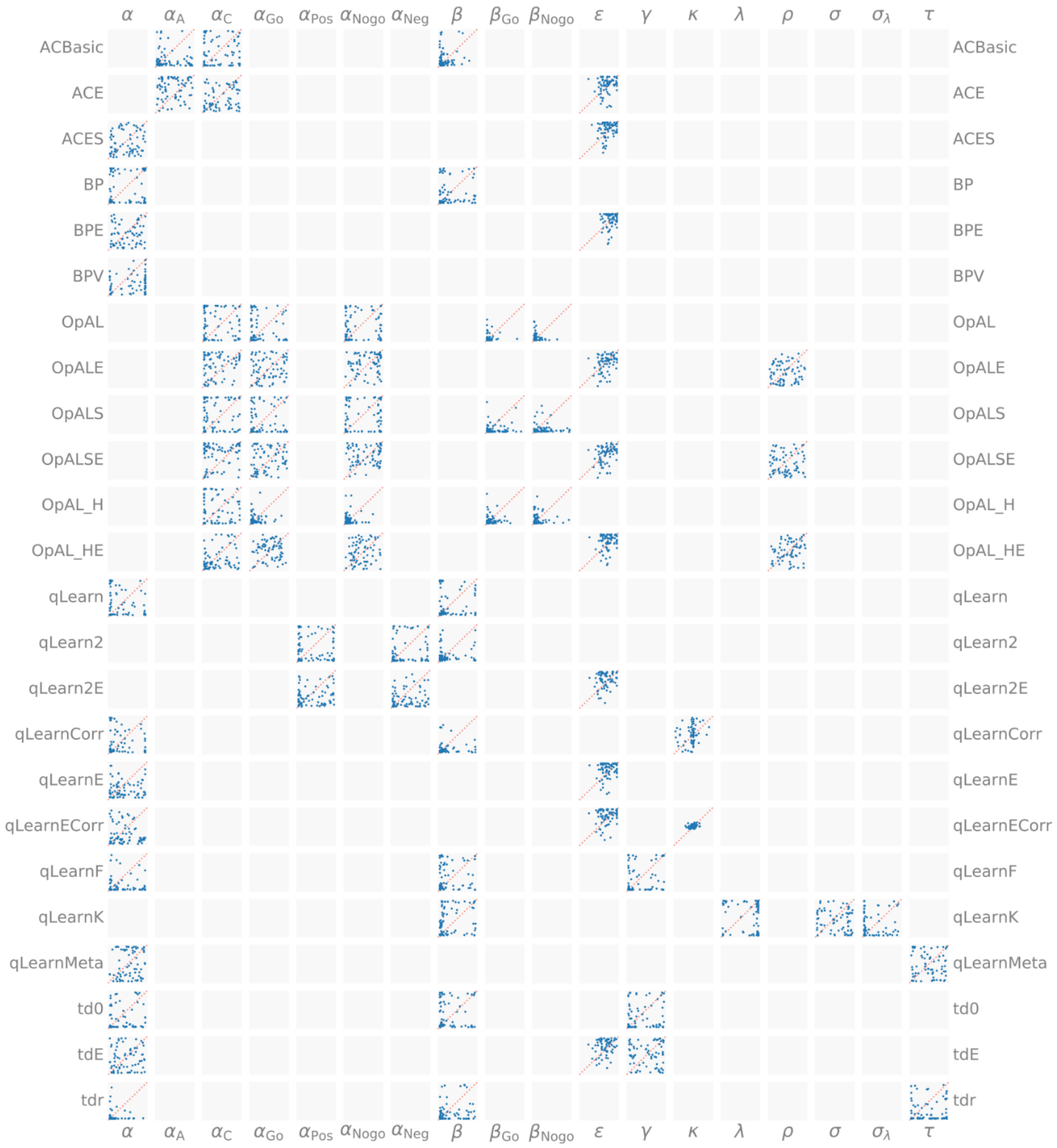


Figure 8-20 A comparison of the recovered parameters for the same participants across the Weather and Probabilistic Selection tasks. Each sub-figure plots recovered values for a parameter from a given model, with the values recovered from the Weather task shown on the horizontal axis and those of the Probabilistic Selection task on the vertical axis. The visible axes range are the same as the support used when fitting the model parameter being plotted. The diagonal dotted line follows the line of equal parameter value.

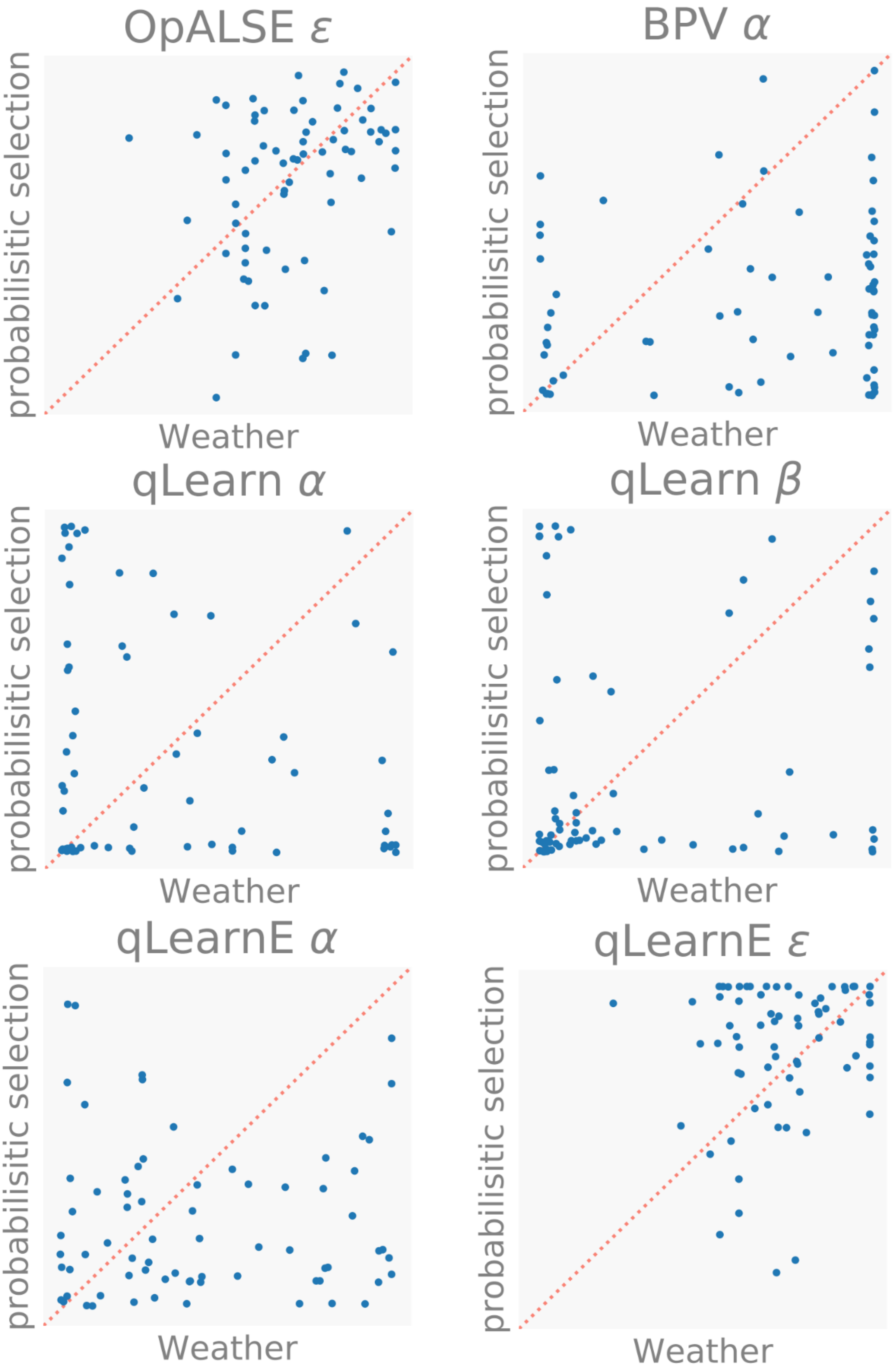


Figure 8-21 A selection of subfigures, from Figure 8-20, comparing the recovered parameters for the same participants across the Weather and Probabilistic Selection tasks. The visible axes range are the same as the support used when fitting the model parameter being plotted. The diagonal dotted line follows the line of equal parameter value.

However, as was seen in chapter 4.4, fitting an action sequence with a model recovers the same parameter values consistently, even when these are not close to the generating parameters. It is therefore plausible that the recovered values in these two datasets are overfitted to the particular participant action sequence, due to the low number of trials and small number of action choices each trial. This is further supported by the low Bayes factors for the model fits compared to the pure random model, suggesting that there is not enough evidence to consider these recovered parameters as accurate.

This is supported by the results of chapter 5, where it was found that the recovery of the α for qLearn and qLearnE were worse, under certain generating β and ε values, than if they had been randomly chosen from a uniform distribution. This was despite the action sequences being longer than those used in the participant datasets: 300 trials compared to 120 for the Probabilistic Selection task and 70 for the Weather task.

If the low Bayes factors were an indication of the poor recovered parameter correlation between tasks, it would be expected that those models with the strongest evidence would have the highest correlations. Looking at the models with the strongest evidence compared to a pure random model, BPV, qLearn and qLearnE, shown in expanded subfigures in Figure 8-21, does not provide much support for this, as their respective parameter correlations are still low.

One way of quantifying how poor the correlations are is by comparing the relationship between the recovered parameters to our ideal: a linear relationship with an intercept of zero. This can be done by calculating the sum of squared residuals (SSR). To make these easier to interpret, before calculating the residuals, the parameter values will be normalised to the support [0,1], i.e. for β values will be transformed from ranging between [0,30] to between [0,1]. Comparing the SSR for qLearn and qLearnE, as shown in Table 8-3, the residuals are found to be smaller for qLearnE than qLearn. This is consistent with the improved parameter recovery of epsilon greedy found in chapter 4.7.2, suggesting that improved parameter recovery does result in a stronger linear correlation.

	α	β/ε
qLearn	19.5	15.2
qLearnE	13.4	4.6

Table 8-3 The normalised sum of squared residuals when compared to a best fit line of equal parameter values

A comparison can also be made of each participant's average recovered model parameter values calculated for the comparisons with extraversion in chapters 7.3 and 8.3, shown in Figure 8-22. These do not show any stronger correlations than those of the individual recovered model parameters for the two tasks.

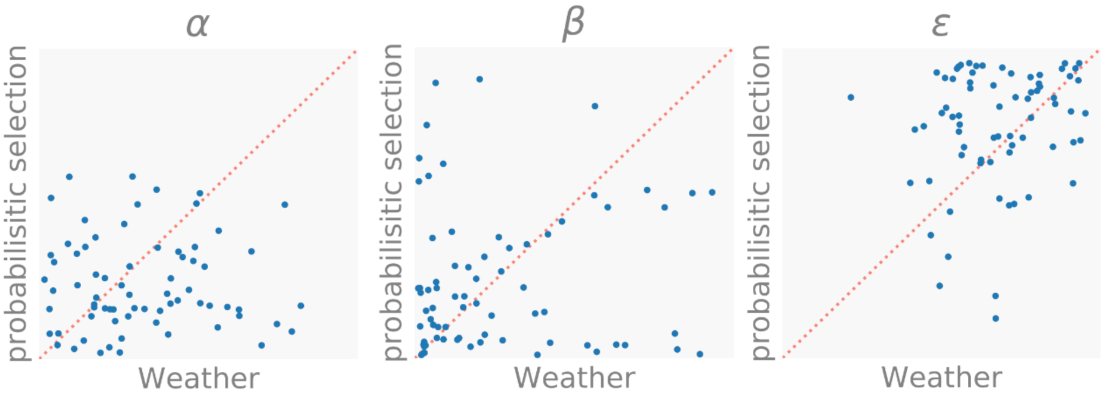


Figure 8-22 A comparison of the combined recovered model parameters for the same participants across the Weather and Probabilistic Selection tasks. Each sub-figure plots the combined recovered values for a parameter, with the values recovered from the Weather task shown on the horizontal axis and those of the Probabilistic Selection task on the vertical axis. The visible axes range are the same as the support used when fitting the model parameter being plotted. The diagonal dotted line follows the line of equal parameter value.

8.5 Discussion

The Weather task tests participant's ability to learn how different combinations of stimulus cues affects the likelihood of each outcome. Three sets of data had previously been collected by two different groups of researchers using the same version of the task. These sets of data were analysed together, and models were fitted to them. The participants as a whole learnt the relationship between the different types of stimulus cue groups. Their success at this determined how easily the models were able to find model parameters to represent their behaviour.

The models were evaluated for their performance in producing the same action choices as those of the participants, with the baseline for their fit quality being set as the performance of the pure random model. The model fits were performed on all the action choice trials performed in the learning and test phases by each participant. Overall, the models fitted the participant data poorly, with only the best performing participants providing models with strong evidence, as defined in chapter 2.2, of a better fit than that of a pure random model. The most successful models were BPV, qLearn and qLearnE. The OpAL models provided the worst fits.

α parameters were inconsistently recovered with most being weakly correlated with each other. The ε-greedy parameters were found to be recovered quite consistently across all the models and much more consistently than the β parameter from equivalent models using softmax. The correlation strength between recovered β values across models was found to be related to the type of model. The most consistently recovered β parameters were those of the Bayesian and Q-learning variants.

No significant correlations were found when comparing an aggregate α calculated from those from Q-learning models and the EPQ-R extraversion measure. The strongest correlation found was for the participants who had strong evidence for at least one Q-learning model using softmax. This correlation was in the same direction as was found in previous two tasks, further suggesting that this correlation between α and extraversion might be a real effect.

For the participants who had performed both the Weather task and the Probabilistic Selection task, their recovered parameters were compared across tasks. Surprisingly, no correlations were found. However, as the parameter recovery was very noisy for all models in both tasks this conclusion should not be taken as evidence against there being underlying stable properties.

It is worth noting that these results reflect the performance of the models only as implemented. There may be other implementations of the same models that perform better or worse with slight tweaks to their implementation, to their starting parameter values or their parameter upper bounds. All of the models here had their learning of expected rewards organised per action-cue pair, so relationships between cues were never captured. It is likely that changing this method of storing the expected feedback would change the capacity of the models to identify task features, thereby providing scope for more participant learning features to be captured.

9 Discussion and future directions

This thesis aimed to evaluate models of human learning for probabilistic decision-making tasks. To do so, an evaluation framework, Tinker Taylor py (TTpy), was developed in Python allowing models to be compared like-for-like across a range of tasks. Models were drawn from the reinforcement learning literature along with a few similarly structured Bayesian learning models.

Datasets from three existing unpublished probabilistic decision-making tasks were examined. These tasks had a gains-only promotion focus: where the rewards only increased, and the overall task motivation was to maximise the received reward. The Decks task tested participant's capacity to learn about changing payoffs. The Probabilistic Selection task tests participant's capacity to learn and then apply an understanding of reward likelihoods from pairs of characters to novel pairs of those same characters. The Weather task tests participant's ability to learn how different combinations of stimulus cues affects the likelihood of a pair of outcomes. The models were fitted on all the task action choices trials performed by each participant. The fitting therefore assumed that the same model was used throughout a task to make all the choices.

9.1 Recovering accurate model parameters

An assessment was made of the capacity for standard model fitting methods to accurately recover model parameters using simulated data generated with the same model. Significant variability was found in recovery of parameters for the qLearn model across a range of common probabilistic decision-making tasks. This result was consistent when tested with both gradient descent and evolutionary fitting methods and was replicated in a MATLAB implementation of the framework. This error in parameter recovery did decrease as the number of trials increased. However, the identifiability between generating parameters still did not reach usable levels for task lengths of 1600 trials, which would be unsuitable for human participants without them getting fatigued or bored. As participants were shown to have learnt the task reward probabilities for the tasks examined here within the first 100 trials this underscores that fitting participant data requires significantly more information than participants, or models, need to learn simple task relationships.

As the distribution of parameter recovery errors varied from task to task and model to model, any prior distribution used to improve parameter recovery would have to be recalculated for each model-task pair, significantly reducing its usefulness. There was little or no variation when an action sequence was fitted multiple times, suggesting that the variability was not due to poor identification of the global minimum. A parameter set can generate many action sequences for a given task and those action sequences could themselves be generated by numerous different model parameter sets. Therefore, the most likely explanation is that the action-sequence was being fit to the parameters that are the most likely to generate the action sequence. Where there is very little difference in the likelihoods between potential generating parameters, the difference between the fit measures for these parameters became equally small, resulting in the ‘valleys' discussed in chapter 4.2.1. As different action sequences have different likelihoods of being generated by each parameter set, the location of the valley will differ for action sequences generated by the same parameter set and can sometimes not overlap.

By looking at the chosen action probabilities for the generated data it became clear that the softmax β parameter was influencing the parameter recovery, biasing the fit value function to highlight parameter values with higher β as being better fits. Alternatives to the conventional softmax function were explored and compared, with the epsilon greedy method found to be the most effective of those tested at providing discernible parameter values. Models were modified where possible to provide ε-greedy versions that could be used to evaluate the performance of the models.

From these results, epsilon greedy was found to have lower errors in parameter recovery then the simulated participant data when it was generated with the same model. Nassar & Frank (2016) simulated then fitted data with both epsilon greedy and softmax. They found that, irrespective of which is used, if the same one is not used to both generate and fit the data this will have a significant impact on the types of errors generated when recovering parameters. Therefore, another source of uncertainty when interpreting recovered parameters from participant data is a potential discrepancy between the method used to transform expected action rewards into action probabilities in the model and those used by the participants.

Nassar & Frank also note that all fitting of this kind assumes that the attention of the participant does not slip during the task, as this would result in action choices chosen using another model. These 'attentional lapse' actions are not acknowledged by the fitting process and will add noise that cannot be estimated by the processes described here, and so will have an impact on the accurate recovery of parameter values.

Simple probabilistic decision-making tasks, such as those studied in this thesis with a small number of trials and only two action options per trial, cannot, with conventional fitting techniques, be used to recover accurate model parameter values for individual participants. Fitting single task runs provides such uncertainty in the true parameter values that no conclusions can be drawn from the recovered parameter values. This issue was found in the models examined, including the Bayesian inspired models. The use of softmax rather than epsilon greedy was found to exacerbate the difficulties in recovering accurate model parameter values. The results found in chapter 8.4 underscore the difficulties in parameter recovery, where the model parameters recovered for participants who performed both the Weather and Probabilistic Selection tasks were compared with no significant correlations found between the same model parameters across tasks. This is somewhat surprising given how established these decision-making tasks are.

The errors in mean recovered parameters, as seen in chapter 5 for tasks of 300 trials, ranged between 10-30% of the support size for α and 4-11% for β and ε. The distribution of the β and ε errors found a larger spread for β than for ε, which was previously seen in chapter 4. Since the recovered parameters are the best fit for the action sequence, a greater spread implies a lower identifiability between parameter values. This in turn results in greater uncertainty for any distributions of recovered parameters, and any studies that look at group level effects will be affected by this issue. This conclusion is supported by the work of Humphries, Bruno, Karpievitch, & Wotherspoon (2015) for the expectancy valence model and Spektor & Kellen (2018) for Q-learning models with one or more learning rate parameter. The errors in the mean recovered parameters still resulted in a general trend that followed the generating parameters. The recovered parameters from such tasks could therefore, in large datasets, be used to identify correlations with other parameters.

The model fits were noted in chapter 2.2 as violating some of the assumptions underlying the BIC. Using other fitting approaches such as Markov chain Monte Carlo (MCMC), or Variational Bayes, used by the MATLAB VBA toolbox (Daunizeau et al., 2014), it is hoped that this would allow easier calculations of free energy (Klaas Enno Stephan et al., 2009) and WBIC (Watanabe, 2012) fit quality measures. This could be integrated using the PyMC3 (Salvatier, Wiecki, & Fonnesbeck, 2016) or Stan (Carpenter et al., 2017) implementations.

The no feedback “test” phase of the task was found to only be helpful for parameter recovery in the Probabilistic Selection task, which provides different stimulus-cue pairs in the no-feedback phase to the feedback phase. This additional information outweighed the hindrance caused by the lack of change to the model action choice probabilities during this phase. With no feedback there is no reward prediction error, so the models do not update. However, people are known to update their reward expectations even when not provided with feedback (Lieberman et al., 2001). This fitting consideration is separate from the discussion concerning the possibility that there might be multiple learning models or policies being used simultaneously, each of which dominate under different circumstances, such as when certain kinds of feedback (corrective, rewarding etc.) are provided or withheld (Frank et al., 2007). If such cases are being investigated then the feedback and no feedback portions should be fitted separately, with the acceptance that there will be an impact on the accuracy of the parameters recovered.

9.2 Model performance at fitting participant data

The models discussed and developed in chapters 3 and 4.8 were fitted to the participant data from all three tasks. The Weather task and Probabilistic Selection tasks used as their Bayes factor comparison the pure random model, described in chapter 3.12, which treats all possible actions as having the same probability of being chosen and consequently has no free parameters. The Decks task used the biased random model, also described in chapter 3.12, which assumes that the probabilities of actions are the same across trials, but vary across actions. The action choice probabilities are recovered for each participant from their action choice trial frequencies, resulting in $\mathcal{D} - 1$ free parameters.

The model fit qualities were similar for models that had many of the same components. The OpAL models performed poorly relative to the other models in all three tasks. The qLearn model was one of the best performing in all three tasks. The Bayesian models, especially BPV, performed very well on both the Weather task and the Probabilistic Selection task. This suggests that when compared to reinforcement learning models with identical parameters the Bayesian models, which store more information, can perform better than basic reinforcement learning models. That this was not the case for the Decks task may be due to these models taking longer to adapt to changes in the task reward probabilities.

While the epsilon greedy variants of models provided fewer edge fits, there was no indication in any of the datasets that epsilon greedy provided better evidence than those models using softmax, despite the previously discussed improved accuracy in recovering parameters from epsilon greedy versions of the models.

Participant fit quality values were found to be highly correlated between models, in Figure 6-10, Figure 7-11 and Figure 8-9, with some participants consistently being well fitted and others consistently worse fitted. This could suggest that some participants were using a model similar to several of the models being fitted in this thesis, while other participants were using models which was quite distinct from all the models examined in this research.

In each of the three reward prediction tasks, a task performance measure has been correlated with the model fit Bayes factors of most models, as seen in Figure 7-15, Figure 8-13 and in the discussion surrounding Figure 6-14 and Figure 6-15. For the Weather and Probabilistic Selection tasks only the most successful participants were the most well captured, whereas for the Decks task, the most successful participants were the least well captured. This suggests that there are elements of the participant behaviour and strategies that are not encapsulated by these models in their current form.

This analysis would need to be extended to a greater range of tasks before strong conclusions could be drawn on the relative performance of models. For example, none of these tasks provided an opportunity for models, such as those based on temporal difference learning, to demonstrate the value of their additional features above those of a simple qLearn model. With the TTpy framework that has been developed, extending this analysis will only require fitting the collected data for any new tasks.

Across all the tasks, the ε-greedy parameters were found to be recovered consistently across all the models and much more consistently than the β parameter from equivalent models using softmax. The only consistently recovered β parameters were those of the Q-learning variants. α parameters were inconsistently recovered, with the best recoveries from the critic learning rates. The correlation between participant recovered α parameters was found to be related to the type of model, as had been seen with the fit quality.

If the correlations between model parameters were reduced, it is likely that the correlations of the same parameters in different models would improve. One method that has been proposed to do so is the Bayesian Expectation-Maximisation method of calculating a fit (Huys et al., 2011). Another approach would be to fit against the free energy as explained by Klaas Enno Stephan et al. (2009).

To strengthen the model comparisons for each participant, *Bayesian Model Averaging* (BMA) could be used (Hoeting, Madigan, Raftery, & Volinsky, 1999). This model probability measure uses the best fit quality measure from each model, weighted by the model probability given the participant's actions. It therefore provides a more balanced perspective on the relative likelihood that each model best encapsulates the participant's actions. However, this requires integrating over each model-data parameter space to estimate the probability that the data would be produced by a model give the possible parameter combinations. This calculation can be approximated in conjunction with MCMC (Fragoso, Bertoli, & Louzada, 2018) or more directly with Variational Bayes (Beal, 2003; Daunizeau et al., 2009), necessitating a change in the fitting routines used before it can be considered. The changes necessary to use these techniques were discussed in chapter 9.1.

A more formal combined approach to both the recovery of accurate model parameters and the evaluation of the relative performance of models is to use Hierarchical Bayesian inference (HBI) (Piray, Dezfouli, Heskes, Frank, & Daw, 2019). This builds upon the Bayesian model selection used in this thesis, discussed in chapter 2.4, fitting participants with the Variational Bayes method. Like maximum a-posteriori (MAP) fitting, discussed in chapter 4.6, this uses a prior distribution to help fit participant data. Unlike MAP, HBI does not assume that all participants use the same model, an assumption that distorts the prior probability distribution. Instead, HBI repeatedly recreates the prior distribution for each model parameter based on the parameter values from participant's likely to have been using that model. The probability that a given participant used each model under consideration is then recalculated, which in turn is used to calculate model weights and the next set of parameter prior distributions. This continues until convergence. As with BMA, this would necessitate changing the fitting routines and rethinking how the fitting of multiple models is performed in the TTpy framework.

As has been stressed across this thesis, the comparative performance of the models reflects only their performance as implemented. There may be other implementations of the same models that perform better or worse with slight tweaks that do not change their principal features. These tweaks could be as simple as changing the starting action-stimulus cue reward values or a parameter's upper bounds. Alongside the analysis in chapter 4.1, an informal exploration of how the upper bound of β affected parameter recovery for the Decks task did show some small variations in recovered parameters but no clear improvement.

Furthermore, all the models here had their learning of expected rewards organised per action-cue pair, hindering the capture of any relationships between cues, such as those found in the Weather task. While encodings, such as conjunction coding, have been used for Weather task variants (Shohamy, Myers, Kalanithi, & Gluck, 2008), the version of the task used here was constructed such that the probability of an action being the correct one could be entirely captured by a linear combination of the predictive properties of each cue present: i.e. the conjunction of cues did not signify anything more than the sum of the individual cues present. Nonetheless, it is possible that changing this method of storing the expected feedback would change the capacity of the models to identify task features, thereby providing scope for more participant learning features to be captured. This could potentially be achieved by drawing upon the work on representation learning (Niv, 2019) or Semantic pointers (Eliasmith, 2013).

While epsilon greedy does result in more consistently recovered model parameters, it does have some less desirable properties, such as a lack of differentiability of the function, unlike softmax. Several alternatives to softmax and e-greedy have been proposed and should be explored, notably Mellowmax (Asadi & Littman, 2017) , e-softmax (Nassar & Frank, 2016), Value-Difference Based Exploration (Tokic, 2010; Tokic & Palm, 2011) and those discussed in the review paper by Schulz & Gershman (2019). These can be easily incorporated as model variants into the TTpy framework and compared to the current models.

Equally, it would have been informative to match every softmax model with an epsilon greedy equivalent, notably qLearnF and tdr are missing an epsilon greedy equivalent and ACES is missing a softmax equivalent. The legibility of the results would have been improved if the implementations of the softmax based OpAL models had a similar form to that of ε and ρ in the OpALE models, as presented by Collins & Frank (2014). This would allow a like-for-like comparison of the impact of softmax and epsilon greedy on parameter recovery in the OpAL models.

One limitation in the models was the assumption that all participants experienced the rewards in the same way. Participants' experience of rewards was also assumed to not change as the task progressed. This seemed a reasonable assumption as the tasks examined were relatively short and involved a relatively small number of rewards. For the Weather and Probabilistic Selection tasks, the nominal rewards were fixed to a constant reward per trial. It was also assumed that for the Decks task, where the rewards ranged from 1 to 10, the magnitude of the rewards as experienced was a simple linear function of the points won on each trial. Pickering & Pesola (2014) explored a reinforcement learning model in which the size of the experienced reward could vary across participants. The accurate recovery of associated parameters results in subtle issues of distinguishability with those of the learning rate (Pickering & Powell, unpublished observations). The capacity to modulate the reward, or more generally change the utility function (Ludvig, Madan, McMillan, Xu, & Spetch, 2018) was built into the TTpy framework in such a way that it could be easily incorporated into any model without modifying them. The analysis will be performed beyond the work of this thesis.

Another modification to the models that could improve model fits is to reduce the precision and range of values within the models by only using fixed point numbers. Fixed point calculations have been used successfully in DNNs (Courbariaux & David, 2015; N. Wang, Choi, Brand, Chen, & Gopalakrishnan, 2018) and in some spiking neuron models, such as the SpiNNaker Project (Furber, Galluppi, Temple, & Plana, 2014). It is possible that the artefacts resulting from lower precision calculations, and parameter values, could result in better descriptions of the participant exploratory behaviour.

9.3 LEARNING RATE RELATIONSHIP WITH EXTRAVERSION

Based on prior indications that the recovered participant learning rate parameter α could be linked to extraversion (Pickering & Pesola, 2014), an examination was made of their relationship. To reduce the noise from parameter recovery discussed in chapter 9.1, we chose to take the novel step of aggregating together the recovered α parameters from the Q-learning models with only one learning rate. This subset of models was found to have the most consistent recovered parameters. The models used were: qLearnCorr, qLearnECorr, qLearnE, qLearnF, qLearn, td0, tdE and tdr. This was then used in a comparison with the EPQ-R extraversion measure available for the participants from all the tasks. As strong correlations had been found between α and β, and weaker correlations between α and ε, partial correlations were also performed between the learning rate and extraversion. Finally, equivalent correlations were calculated for an average learning rate calculated only from the recovered parameters that were not edge fits and whose model fit Bayes factor was over 20.

Across the three datasets, the strongest correlations, and the only Bonferroni corrected significant correlations, were found for the recovered learning rates from participants who were well fitted by at least one of the models that used softmax. This negative partial correlation can also be seen when the recovered parameters and extraversion scores from all three datasets are concatenated and analysed together, as shown in Table 9-1. The consistency of this significant correlation despite poor model recovery for two of the task datasets suggests that it should be explored further.

This result points to extraversion being correlated with decreasing sensitivity to errors in expected rewards, which would result in extraverts learning more slowly than introverts for the same RPE. If this were the case, in probabilistic rewarding tasks such as the ones examined here, it would suggest that extraverts would be less sensitive to the reward variability and more able to identify changes in average reward values than introverts. From this, extroverts would be expected to perform better in tasks where there is greater variability in feedback.

Participant model fit parameters used	*All*		*Good edge & Bayes*	
	ρ (DF)	*p*	*ρ (DF)*	*p*
Q-learning models	-0.006 (466)	0.902	-0.068 (244)	0.291
Q-learning models with β	-0.053 (463)	0.250	-0.207 (222)	0.002
Q-learning models with ε	0.058 (463)	0.209	-0.010 (207)	0.890

Table 9-1 Using the data from all the task datasets, the correlations between each participant's averaged Q-learning model parameter α and the extraversion measure of EPQ-R. The α, β and ε values in these correlations are averaged, for each participant, across the relevant subset of models. The β and ε subset correlations were partial correlations whereas the others were simple correlations. As the data from the Weather and Probabilistic Selection tasks were poorly fitted, just over half of the "Good edge and Bayes" participants came from the Decks task datasets.

Pickering & Pesola (2014) suggest that α could correspond to the density of certain kinds of dopamine receptors controlling dopaminergic-mediated reinforcement learning. Given the negative correlation between extraversion and α and the positive correlation between extraversion and RPE magnitude, the impact on learning remains unclear.

Across all three tasks, only one measure of task performance correlated with extraversion: choose A in the Probabilistic Selection task. Without modelling, the correlation between extraversion and the learning rate, as expressed through α, would not be identifiable. This illustrates the potential value of model-based analyses of tasks over standard simple performance measures.

9.4 EXTENSIONS OF THE COMPUTATIONAL FRAMEWORK

This model evaluation framework sets the groundwork for future evaluation of more complex models and tasks. It has been written with a great amount of flexibility, much of it unused in these tasks. For example, arbitrary numbers of actions and stimulus cues are allowed, and no restrictions are imposed on the forms of the stimuli or feedback, allowing this to be used for both continuous time and instantaneous trialsteps, although the current model implementations are written for instantaneous trialsteps. The models also have the capacity to be extended to adapt to the introduction of new stimulus cues, actions or feedback elements during a task. Equally it can be used to fit competitive tasks and, with a little tweaking, simulate those tasks as well, such as the Dice decision task (Schulze, van Ravenzwaaij, & Newell, 2015).The task themselves can have a mix of forms of feedback, notably a reward, the correct answer, no feedback or the next trial state. While the SciPy gradient descent and evolutionary fitting methods have been used throughout this thesis, the framework allows for any suitable fitting library to be used as a backend.

This framework was initially written at a time when Python 3 was not fully supported by many Python libraries. As of 2020, Python 2.7 will not be supported by most Python libraries, including those used in this framework. An urgent task will be to modify the code to work with Python 3. Much of the code has been written with this transition in mind, so this is not expected to be difficult.

It is hoped that by publishing this research framework, researchers will be able to use, and reference, verified, preregistered, versions of common models that can be compared to their new model ideas (M. D. Lee et al., 2019). This could be enhanced by using standard task datasets that provide known baselines of model performance.

Several models are planned to be included in the suite of available models, notably more Bayesian models (Hampton et al., 2006; Mathys et al., 2011; Solway & Botvinick, 2012), other reinforcement learning models (Alexander & Brown, 2011; Steingroever, Wetzels, & Wagenmakers, 2016) and drift diffusion models

(Fontanesi, Gluth, Spektor, & Rieskamp, 2019; Pedersen, Frank, & Biele, 2016) among others.

The analysis of the performance of models fitting to a task or dataset could be extended by integrating tools such as the VBA toolbox (Daunizeau et al., 2014) or other model comparative methods such as (Piray et al., 2019).

Models themselves could also have their brain-complexity quantified. This could be achieved, for example, by evaluating the complexity of representing a model in a spiking neural network.

The scripting interface creates a high knowledge bar for using this. A graphical interface would make the framework more accessible without removing access to the current text interface. Initially this could be produced for simple tasks such as running simulations and fitting data that do not require additions to the framework. Another approach would be to create task translators, such that tasks written in OpenAI Gym (Brockman et al., 2016) or PsychoPy (Peirce et al., 2019) can be performed by the encoded models, allowing a more direct match between the forms in which tasks are typically written and those of the framework.

Finally, the framework has been written in such a way that multiple models could be used concurrently in a task, either united in some form of policy based model, or separately in cooperative or adversarial tasks (Buşoniu, Babuška, & De Schutter, 2010; Hunt, 2012; Littman, 1994; Silva & Costa, 2019). The latter would allow for another approach to measuring the validity of models by evaluating emergent task state properties to those found in real situations.

9.5 Recommendations for improving identifiability from tasks

Based on the conclusions of this thesis, there are some areas in the design of repeated decision-making tasks that have been identified as impacting on participant data model fitting and analysis. From these, a series of recommendations can be drawn:

- Providing more than two choices to participants during each trial will help distinguish between an expected optimal choice and a random choice or exploratory choice. This will allow for more insight into participant's thinking without increasing the number of trials.
- Task test phases, where participant's learnt preferences are examined without feedback, only provide a benefit if the trials contain novel variations of previously learnt relationships. Unusual probabilistic rewards at the end of a learning phase can result in a misunderstanding, or suboptimal encoding, of action-stimulus-cue relationships that will continue throughout the trials without feedback. Novel trials provide new insights into the participant's perceived action-stimulus-cue relationships and task decision making processes that are less affected by unusual final feedback.
- The identifiability of participant behaviour can be improved by increasing the number of trials in the learning phase of a task. In the Probabilistic Selection task 15 trials character-pair combination was found to be too little for most participants to learn which to choose, as was 14 trials per stimulus cue in the Weather task. In the decks task, where the stimuli and action choices were the same throughout, participant's proportion of choices for the initially advantageous choice peaked at around 25 trials. A good starting point would therefore be a minimum of 25 trials for each stimulus. More may be necessary for tasks where there are more than two choices per trial. As was seen in chapter 4, this will not be a sufficient number of trials to recover model parameters accurately with the methods used here, only to allow participants to begin to express their learnt action preferences. Furthermore, increasing the number of trials needs to be balanced with the risk that a participant's concentration slips or they become disengaged with

the task, especially for long tasks with similar trials. Such disengagement would result in the participant no longer using the model they had previously used, and so making the fitting of the original model more difficult.

- When a task is being designed, simulating potential participant behaviour provides a useful sanity check that participants will behave in the expected way, without the need for costly and time-consuming pilot studies. Through this, variations of a task can be quickly compared using conventional measures of task performance, such as points scored or ratios of chosen actions. These simulated participants can also be fitted to examine how the different task design choices impact model parameter recoverability. As the purpose of this assessment is not to optimise a task for a given model, the model simulation and subsequent fitting would need to be performed with only one model. Ideally, the model should be as simple as possible, i.e. have as few parameters and features, while still able to learn the necessary task action preferences. For example, for the Decks, Biased coins and Probabilistic Selection tasks the qLearn model would be a good candidate.
- For individual differences researchers, this thesis provides some encouragement that fitting formal models to participant data is a viable method for finding relationships with traits that cannot be seen with traditional ways of scoring performance. Here, a link was tentatively drawn between extraversion and the recovered learning rate parameter from some Q-learning models, which was not evident from the traditional ways of scoring learning performance.

References

Ahn, W. Y., Vasilev, G., Lee, S. H., Busemeyer, J. R., Kruschke, J. K., Bechara, A., & Vassileva, J. (2014). Decision-making in stimulant and opiate addicts in protracted abstinence: Evidence from computational modeling with pure users. *Frontiers in Psychology*, *5*(AUG), 1–15. https://doi.org/10.3389/fpsyg.2014.00849

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, *19*(6), 716–723. https://doi.org/10.1109/TAC.1974.1100705

Alexander, W. H., & Brown, J. W. (2011). Medial prefrontal cortex as an action-outcome predictor. *Nature Neuroscience*, *14*(10), 1338–1344. https://doi.org/10.1038/nn.2921

Asadi, K., & Littman, M. L. (2017). An Alternative Softmax Operator for Reinforcement Learning. In D. Precup & Y. W. Teh (Eds.), *Proceedings of Machine Learning Research* (Vol. 70, pp. 243–252). PMLR. Retrieved from http://proceedings.mlr.press/v70/asadi17a.html

Beal, M. (2003). *Variational algorithms for approximate Bayesian inference*. Retrieved from http://www.cse.buffalo.edu/faculty/mbeal/papers/beal03.pdf

Bechara, A., Damasio, A. R., Damasio, H., & Anderson, S. W. (1994). Insensitivity to future consequences following damage to human prefrontal cortex. *Cognition*, *50*(1–3), 7–15. https://doi.org/10.1016/0010-0277(94)90018-3

Behrens, T. E. J., Woolrich, M. W., Walton, M. E., & Rushworth, M. F. S. (2007). Learning the value of information in an uncertain world. *Nature Neuroscience*, *10*(9), 1214–1221. https://doi.org/10.1038/nn1954

Beierholm, U. R., Anen, C., Quartz, S., & Bossaerts, P. (2011). Separate encoding of model-based and model-free valuations in the human brain. *NeuroImage*, *58*(3), 955–962. https://doi.org/10.1016/j.neuroimage.2011.06.071

Box, G. E. P. (1979). Robustness in the Strategy of Scientific Model Building. In R. L.

Launer & G. N. Wilkinson (Eds.), *Robustness in Statistics* (pp. 201–236). Academic Press. https://doi.org/10.1016/B978-0-12-438150-6.50018-2

Brandl, G., Ronacher, A., Shimizukawa, T., Neuhäuser, D., Waltman, J., Ruana, R., … Kampik, T. (2018). Sphinx. Retrieved from http://www.sphinx-doc.org

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). OpenAI Gym, 1–4. Retrieved from http://arxiv.org/abs/1606.01540

Brown, K. S., & Sethna, J. P. (2003). Statistical mechanical approaches to models with many poorly known parameters. *Physical Review E - Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics*, *68*(2), 9. https://doi.org/10.1103/PhysRevE.68.021904

Burnham, K. P., & Anderson, D. R. (2004). Multimodel Inference. *Sociological Methods & Research*, *33*(2), 261–304. https://doi.org/10.1177/0049124104268644

Busemeyer, J. R., & Stout, J. C. (2002). A contribution of cognitive decision models to clinical assessment: Decomposing performance on the Bechara gambling task. *Psychological Assessment*, *14*(3), 253–262. https://doi.org/10.1037//1040-3590.14.3.253

Buşoniu, L., Babuška, R., & De Schutter, B. (2010). Multi-agent reinforcement learning: An overview. In D. Srinivasan & L. C. Jain (Eds.), *Innovations in Multi-Agent Systems and Applications – 1* (Studies in, Vol. 310, pp. 183–221). Berlin, Germany: Springer. https://doi.org/10.1007/978-3-642-14435-6_7

Byrd, R. H., Lu, P., Nocedal, J., & Zhu, C. (1994). A Limited-Memory Algorithm for Bound Constrained Optimization. *SIAM Journal on Scientific Computing*, *16*, 1190–1208. https://doi.org/10.1.1.15.7343

Cafferkey, K., Murphy, J., & Shevlin, M. (2013). Jumping to conclusions: the association between delusional ideation and reasoning biases in a healthy student population. *Psychosis*, (February 2014), 1–9. https://doi.org/10.1080/17522439.2013.850734

Carpenter, B., Gelman, A., Hoffman, M. D., Lee, D., Goodrich, B., Betancourt, M., … Riddell, A. (2017). Stan : A Probabilistic Programming Language. *Journal of Statistical Software*, *76*(1). https://doi.org/10.18637/jss.v076.i01

Collins, A. G. E., & Frank, M. J. (2014). Opponent actor learning (OpAL): Modeling interactive effects of striatal dopamine on reinforcement learning and choice incentive. *Psychological Review*, *121*(3), 337–366. https://doi.org/10.1037/a0037015

Cooper, A. J., Duke, E., Pickering, A. D., & Smillie, L. D. (2014). Individual differences in reward prediction error: contrasting relations between feedback-related negativity and trait measures of reward sensitivity, impulsivity and extraversion. *Frontiers in Human Neuroscience*, *8*(April), 248. https://doi.org/10.3389/fnhum.2014.00248

Courbariaux, M., & David, J. (2015). Raining deep neural networks with low precision multiplications, (Section 5), 1–10.

Daunizeau, J., Adam, V., & Rigoux, L. (2014). VBA: A Probabilistic Treatment of Nonlinear Models for Neurobiological and Behavioural Data. *PLoS Computational Biology*, *10*(1). https://doi.org/10.1371/journal.pcbi.1003441

Daunizeau, J., Friston, K. J., & Kiebel, S. J. (2009). Variational Bayesian identification and prediction of stochastic nonlinear dynamic causal models. *Physica D: Nonlinear Phenomena*, *238*(21), 2089–2118. https://doi.org/10.1016/j.physd.2009.08.002

Daw, N. D. (2011). Trial-by-trial data analysis using computational models. In *Decision Making, Affect, and Learning: Attention and Performance XXIII* (pp. 3–38). Oxford University Press. https://doi.org/10.1093/acprof:oso/9780199600434.003.0001

Daw, N. D., & Doya, K. (2006). The computational neurobiology of learning and reward. *Current Opinion in Neurobiology*, *16*(2), 199–204. https://doi.org/10.1016/j.conb.2006.03.006

Daw, N. D., O'Doherty, J. P., Dayan, P., Dolan, R. J., Seymour, B., Dolan, R. J., & Seymour, B. (2006). Cortical substrates for exploratory decisions in humans.

Nature, *441*(7095), 876–879. https://doi.org/10.1038/nature04766

Daw, N. D., & Touretzky, D. S. (2002). Long-term reward prediction in TD models of the dopamine system. *Neural Computation*, *14*, 2567–2583. https://doi.org/10.1162/089976602760407973

Dayan, P., & Berridge, K. C. (2014). Model-based and model-free Pavlovian reward learning: revaluation, revision, and revelation. *Cognitive, Affective & Behavioral Neuroscience*, *14*(2), 473–492. https://doi.org/10.3758/s13415-014-0277-8

Depue, R. A., & Collins, P. F. (1999). Neurobiology of the structure of personality: Dopamine, facilitation of incentive motivation, and extraversion. *Behavioral and Brain Sciences*, *22*(3), 491–517. https://doi.org/10.1017/S0140525X99002046

Eglen, S., Marwick, B., Halchenko, Y., Hanke, M., Sufi, S., Gleeson, P., … Poline, J.-B. (2016). Towards standard practices for sharing computer code and programs in neuroscience. *BioRxiv*, *20*(6), 045104. https://doi.org/10.1101/045104

Eliasmith, C. (2013). *How to Build a Brain: A Neural Architecture for Biological Cognition*. Oxford University Press. https://doi.org/10.1093/acprof:oso/9780199794546.001.0001

Eppinger, B., Kray, J., Mock, B., & Mecklinger, A. (2008). Better or worse than expected? Aging, learning, and the ERN. *Neuropsychologia*, *46*(2), 521–539. https://doi.org/10.1016/j.neuropsychologia.2007.09.001

Eysenck, H. J. (1975). *Manual of the Eysenck Personality Questionnaire*. Hodder and Stoughton.

Eysenck, S. B. G., Eysenck, H. J., & Barrett, P. (1985). A revised version of the psychoticism scale. *Personality and Individual Differences*, *6*(1), 21–29. https://doi.org/10.1016/0191-8869(85)90026-1

Faragher, R. (2012). Understanding the Basis of the Kalman Filter Via a Simple and Intuitive Derivation [Lecture Notes]. *IEEE Signal Processing Magazine*, *29*(5), 128–132. https://doi.org/10.1109/MSP.2012.2203621

Fontanesi, L., Gluth, S., Spektor, M. S., & Rieskamp, J. (2019). A reinforcement learning diffusion decision model for value-based decisions. *Psychonomic Bulletin & Review*. https://doi.org/10.3758/s13423-018-1554-2

Fragoso, T. M., Bertoli, W., & Louzada, F. (2018). Bayesian Model Averaging: A Systematic Review and Conceptual Classification. *International Statistical Review*, *86*(1), 1–28. https://doi.org/10.1111/insr.12243

Frank, M. J., Moustafa, A. a, Haughey, H. M., Curran, T., & Hutchison, K. E. (2007). Genetic triple dissociation reveals multiple roles for dopamine in reinforcement learning. *Proceedings of the National Academy of Sciences of the United States of America*, *104*(41), 16311–16316. https://doi.org/10.1073/pnas.0706111104

Frank, M. J., Seeberger, L. C., & O'Reilly, R. C. (2004). By Carrot or by Stick: Cognitive Reinforcement Learning in Parkinsonism. *Science*, *306*(5703), 1940–1943. https://doi.org/10.1126/science.1102941

Friel, N., McKeone, J. P., Oates, C. J., & Pettitt, A. N. (2017). Investigation of the widely applicable Bayesian information criterion. *Statistics and Computing*, *27*(3), 833–844. https://doi.org/10.1007/s11222-016-9657-y

Furber, S. B., Galluppi, F., Temple, S., & Plana, L. A. (2014). The SpiNNaker project. *Proceedings of the IEEE*, *102*(5), 652–665. https://doi.org/10.1109/JPROC.2014.2304638

Gershman, S. J. (2016). Empirical priors for reinforcement learning models. *Journal of Mathematical Psychology*, *71*, 1–6. https://doi.org/10.1016/j.jmp.2016.01.006

Glimcher, P. W. (2011). Understanding dopamine and reinforcement learning: the dopamine reward prediction error hypothesis. *Proceedings of the National Academy of Sciences of the United States of America*, *108 Suppl*, 15647–15654. https://doi.org/10.1073/pnas.1014269108

Gluck, M. A., & Bower, G. H. (1988). From Conditioning to Category Learning: An Adaptive Network Model. *Journal of Experimental Psychology: General*, *117*(3), 227–247. https://doi.org/10.1037/0096-3445.117.3.227

Gray, J. A. (1970). The psychophysiological basis of introversion-extraversion.

Behaviour Research and Therapy, *8*(3), 249–266. https://doi.org/10.1016/0005-7967(70)90069-0

Gureckis, T. M., & Love, B. C. (2009). Short-term gains, long-term pains: How cues about state aid learning in dynamic environments. *Cognition*, *113*(3), 293–313. https://doi.org/10.1016/j.cognition.2009.03.013

Halpern, D., & Gureckis, T. M. (2013). Thinking on Thinking - The Gureckis Lab blog: On the identifiability of parameters in reinforcement learning models. Retrieved May 18, 2017, from http://gureckislab.org/blog/?p=3450

Hampton, A. N., Bossaerts, P., & O'Doherty, J. P. (2006). The Role of the Ventromedial Prefrontal Cortex in Abstract State-Based Inference during Decision Making in Humans. *Journal of Neuroscience*, *26*(32), 8360–8367. https://doi.org/10.1523/JNEUROSCI.1010-06.2006

Haykin, S. (2009). *Neural Networks and Learning Machines* (Third). Pearson Prentice Hall.

Heathcote, A., Brown, S. D., & Wagenmakers, E.-J. (2015). An introduction to good practices in cognitive modeling. *An Introduction to Model-Based Cognitive Neuroscience*, 25–48. https://doi.org/10.1007/978-1-4939-2236-9_2

Hoeting, J. A., Madigan, D., Raftery, A. E., & Volinsky, C. T. (1999). Bayesian model averaging: A tutorial. *Statistical Science*, *14*(4), 382–401. https://doi.org/10.1214/ss/1009212519

Holroyd, C. B., & Coles, M. G. H. (2002). The neural basis of human error processing: Reinforcement learning, dopamine, and the error-related negativity. *Psychological Review*, *109*(4), 679–709. https://doi.org/10.1037//0033-295X.109.4.679

Humphries, M. A., Bruno, R., Karpievitch, Y., & Wotherspoon, S. (2015). The expectancy valence model of the iowa gambling task: Can it produce reliable estimates for individuals? *Journal of Mathematical Psychology*, *64–65*, 17–34. https://doi.org/10.1016/j.jmp.2014.10.002

Hunt, D. (2012). *An exploration of harvesting strategies in a spatial dynamic game*.

University of Gothenburg.

Huys, Q. J. M., Cools, R., Gölzer, M., Friedel, E., Heinz, A., Dolan, R. J., & Dayan, P. (2011). Disentangling the roles of approach, activation and valence in instrumental and pavlovian responding. *PLoS Computational Biology*, *7*(4). https://doi.org/10.1371/journal.pcbi.1002028

Jones, M., & Love, B. C. (2011). Bayesian Fundamentalism or Enlightenment? On the explanatory status and theoretical contributions of Bayesian models of cognition. *Behavioral and Brain Sciences*, *34*(4), 169–188. https://doi.org/10.1017/S0140525X10003134

Kass, R., & Raftery, A. E. (1995). Bayes Factors. *Journal of the American Statistical Association*. https://doi.org/10.1080/01621459.1995.10476572

Kirsch, I., Lynn, S. J., Vigorito, M., & Miller, R. R. (2004). The Role of Cognition in Classical and Operant Conditioning. *Journal of Clinical Psychology*, *60*(4), 369–392. https://doi.org/10.1002/jclp.10251

Knill, D. C., & Pouget, A. (2004). The Bayesian brain: the role of uncertainty in neural coding and computation. *Trends in Neurosciences*, *27*(12), 712–719. https://doi.org/10.1016/j.tins.2004.10.007

Knowlton, B. J., Squire, L. R., & Gluck, M. A. (1994). Probabilistic classification learning in amnesia. *Learning & Memory (Cold Spring Harbor, N.Y.)*, *1*(2), 106–120. https://doi.org/10.1101/lm.1.2.106

Kraft, D. (1988). A Software Package for Sequential Quadratic Programming.

Krekel, H. (2017). pytest. Retrieved from https://docs.pytest.org

Lak, A., Stauffer, W. R., & Schultz, W. (2014). Dopamine prediction error responses integrate subjective value from different reward dimensions. *Proceedings of the National Academy of Sciences*, *111*(6), 2343–2348. https://doi.org/10.1073/pnas.1321596111

Lau, B., & Glimcher, P. W. (2005). Dynamic Response-by-Response Models of Matching Behavior in Rhesus Monkeys. *Journal of the Experimental Analysis of Behavior*, *84*(3), 555–579. https://doi.org/10.1901/jeab.2005.110-04

Lee, M. D., Criss, A. H., Devezer, B., Donkin, C., Etz, A., Leite, F. P., … Vandekerckhove, J. (2019). Robust Modeling in Cognitive Science. *Computational Brain & Behavior*. https://doi.org/10.1007/s42113-019-00029-y

Lee, S. W., Shimojo, S., & O'Doherty, J. P. (2014). Neural Computations Underlying Arbitration between Model-Based and Model-free Learning. *Neuron*, *81*(3), 687–699. https://doi.org/10.1016/j.neuron.2013.11.028

Legendre, P. (2010). Coefficient of Concordance. In *Encyclopedia of Research Design* (pp. 164–169). 2455 Teller Road, Thousand Oaks California 91320 United States: SAGE Publications, Inc. https://doi.org/10.4135/9781412961288.n55

Lieberman, M. D., Ochsner, K. N., Gilbert, D. T., & Schacter, D. L. (2001). Do Amnesics Exhibit Cognitive Dissonance Reduction? The Role of Explicit Memory and Attention in Attitude Change. *Psychological Science*, *12*(2), 135–140. https://doi.org/10.1111/1467-9280.00323

Littman, M. L. (1994). Markov games as a framework for multi-agent reinforcement learning. In *Machine Learning Proceedings 1994* (pp. 157–163). Brunswick, NJ: Elsevier. https://doi.org/10.1016/B978-1-55860-335-6.50027-1

Ludvig, E. A., Madan, C. R., McMillan, N., Xu, Y., & Spetch, M. L. (2018). Living near the edge: How extreme outcomes and their neighbors drive risky choice. *Journal of Experimental Psychology: General*, *147*(12), 1905–1918. https://doi.org/10.1037/xge0000414

Mathys, C., Daunizeau, J., Friston, K. J., & Stephan, K. E. (2011). A bayesian foundation for individual learning under uncertainty. *Frontiers in Human Neuroscience*, *5*(May), 39. https://doi.org/10.3389/fnhum.2011.00039

Matsumoto, M., Matsumoto, K., Abe, H., & Tanaka, K. (2007). Medial prefrontal cell activity signaling prediction errors of action values. *Nature Neuroscience*, *10*(5), 647–656. https://doi.org/10.1038/nn1890

McCrae, R. R., & Allik, J. (Eds.). (2002). *The Five-Factor Model of Personality Across Cultures*. Boston, MA: Springer US. https://doi.org/10.1007/978-1-4615-0763-5

Millman, K. J., & Aivazis, M. (2011). Python for Scientists and Engineers. *Computing* *in Science & Engineering*, *13*(2), 9–12. https://doi.org/10.1109/MCSE.2011.36

Moore, S. C., & Sellen, J. L. (2006). Jumping to conclusions: a network model predicts schizophrenic patients' performance on a probabilistic reasoning task. *Cognitive, Affective & Behavioral Neuroscience*, *6*(4), 261–269. https://doi.org/10.3758/CABN.6.4.261

Moran, R. (2016). Thou shalt identify! The identifiability of two high-threshold models in confidence-rating recognition (and super-recognition) paradigms. *Journal of Mathematical Psychology*, *73*, 1–11. https://doi.org/10.1016/j.jmp.2016.03.002

Nash, S. G. (1984). Newton-Type Minimization via the Lanczos Method. *SIAM Journal on Numerical Analysis*, *21*, 770–788. https://doi.org/10.1137/0721052

Nassar, M. R., & Frank, M. J. (2016). Taming the beast: Extracting generalizable knowledge from computational models of cognition. *Current Opinion in Behavioral Sciences*, *11*, 49–54. https://doi.org/10.1016/j.cobeha.2016.04.003

Niv, Y. (2019). Learning task-state representations. *Nature Neuroscience*, *22*(10), 1544–1553. https://doi.org/10.1038/s41593-019-0470-8

O'Doherty, J. P., Lee, S. W., & McNamee, D. (2015). The structure of reinforcement-learning mechanisms in the human brain. *Current Opinion in Behavioral Sciences*, *1*, 94–100. https://doi.org/10.1016/j.cobeha.2014.10.004

Oliphant, T. E. (2007). Python for Scientific Computing. *Computing in Science & Engineering*, *9*(3), 10–20. https://doi.org/10.1109/MCSE.2007.58

Pedersen, M. L., Frank, M. J., & Biele, G. (2016). The drift diffusion model as the choice rule in reinforcement learning. *Psychonomic Bulletin & Review*. https://doi.org/10.3758/s13423-016-1199-y

Peirce, J., Gray, J. R., Simpson, S., MacAskill, M., Höchenberger, R., Sogo, H., … Lindeløv, J. K. (2019). PsychoPy2: Experiments in behavior made easy. *Behavior Research Methods*, *51*(1), 195–203. https://doi.org/10.3758/s13428-018-01193-y

Piaget, J. (1937). *La construction du réel chez le̓nfant*. Delachaux et Niestlé.

Pickering, A. D. (2004). The Neuropsychology of Impulsive Antisocial Sensation

Seeking Personality Traits: From Dopamine to Hippocampal Function? In *On the Psychobiology of Personality* (pp. 453–476). Elsevier. https://doi.org/10.1016/B978-008044209-9/50024-5

Pickering, A. D. (2011). The psychobiology of major personality dimensions: From genetics to computational models. In *1st Meeting of the Latin-American Association for the Investigation of Individual Differences. Castellon, Spain*.

Pickering, A. D., & Pesola, F. (2014). Modeling dopaminergic and other processes involved in learning from reward prediction error: contributions from an individual differences perspective. *Frontiers in Human Neuroscience*, *8*(September), 740. https://doi.org/10.3389/fnhum.2014.00740

Piray, P., Dezfouli, A., Heskes, T., Frank, M. J., & Daw, N. D. (2019). Hierarchical Bayesian inference for concurrent model fitting and comparison for group studies. *PLoS Computational Biology*, *15*(6), e1007043. https://doi.org/10.1371/journal.pcbi.1007043

Pitt, M., Myung, I., & Zhang, S. (2002). Toward a method of selecting among computational models of cognition. *Psychological Review*, *109*(3), 472–491. https://doi.org/10.1037/0033-295X.109.3.472

Plonsky, O., Teodorescu, K., & Erev, I. (2015). Reliance on small samples, the wavy recency effect, and similarity-based learning. *Psychological Review*, *122*(4). https://doi.org/10.1037/a0039413

Poldrack, R. A., Feingold, F., Frank, M. J., Gleeson, P., Hollander, G. de, Huys, Q. J. M., … Cohen, J. D. (2019). The importance of standards for sharing of computational models and data.

Potts, G. F., Martin, L. E., Burton, P., & Montague, P. R. (2006). When things are better or worse than expected: the medial frontal cortex and the allocation of processing resources. *Journal of Cognitive Neuroscience*, *18*(7), 1112–1119. https://doi.org/10.1162/jocn.2006.18.7.1112

Raftery, A. E. (1995). Bayesian Model Selection in Social Research. *Sociological Methodology*, *25*, 111. https://doi.org/10.2307/271063

Rescorla, R. A., & Wagner, A. R. (1972). A theory of Pavlovian conditioning: variations in the effectiveness of reinforcement and nonreinforcement. In A. Black & W. F. Prokasy (Eds.), *Classical conditioning II: current research and theory* (pp. 64–99). Appleton-Century-Crofts, New York.

Reverdy, P., & Leonard, N. E. (2015). Parameter Estimation in Softmax Decision-Making Models With Linear Objective Functions. *IEEE Transactions on Automation Science and Engineering*, *13*(1), 54–67. https://doi.org/10.1109/TASE.2015.2499244

Rigoux, L., Stephan, K. E., Friston, K. J., & Daunizeau, J. (2014). Bayesian model selection for group studies - revisited. *NeuroImage*, *84*, 971–985. https://doi.org/10.1016/j.neuroimage.2013.08.065

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386–408. https://doi.org/10.1037/h0042519

Rosenblatt, F. (1961). *Perceptrons and the Theory of Brain Mechanics* (Vol. VG-1196-G).

Salomon, R. (1998). Evolutionary algorithms and gradient search: Similarities and differences. *IEEE Transactions on Evolutionary Computation*, *2*(2), 45–55. https://doi.org/10.1109/4235.728207

Salvatier, J., Wiecki, T. V., & Fonnesbeck, C. (2016). Probabilistic programming in Python using PyMC3. *PeerJ Computer Science*, *2*, e55. https://doi.org/10.7717/peerj-cs.55

Schmiedek, F., Oberauer, K., Wilhelm, O., Süss, H.-M., & Wittmann, W. W. (2007). Individual differences in components of reaction time distributions and their relations to working memory and intelligence. *Journal of Experimental Psychology. General*, *136*(3), 414–429. https://doi.org/10.1037/0096-3445.136.3.414

Schmittmann, V. D., Dolan, C. V., Raijmakers, M. E. J., & Batchelder, W. H. (2010). Parameter identification in multinomial processing tree models. *Behavior Research Methods*, *42*(3), 836–846. https://doi.org/10.3758/BRM.42.3.836

Schultz, W. (1998). Predictive Reward Signal of Dopamine Neurons. *Journal of Neurophysiology*, *80*(1), 1–27. https://doi.org/10.1152/jn.1998.80.1.1

Schultz, W. (2000). Multiple reward signals in the brain. *Nature Reviews. Neuroscience*, *1*(3), 199–207. https://doi.org/10.1038/35044563

Schultz, W. (2007). Multiple Dopamine Functions at Different Time Courses. *Annual Review of Neuroscience*, *30*(1), 259–288. https://doi.org/10.1146/annurev.neuro.28.061604.135722

Schultz, W. (2015). Neuronal Reward and Decision Signals: From Theories to Data. *Physiological Reviews*, *95*(3), 853–951. https://doi.org/10.1152/physrev.00023.2014

Schultz, W. (2016). Dopamine reward prediction- error signalling: a two-component response. *Nature Publishing Group*, *17*(3), 183–195. https://doi.org/10.1038/nrn.2015.26

Schultz, W., Dayan, P., & Montague, P. R. (1997). A neural substrate of prediction and reward. *Science (New York, N.Y.)*, *275*(5306), 1593–1599. https://doi.org/10.1126/science.275.5306.1593

Schultz, W., & Dickinson, A. (2000). Neuronal coding of prediction errors. *Annual Review of Neuroscience*, *23*, 473–500. https://doi.org/10.1146/annurev.neuro.23.1.473

Schulz, E., & Gershman, S. J. (2019). The algorithmic architecture of exploration in the human brain. *Current Opinion in Neurobiology*, *55*, 7–14. https://doi.org/10.1016/j.conb.2018.11.003

Schulze, C., van Ravenzwaaij, D., & Newell, B. R. (2015). Of matchers and maximizers: How competition shapes choice under risk and uncertainty. *Cognitive Psychology*, *78*, 78–98. https://doi.org/10.1016/j.cogpsych.2015.03.002

Schwarz, G. (1978). Estimating the Dimension of a Model. *The Annals of Statistics*, *6*(2), 461–464. https://doi.org/10.1214/aos/1176344136

Schweighofer, N., & Doya, K. (2003). Meta-learning in reinforcement learning. *Neural Networks*, *16*(1), 5–9. https://doi.org/10.1016/S0893-6080(02)00228-9

Shohamy, D., Myers, C. E., Kalanithi, J., & Gluck, M. A. (2008). Basal ganglia and dopamine contributions to probabilistic category learning. *Neuroscience & Biobehavioral Reviews*, *32*(2), 219–236. https://doi.org/10.1016/j.neubiorev.2007.07.008

Silva, F. L. Da, & Costa, A. H. R. (2019). A Survey on Transfer Learning for Multiagent Reinforcement Learning Systems. *Journal of Artificial Intelligence Research*, *64*, 645–703. https://doi.org/10.1613/jair.1.11396

Simon, J. R., Howard, J. H., & Howard, D. V. (2010). Adult Age Differences in Learning From Positive and Negative Probabilistic Feedback. *Neuropsychology*, *24*(4), 534–541. https://doi.org/10.1037/a0018652

Slagter, H. A., Georgopoulou, K., & Frank, M. J. (2015). Spontaneous eye blink rate predicts learning from negative, but not positive, outcomes. *Neuropsychologia*, *71*, 126–132. https://doi.org/10.1016/j.neuropsychologia.2015.03.028

Smillie, L. D. (2013). Extraversion and Reward Processing. *Current Directions in Psychological Science*, *22*(3), 167–172. https://doi.org/10.1177/0963721412470133

Smillie, L. D., Cooper, A. J., & Pickering, A. D. (2011). Individual differences in reward-prediction-error: Extraversion and feedback-related negativity. *Social Cognitive and Affective Neuroscience*, *6*, 646–652. https://doi.org/10.1093/scan/nsq078

Smillie, L. D., Jach, H. K., Hughes, D. M., Wacker, J., Cooper, A. J., & Pickering, A. D. (2019). Extraversion and reward-processing: Consolidating evidence from an electroencephalographic index of reward-prediction-error. *Biological Psychology*, *146*(September), 107735. https://doi.org/10.1016/j.biopsycho.2019.107735

Smillie, L. D., Pickering, A. D., & Jackson, C. J. (2006). The new reinforcement sensitivity theory: implications for personality measurement. *Personality and Social Psychology Review : An Official Journal of the Society for Personality and Social Psychology, Inc*, *10*(4), 320–335. https://doi.org/10.1207/s15327957pspr1004_3

Sojitra, R. B., Lerner, I., Petok, J. R., & Gluck, M. A. (2018). Age affects reinforcement learning through dopamine-based learning imbalance and high decision noise—not through Parkinsonian mechanisms. *Neurobiology of Aging*, *68*, 102–113. https://doi.org/10.1016/j.neurobiolaging.2018.04.006

Solway, A., & Botvinick, M. M. (2012). Goal-directed decision making as probabilistic inference: a computational framework and potential neural correlates. *Psychological Review*, *119*(1), 120–154. https://doi.org/10.1037/a0026435

Spektor, M. S., & Kellen, D. (2018). The relative merit of empirical priors in non-identifiable and sloppy models: Applications to models of learning and decision-making: Empirical priors. *Psychonomic Bulletin and Review*, *25*(6), 1–22. https://doi.org/10.3758/s13423-018-1446-5

Stankevicius, A., Huys, Q. J. M., Kalra, A., & Seriès, P. (2014). Optimism as a Prior Belief about the Probability of Future Reward. *PLoS Computational Biology*, *10*(5). https://doi.org/10.1371/journal.pcbi.1003605

Steingroever, H., Wetzels, R., & Wagenmakers, E.-J. (2016). Bayes factors for reinforcement-learning models of the Iowa gambling task. *Decision*, *3*(2), 115–131. https://doi.org/10.1037/dec0000040

Stephan, Klaas E., Marshall, J. C., Penny, W. D., Friston, K. J., & Fink, G. R. (2007). Interhemispheric integration of visual processing during task-driven lateralization. *Journal of Neuroscience*, *27*(13), 3512–3522. https://doi.org/10.1523/JNEUROSCI.4766-06.2007

Stephan, Klaas Enno, Penny, W. D., Daunizeau, J., Moran, R. J., & Friston, K. J. (2009). Bayesian model selection for group studies. *NeuroImage*, *46*(4), 1004–1017. https://doi.org/10.1016/j.neuroimage.2009.03.025

Storn, R., & Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 341–359. https://doi.org/10.1023/A:1008202821328

Strathern, M. (1997). 'Improving ratings': audit in the British University system. *European Review*, *5*(03), 305. https://doi.org/10.1017/s1062798700002660

Sun, R. (2008). Introduction to computational cognitive modeling. In *Cambridge handbook of computational psychology* (pp. 3–20). New York, NY, US: Cambridge University Press. https://doi.org/https://doi.org/10.1017/CBO9780511816772.003

Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine Learning*, *3*(1), 9–44. https://doi.org/10.1007/BF00115009

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement Learning : An Introduction*. *A Bradford Book*. https://doi.org/10.1109/TNN.1998.712192

Tokic, M. (2010). Adaptive ε-greedy exploration in reinforcement learning based on value differences. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *6359 LNAI*, 203–210. https://doi.org/10.1007/978-3-642-16111-7_23

Tokic, M., & Palm, G. (2011). Value-Difference Based Exploration: Adaptive Control between Epsilon-Greedy and Softmax. In J. Bach & S. Edelkamp (Eds.), *KI 2011: Advances in Artificial Intelligence* (Vol. 7006 LNAI, pp. 335–346). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-24455-1_33

Tsitsiklis, J. N., & Van Roy, B. (2002). On average versus discounted reward temporal-difference learning. *Machine Learning*, *49*(2–3), 179–191. https://doi.org/10.1023/A:1017980312899

Wacker, J., & Smillie, L. D. (2015). Trait Extraversion and Dopamine Function. *Social and Personality Psychology Compass*, *9*(6), 225–238. https://doi.org/10.1111/spc3.12175

Waltz, R. A., Morales, J. L., Nocedal, J., & Orban, D. (2006). An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Mathematical Programming*, *107*(3), 391–408. https://doi.org/10.1007/s10107-004-0560-5

Wang, J. X., Kurth-Nelson, Z., Kumaran, D., Tirumala, D., Soyer, H., Leibo, J. Z., … Botvinick, M. (2018). Prefrontal cortex as a meta-reinforcement learning system. *Nature Neuroscience*. https://doi.org/10.1038/s41593-018-0147-8

Wang, N., Choi, J., Brand, D., Chen, C. Y., & Gopalakrishnan, K. (2018). Training deep neural networks with 8-bit floating point numbers. *Advances in Neural Information Processing Systems*, *2018-Decem*(NeurIPS), 7675–7684.

Watanabe, S. (2012). A Widely Applicable Bayesian Information Criterion. *Machine Learning Research*, *14*(1), 867–897. Retrieved from http://www.jmlr.org/papers/v14/watanabe13a.html

Watkins, C. J. C. H. (1989). *Learning from Delayed Rewards*. Cambridge. Retrieved from http://www.cs.rhul.ac.uk/~chrisw/new_thesis.pdf

Wetzels, R., Matzke, D., Lee, M. D., Rouder, J. N., Iverson, G. J., & Wagenmakers, E.-J. (2011). Statistical Evidence in Experimental Psychology: An Empirical Comparison Using 855 t Tests. *Perspectives on Psychological Science*, *6*(3), 291–298. https://doi.org/10.1177/1745691611406923

Wetzels, Ruud, Vandekerckhove, J., Tuerlinckx, F., & Wagenmakers, E.-J. (2010). Bayesian parameter estimation in the Expectancy Valence model of the Iowa gambling task. *Journal of Mathematical Psychology*, *54*(1), 14–27. https://doi.org/10.1016/j.jmp.2008.12.001

Wilt, J., & Revelle, W. (2016). Extraversion. In T. A. Widiger (Ed.), *The Oxford Handbook of the Five Factor Model* (Vol. 1). New York, N.Y: Oxford University Press. https://doi.org/10.1093/oxfordhb/9780199352487.013.15

Worthy, D. a, Maddox, W. T., & Markman, A. B. (2007). Regulatory fit effects in a choice task. *Psychonomic Bulletin & Review*, *14*(6), 1125–1132. https://doi.org/10.3758/BF03193101

Ziegler, M., Rief, W., Werner, S.-M. M., Mehl, S., & Lincoln, T. M. (2008). Hasty decision-making in a variety of tasks: Does it contribute to the development of delusions? *Psychology and Psychotherapy: Theory, Research and Practice*, *81*(3), 237–245. https://doi.org/10.1348/147608308X297104

Zuckerman, M. (2005). *Psychobiology of Personality*. *Psychobiology of personality, 2nd ed., rev. & updated*. Cambridge: Cambridge University Press. https://doi.org/10.1017/CBO9780511813733

Appendix I. NOTATION SUMMARY

Symbol	
$\boldsymbol{K}$	The set of all models, $k \in K$
$\boldsymbol{\mathcal{N}}$	The set of participants who have performed a task sequence, $n \in \mathcal{N}$
$\boldsymbol{T}$	The set of all trialsteps, $t \in T$
Fitting values and functions	
$\boldsymbol{\mathcal{B}}$	The Bayes factor
$\boldsymbol{\mathcal{B}_{min}}$	The minimum Bayes factor considered to provide strong evidence
$\boldsymbol{BIC_{mod}}$	The Bayesian information criterion for the model
$\boldsymbol{BIC_{rand}}$	The Bayesian information criterion for the random model
$\boldsymbol{BIC_{diff}}$	The difference, $BIC_{rand} - BIC_{mod}$
$\boldsymbol{c_t}$	The chosen action at time t
$\boldsymbol{C}$	The sequence of actions, $c_1, c_2 \cdots c_T$
$\boldsymbol{d_t}$	An action available at time t
$\boldsymbol{\mathcal{D}}$	The set of actions available
$\boldsymbol{\mathcal{D}_t}$	The set of actions available at time t
$\lVert\boldsymbol{\mathcal{D}_t}\rVert$	The number of different actions available at time t
$\boldsymbol{EF}$	The expectation of the model frequencies
$\boldsymbol{f}$	The maximum likelihood value
$\boldsymbol{f_{\mathcal{B}}}$	The Normalised Bayes factor fit quality
$\boldsymbol{f}_{\mathrm{mod}}$	The maximum likelihood value for the model
$\boldsymbol{f}_{\mathrm{rand}}$	The maximum likelihood value for the random choices
$\boldsymbol{\Delta f}$	The difference between the maximum likelihood value of the random and model choices
$\boldsymbol{H}$	A hypothesis
$\boldsymbol{H}_{\mathrm{mod}}$	The hypothesis that the action sequence can be explained by the model
$\boldsymbol{H}_{\mathrm{rand}}$	The hypothesis that the action sequence can be explained by random choices
$\boldsymbol{\mathcal{L}}$	The log model evidence
$\boldsymbol{p}(\ast)$	The probability of $\ast$
$\boldsymbol{p_t}$	The probability that the model would provide the same response as the participant at time t

$\boldsymbol{R}^2$	The pseudo-R^2
$\boldsymbol{\Theta}$	The number of parameters in a model, $\Theta = \lVert\theta\rVert$

Model values and functions

$\boldsymbol{S}$	The stimulus cues pertinent to the experiment
$\boldsymbol{S_t}$	The stimulus cues available during the trialstep t
$\boldsymbol{s}$	A stimulus cue
$\boldsymbol{s_t}$	The value of the stimulus cue s at time t
$\boldsymbol{D}$	The sequence of actions over time range T
$\boldsymbol{d}$	An available action
$\boldsymbol{V}$	The expected rewards for each action
$\boldsymbol{E}$	The expectation values for each action-stimulus cue
$\boldsymbol{r}$	The reward value
$\boldsymbol{R}$	The maximum reward
$\boldsymbol{\mathcal{R}}$	The possible rewards
$\boldsymbol{I}$	Reward impact on all possible actions
$\boldsymbol{\delta}$	The reward prediction error
$\boldsymbol{A}$	The actor values for each action-stimulus cue
$\boldsymbol{A}^*$	The actor values for each action
$\boldsymbol{P}$	Probability of each action
$\boldsymbol{\mathcal{C}}(\boldsymbol{P})$	The action choice function. The chosen action, c_t is chosen based on the probability of each action at time t, P_t
$\boldsymbol{G}$	The Go value for each action-stimulus cue
$\boldsymbol{N}$	The Nogo value for each action-stimulus cue
$\boldsymbol{B_d}$	Likely highest reward action, $B_d \in \{0,1\}$
$\boldsymbol{\Delta}$	The average reward
$\boldsymbol{\Delta}^*$	The average of the average reward Δ
$\boldsymbol{\sigma}^2$	The prediction uncertainty measure
$\boldsymbol{\hat{\sigma}}^2$	The updated prediction uncertainty measure
$\boldsymbol{\hat{E}}$	The updated expectation values for each action-stimulus cue
$\boldsymbol{\omega}$	The Dirichlet distribution count parameter, $\omega \in \mathbb{R}_{\geq 0}$
$\boldsymbol{\mathfrak{D}}$	The Dirichlet function
$\boldsymbol{\Psi}$	The Digamma function

	Model parameters
$\boldsymbol{\theta}$	The set of all parameters in a model
$\boldsymbol{\alpha}$	The learning rate parameter, $\alpha \in [0,1]$
$\boldsymbol{\alpha_C}$	The critic learning rate parameter, $\alpha_E \in [0,1]$
$\boldsymbol{\alpha_A}$	The actor learning rate parameter, $\alpha_A \in [0,1]$
$\boldsymbol{\alpha^+}$	The positive reward learning rate parameter, $\alpha^+ \in [0,1]$
$\boldsymbol{\alpha^-}$	The negative reward learning rate parameter, $\alpha^- \in [0,1]$
$\boldsymbol{\alpha_G}$	The Go learning rate parameter, $\alpha_G \in [0,1]$
$\boldsymbol{\alpha_N}$	The No-go learning rate parameter, $\alpha_N \in [0,1]$
$\boldsymbol{\beta}$	The exploration-exploitation parameter, otherwise known as the inverse temperature parameter, $\beta \in \mathbb{R}_{\geq 0}$
$\boldsymbol{\varepsilon}$	The likelihood of a non-optimal choice being made, $\varepsilon \in [0,1]$
$\boldsymbol{E_\lambda}$	The baseline expected reward, $E_\lambda \in \mathbb{R}_{\geq 0}$
$\boldsymbol{\gamma}$	The time discounting parameter, $\gamma \in [0,1]$
$\boldsymbol{\kappa}$	The autocorrelation parameter, $\kappa \in [-1,1]$
$\boldsymbol{M}$	The maximal absolute value allowed for Go or Nogo, $M \in \mathbb{R}_{\geq 0}$
$\boldsymbol{o}$	The biased random probability for a given action choice, $o \in [0,1]$
$\boldsymbol{\rho}$	The Go-Nogo actor asymmetry, $\rho \in [0,1]$
$\boldsymbol{\sigma_\alpha^2}$	Learning rate uncertainty measurement, $\sigma_\alpha^2 \in \mathbb{R}_{\geq 0}$
$\boldsymbol{\sigma_\lambda^2}$	The baseline drift rate uncertainty, $\sigma_\lambda^2 \in \mathbb{R}_{\geq 0}$
$\boldsymbol{\tau}$	The learning rate parameter, $\tau \in [0,1]$
$\boldsymbol{\lambda}$	The drift rate, $\lambda \in \mathbb{R}_{\geq 0}$

Appendix II. Model parameter correlations

i. Decks task dataset

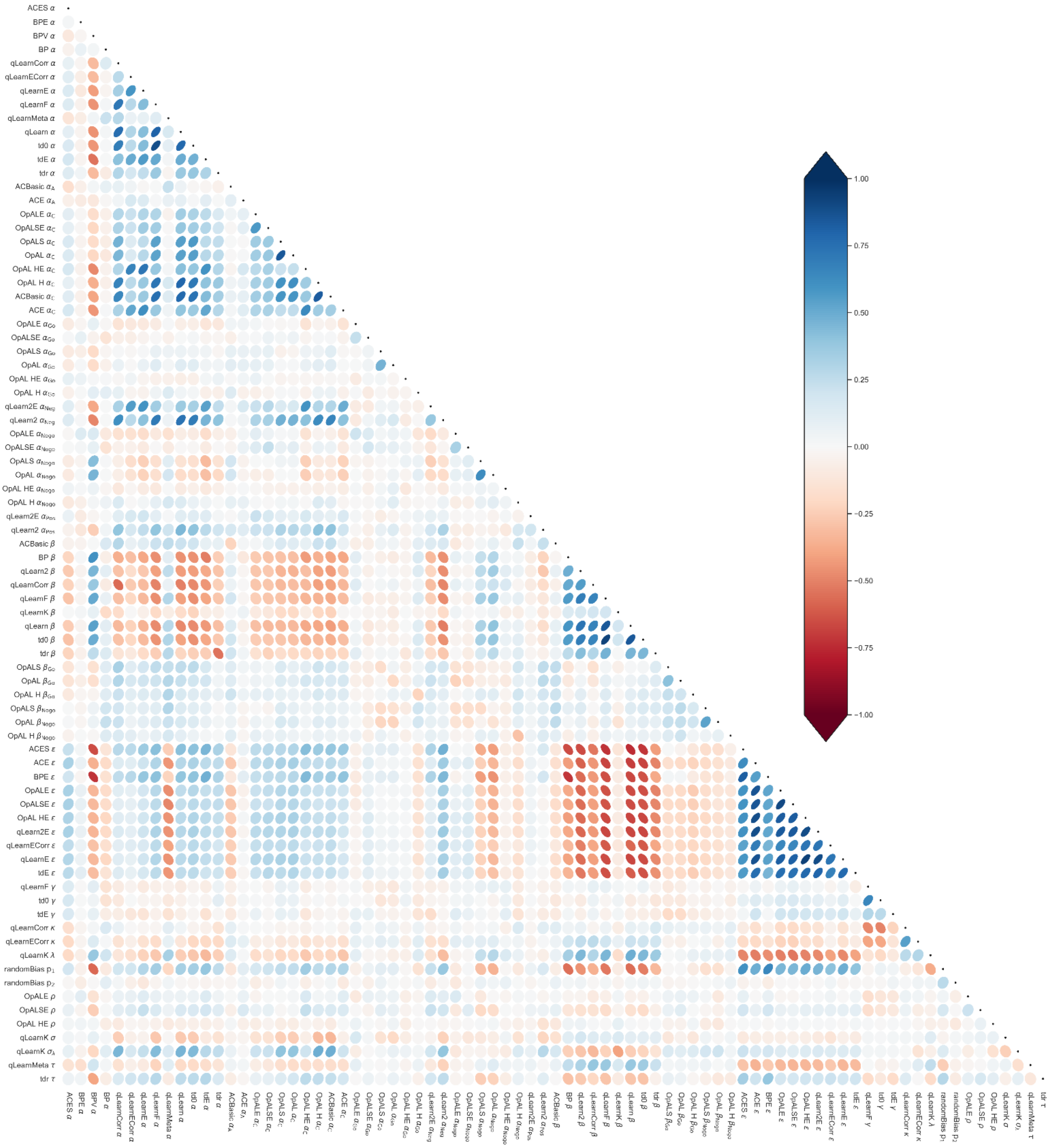


Figure II - 1 The correlations between recovered parameter values from the Decks task participants. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation.

ii. Probabilistic Selection task dataset

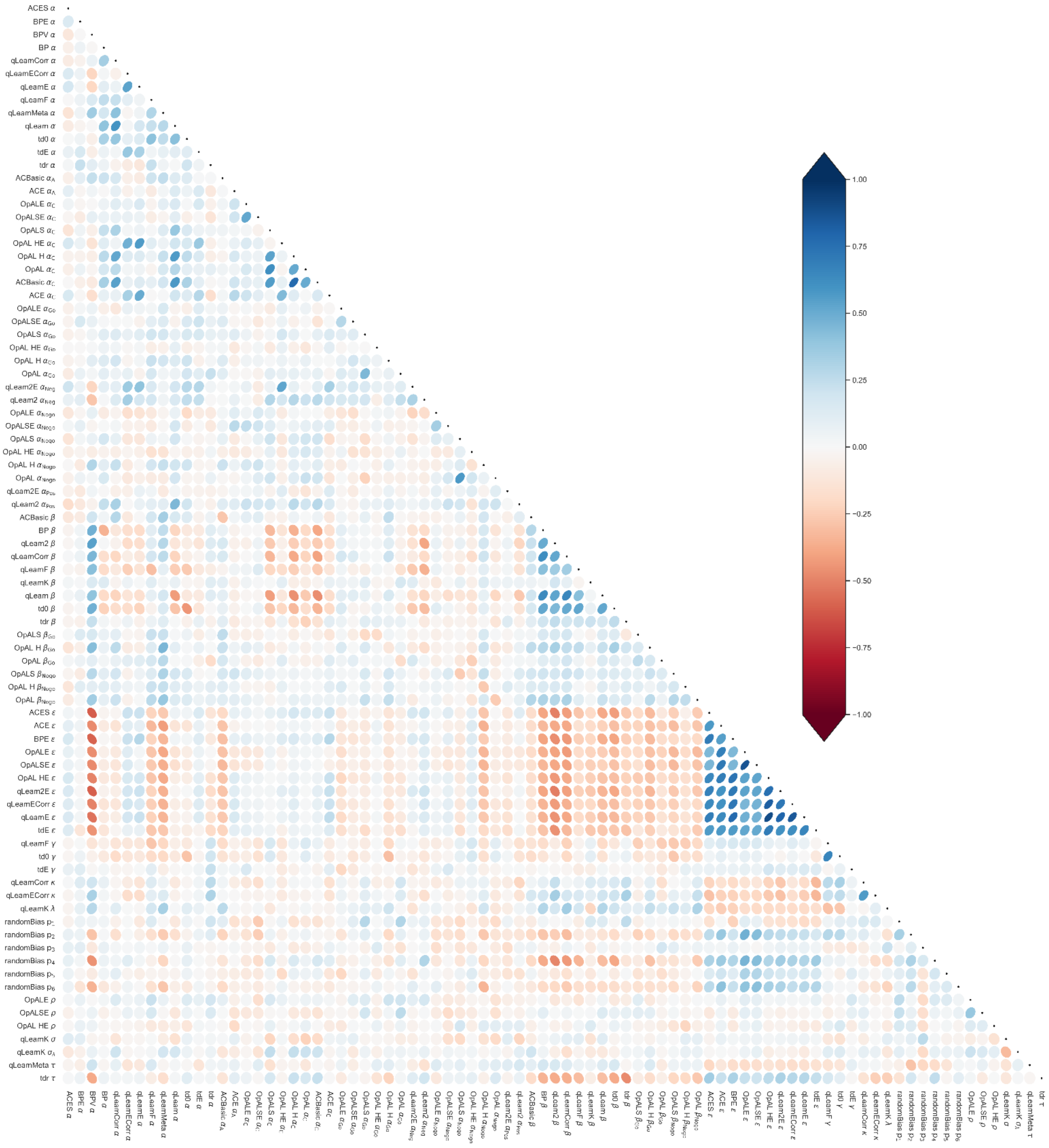


Figure II - 2 The correlations between recovered parameter values from the Probabilistic Selection task participants. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation.

iii. Weather task

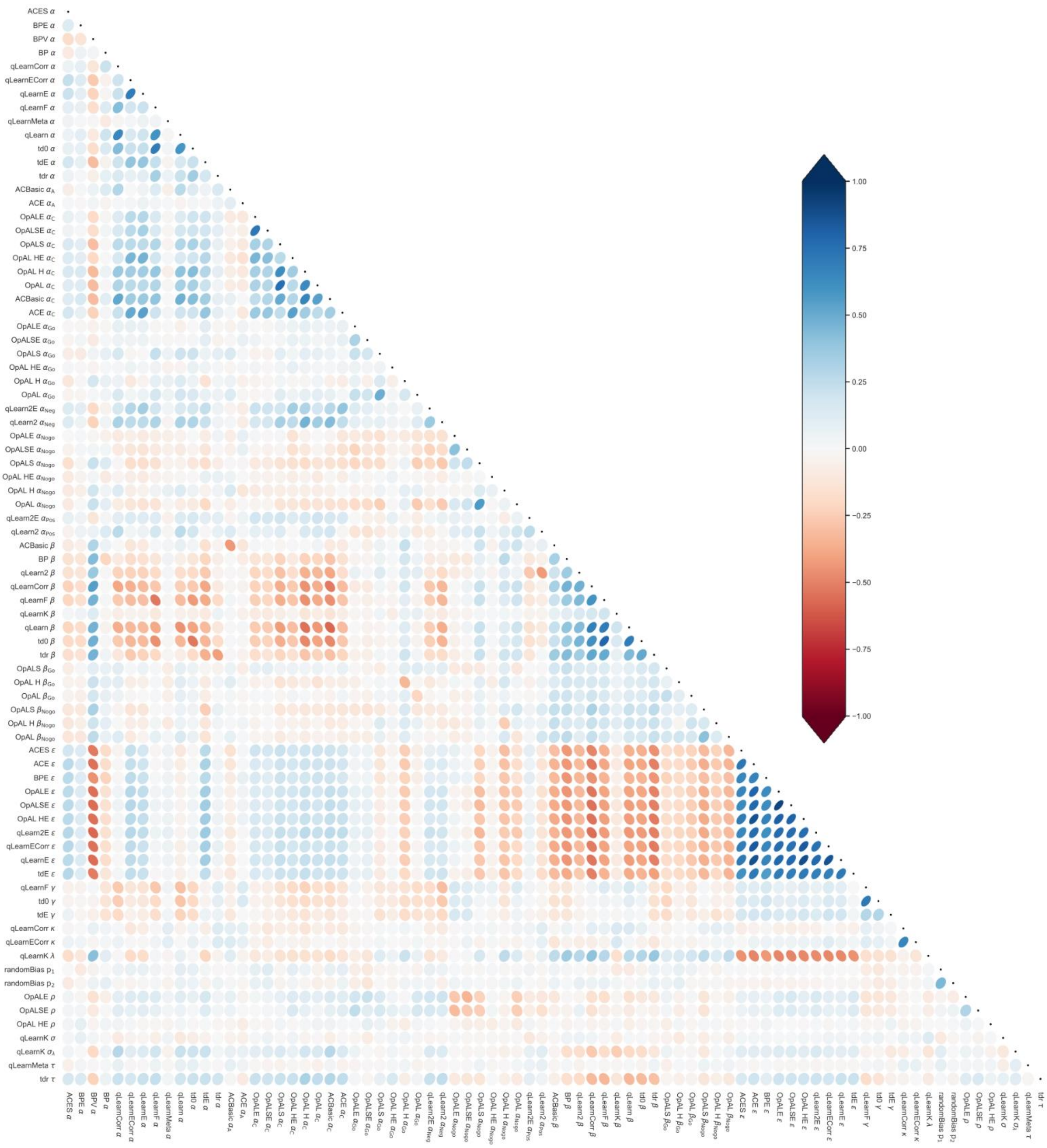


Figure II - 3 The correlations between recovered parameter values from the Weather task participants. A dark blue oval pointing to the top right signifies a strong positive correlation, a white circle no correlation and a dark red oval pointing to the top left signifies a strong negative correlation.