Deep learning as an alternative to global optimization in diffusion model for conflict tasks

Author:

Solveiga Stonkute

A thesis presented for the degree of:

Doctor of Philosophy

School of Psychology

Cardiff University

United Kingdom

January 2019

Summary

To apply mathematical models of decision making in psychological research, researchers need ways to extract model parameters from behavioural studies. The expansion of the drift diffusion model to conflict tasks (DMC) (Ulrich, Schröter, Leuthold, & Birngruber, 2015) resulted in the model being non-differentiable, which means that the parameters of DMC can only be estimated.

The current methods for recovering parameters from DMC rely on comparing reaction time (RT) distributions. Such methods will struggle to recover all DMC parameters well due to the of the solution space of DMC, which means that some parameters can be confused with others when RT distributions are compared.

Following that, five global optimization algorithms from different optimization families were compared to create a benchmark for parameter recovery from DMC. The results revealed that differential evolution outperformed the other four optimization algorithms in recovery of parameters from both distributions with high and low trial numbers.

Even though differential evolution is capable of recovering parameters well, it is very expensive in computational time, which means that researchers who do not have access to vast computational resources cannot apply DMC in their research. Due to this, deep learning was investigated in application of parameter recovery from DMC. The results showed that deep learning recovered all model parameters exceptionally well from RT distributions with large trial numbers, and as well as differential evolution from RT distributions with low trial numbers, which allows application of deep learning models in deployment pipelines that take seconds rather than months.

Finally, deep learning models were applied in several experimental studies investigating the effects of speed-accuracy trade-off (SAT) in response inhibition and perceptual decision making tasks, and how the performance relates between the tasks and over two different testing sessions, and demonstrated the effects of SAT on DMC parameters in different tasks.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Impulsivity and response inhibition

Impulsivity, a trait that leads to individuals acting without foresight, is a personality characteristic seen in healthy individuals. However excessive impulsivity has been associated with many psychiatric disorders, such as attention deficit/hyperactivity disorder (Winstanley, Eagle, & Robbins, 2006; Oades et al., 2008; Newcorn et al., 2001), obsessive compulsive disorder (Grassi et al., 2015), bipolar disorder (Fears et al., 2015; Newman & Meyer, 2014), autism (Carlisi et al., 2017; Richards, Moss, Nelson, & Oliver, 2016), pathological gambling (Steel & Blaszczynski, 1998; Alessi & Petry, 2003; Petry, 2001), internet gaming disorder (Kim et al., 2017), compulsive buying disorder (Black, Shaw, McCormick, Bayless, & Allen, 2012), and Parkinson's disease (Weintraub, David, Evans, Grant, & Stacy, 2015). Impulsivity is also associated with suicidal behaviour (Baca-Garcia et al., 2001; Gvion & Apter, 2017) and aggressiveness (Bousardt, Hoogendoorn, Noorthoorn, Hummelen, & Nijman, 2016).

Even though impulsivity can be adaptive and advantageous in circumstances where it is important to respond rapidly and to take advantage of unexpected opportunities, most often it is associated with negative aspects of human behaviour, as even some definitions of impulsivity focus on the negative maladaptive aspects, such as "actions which are poorly conceived, prematurely expressed, unduly risky or inappropriate to the situation and that often result in undesirable consequences" (Durana, Barnes, Johnson, & Shure, 1993). Dalley, Everitt, and Robbins (2011) deconstructed this definition to suggest that impulsivity, like many behavioural constructs, is multifaceted, as the definition describes behaviour that has not adequately sampled sensory evidence ("reflection impulsivity"), a failure of motor inhibition ("impulsive action"), a tendency to accept small immediate or likely rewards versus large delayed or unlikely ones ("impulsive choice") and risky behaviour, in the context of decision-making. Constructs of impulsivity depend on measures (Robbins, Gillan, Smith, de Wit, & Ersche, 2012), and conversely, different measures of impulsivity have been developed to assess certain impulsivity constructs.

The three most general approaches to measuring impulsivity in humans are self-report measures of impulsivity, impulsive choice, and impulsive action. Self-report measures of impulsivity are usually assessed with questionnaires, for example, the Barratt Impulsivity

Scale (BIS-11) (Stanford et al., 2009), the UPPS-P Impulsive Behaviour Scale (UPPS) (Lynam, Smith, Whiteside, & Cyders, 2006), the Impulsivity Rating Scale (Lecrubier, Braconnier, Said, & Payan, 1995), or the Eysenck Personality Questionnaire (EPQ) (Eysenck & Eysenck, 1985). Self-report measures of impulsivity are easy to administer and display highly stable trait-like characteristics (Weafer, Baggott, & de Wit, 2013), however the questionnaires tend to be more reflective of the subjective view that an individual has of his own behaviour (Robbins et al., 2012).

Impulsive choice describes a situation when an individual chooses an immediately available small reward over a larger one in future (Bickel, Odum, & Madden, 1999; Weafer et al., 2013). Impulsive choice in humans is often measured with delay tasks or probability discounting tasks, in which participants choose between small, immediate rewards and large, delayed or probabilistic rewards. In order to evaluate impulsive choice in laboratory settings, risk-taking tasks, such as the Balloon Analogue Risk Task (BART), have also been employed. However, interpreting delayed discounting and its relation to impulsivity needs to be done with caution, as Anokhin, Golosheykin, Grant, and Heath (2017) reported that lower socioeconomic status of the families of participants predicted the tendency of adolescents to place higher value on immediate reward. However, impulsive choice tasks seem to have good test-retest stability (White, Lejuez, & de Wit, 2008; Beck & Triplett, 2009; Smits, Stein, Johnson, Odum, & Madden, 2013), indicating that it might be a stable construct of impulsivity. Moreover, impulsive choice has been reported to correlate with other aspects of impulsivity, such as BIS (de Wit, Flory, Acheson, McCloskey, & Manuck, 2007).

Impulsive action (also known as behavioural inhibition) refers to the ability to inhibit inappropriate or unwanted behaviours (Weafer et al., 2013). Impulsive action is easy to evaluate in experimental settings, however it is typically thought thought to assess fluctuating states of impulsivity (Dick et al., 2010). Impulsive action is often assessed using response inhibition tasks. Response inhibition is defined as the ability to cancel a prepotent response or to suppress an action which is inappropriate, irrelevant, or no longer required (Friedman & Miyake, 2004). Motor response inhibition includes go/no-go (Iaboni, Douglas, & Baker, 1995) and stop-signal tasks (Logan, 1994), while interference inhibition is tested by tasks like Simon (Simon & Rudell, 1967), Eriksen flanker task (Eriksen & Eriksen, 1974) and Stroop task (Stroop, 1935). All these tasks measure the ability to resolve response conflict due to interfering stimulus features that are irrelevant, but might lead to incorrect responses if not suppressed correctly (Wöstmann et al., 2013).

Using response inhibition tasks instead of self-report measures is advantageous because response inhibition tasks are less susceptible to social desirability biases. They are also advantageous because they are sensitive to experimental state manipulations. Deficits in response inhibition tasks have been consistently found among clinical populations defined by impaired impulsivity, like ADHD (Wodka et al., 2007; Aron & Poldrack, 2005; Booth et al., 2005), bulimia (Marsh et al., 2009; Wu, Hartmann, Skunde, Herzog, & Friederich, 2013), bipolar personality disorder (Hajek, Alda, Hajek, & Ivanoff, 2013), and autism (Agam, Joseph, Barton, & Manoach, 2010; Kana, Keller, Minshew, & Just, 2007). However, relationship between performance in response inhibition tasks and trait impulsivity (as measured by self-report questionnaires) has not been so straightforward to establish, as some researchers report significant correlations between response inhibition tasks and impulsivity questionnaires (Aichert et al., 2012; Enticott, Ogloff, & Bradshaw, 2006; Reynolds, Ortengren, Richards, & de Wit, 2006); while others find no relationship (Horn, Dolan, Elliott, Deakin, & Woodruff, 2003; Cheung, Mitsis, & Halperin, 2004; Cyders & Coskunpinar, 2011; Sharma, Markon, & Clark, 2014).

The lack of relationship between self-reported measures of impulsivity and inhibitory control translates even to research on impulsivity as endophenotype for highly heretable disorders. Impulsivity can be an endophenotype of a disorder - meaning that those who do not have disorder, but are at increased genetic risk (like twins, siblings, first-degree relatives), also have affected impulsivity (Fortgang, Hultman, van Erp, & Cannon, 2016). The researchers, using twin probands (meaning pairs of twins where at least one individual is affected by the disorder, or a proband) with bipolar disorder, schizophrenia, and major depressive disorder, as well as healthy controls, found that even though bipolar disorder is usually thought of having impulsivity as endophenotype, the same is true for schizophrenia and major depressive disorder, probably due to shared genetic overlap between the disorders. This finding is interesting as both schizophrenia and major depressive disorder are not usually considered impulse control disorders, or associated with extreme and risky impulsive actions. Importantly, the researchers found that only self-report measures of impulsivity were moderately heritable and patterned as endophenotypes for the three disorders, however impulsive action, as measured by stop signal task, was not affected in twins of participants with schizophrenia and bipolar disorder, even though the affected twins showed impaired performance in the task. The authors concluded that impulsive action seemed genetically distinct from attentional, motor, and non-planning types of impulsivity, as measured by Barratt Impulsiveness Scale-11 (BIS-11) (Patton, Stanford, & Barratt, 1995) (which is one of the most commonly used self-report questionnaires designed to assess the personality and behavioral construct of impulsiveness), even though the former seemed to be more sensitive to clinical state than the self-report measures.

Conversely to this view, Anokhin et al. (2017) investigated the heretability of neural correlates of response inhibition using go/no-go task in adolescents with ADHD and their unaffected twins, and found that the neural correlates were highly influenced by genetic factors, throughout adolescence, with little change in heritability over age. However this study only reported results for neural correlates of the go/no-go task, but not the behavioural results, therefore it is unclear whether behavioural measures could also be an endophenotype for ADHD.

The two studies by Anokhin et al. (2017) and Fortgang et al. (2016) investigated hereditability by examining performance of individuals suffering from an impulsive disorder and their unaffected twins, and then modelling different genetic components of that performance. Cummins et al. (2012), on the other hand, looked at the genetic variants in healthy individuals and investigated how variants associated with genes that are involved in synthesis, degradation, transport and receptor signalling of dopamine and noradrenaline relate to response inhibition in healthy individuals. The response inhibition phenotype was measured by the stop signal task. The authors found significant associations between dopamine transporter genotype and stop signal reaction time that survived corrections for multiple comparisons. These associations were independent of other associations with behavioural measures such as response speed and reaction time variability. The results suggest that response inhibition could be a stable measure of impulsivity if it is affected by DNA variations.

A big problem in response inhibition literature is that seemingly similar response inhibition tasks with presumably similar underlying constructs do not correlate when behaviour is compared within the same individuals using different tasks. Even studies with relatively large samples assessing relationships between performance in different response inhibition tasks has not provided good results. For example, Aichert et al. (2012) has investigated the relationship between trait impulsivity as assessed by the BIS-11 questionnaire and four different response inhibition tasks (Stroop, antisaccade, go/no-go, and stop signal task) while also controlling for gender and intelligence, in a large (N=504) healthy sample. They found only very modest correlations between antisaccade error rates and go/no-go commission error rates; and antisaccade error rates and Stroop interference score. The performance in the stop-signal task was not related to performance in any other response inhibition task. Presumably this resulted from stop-signal task requiring a cancellation of a response that is already being initiated, unlike other response inhibition tasks where an automatically activated interfering information needs to be suppressed before response is being activated. Kertzman, Vainder, Aizer, Kotler, and Dannon (2017) found that multiple different response inhibition measures (go/no-go, Stroop, Matching Familiar Figures) showed different means between pathological gamblers and healthy controls, yet the measures themselves were not correlated in the same groups.

The lack of correlations between tasks are difficult to interpret, according to Friedman and Miyake (2004), because there are multiple reasons for such results. It could be that inhibition is a multifaceted construct and different response inhibition tasks tap into different aspects of inhibition. There could also be a lack of construct validity - Friedman and Miyake (2004) argue that the negative priming effect is frequently used as an inhibition measure, but that it is not universally agreed that the negative priming effect is due to inhibition. Another reason could also come from the difference scores as Hedge, Powell, and Sumner (2018) discussed, the difference between two measures is less reliable than the individual measures themselves when the measures are highly correlated and have similar variance, which is usually the case with measures in response inhibition tasks. Finally, no task can solely measure just inhibition, so only a small percentage of variance of the task may be attributable to inhibitory processes (Hedge, Powell, Bompas, Vivian-Griffiths, & Sumner, 2018).

1.2 Speed-Accuracy trade-off in response inhibition tasks

Another reason for low reliability of response inhibition tasks could be that researchers rarely take into account how task performance is affected by speed-accuracy trade-offs (SAT). SAT describes a phenomenon where under speed pressure, responses are made faster but are more prone to errors, compared to decisions which are more accurate but take much longer time to make. SAT has been studied extensively in perceptual decision making (J. Zhang & Rowe, 2014; B. U. Forstmann et al., 2010; Rae, Heathcote, Donkin, Averell, & Brown, 2014; Pote et al., 2016; Winkel et al., 2012), however the effects of SAT have not been studied much in inhibitory control. Even a study by

Mulder et al. (2010), investigating the impairments in regulating the speed-accuracy trade-off in individuals with ADHD, used a perceptual decision making task rather than a response inhibition task. The main reasons for that is because it is difficult to see what strategies individuals choose when performing tasks, as SAT is almost never explicitly evaluated.

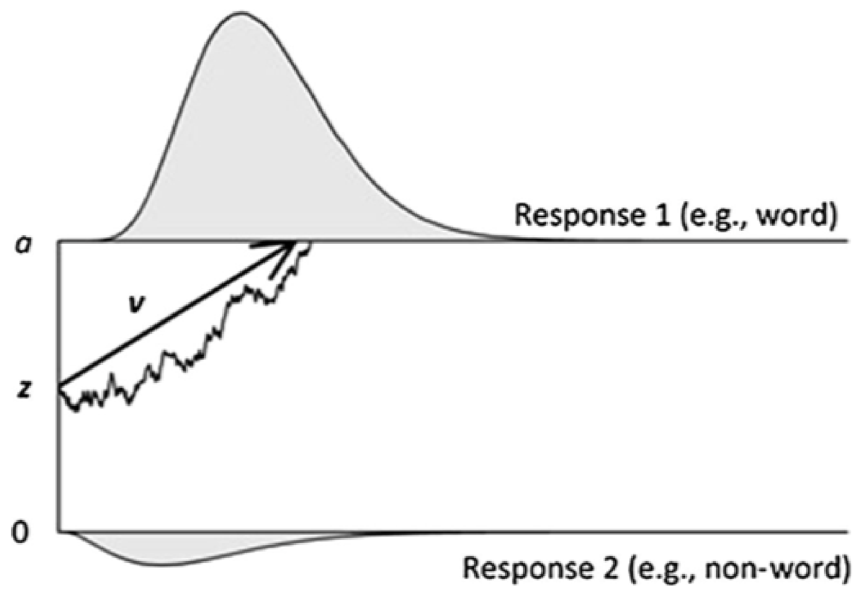
Only a few studies so far have looked at these behaviour adjustments in response inhibition. One of them, by Wylie et al. (2009), looked at the effect of speed-accuracy strategy on Eriksen flanker task performance in individuals with Parkinson's disease and healthy controls. The participants in this study were instructed to either respond as accurately as possible or as fast as possible to the stimuli, and found that both response times and accuracy of responding differed in both individuals with Parkinson's disease and healthy controls depending on the instruction set. Another study by Leotti and Wager (2010) investigated response strategies in the stop signal paradigm and found that individuals varied in strop signal performance as some complied with regular stop signal paradigm instructions that emphasize speed, while others do not comply with instructions and prefer slower but more accurate style of responding. When the researchers manipulated speed and accuracy by varying rewards for either fast or accurate responses, individuals' behaviour adapted to these situations. A study by Van Wouwe et al. (2014) looked into the effects of Parkinson's disease on the ability to resolve conflicts in Simon task under speed and accuracy pressures. The researchers found that both healthy controls and individuals with Parkinson's disease showed typical performance under SAT conditions: when instructions emphasized response speed, participants responded faster and made more errors than when response accuracy was emphasized. They also found that patients with Parkinson's disease struggled to suppress incorrect response impulses during speed but not accuracy conditions, suggesting that specific patterns of impulsivity could be revealed by manipulating SAT during response inhibition tasks. Even though these studies provide evidence that individuals' inhibitory behaviour does vary with task demands, there are no studies yet investigating whether this ability is stable in time and between different response inhibition tasks.

Another issue in evaluating impulsivity in response inhibition tasks is that reaction times and error rates are not ideal measures of impulsivity, as they say little about underlying cognitive processes in decision making and are not pure measures of of any one single cognitive process. Response inhibition tasks record performance in terms of both reactions times and error rates, and increases in either tend to reflect increased task difficulty. These two measures are used interchangeably in research, even though there is evidence that reaction times and error costs from the same tasks do not correlate well (Hedge, Powell, Bompas, et al., 2018).

1.3 Diffusion model for conflict tasks

All of the above issues could be addressed with the cognitive modelling framework. The most popular class of cognitive models assumes that when decisions are made, noisy samples of information are accumulated until a threshold of evidence is reached. Such accumulation-to-threshold models are known as sequential sampling models. They provide a theoretical framework for dissociating underlying cognitive mechanisms from decision making tasks while also accounting for speed-accuracy trade-offs (B. Forstmann, Ratcliff, & Wagenmakers, 2016). Cognitive modeling is advantageous because it goes beyond description of data and seeks to provide an explanation of behaviour while being designed to be much simpler and abstract versions of human cognition, as they keep the essential features, but discard unnecessary details (Lewandowsky & Farrell, 2010).

The most famous of sequential sampling models is the drift diffusion model by Ratcliff (1978). In the drift diffusion model, evidence about a stimulus accumulates from a starting point to a boundary. There are two boundaries, one boundary for each choice, in a two-choice task, which represent the amount of evidence that must be accumulated before a response is made (Ratcliff, Smith, Brown, & McKoon, 2016). The drift diffusion model has four key parameters: the quality of evidence (drift rate), the amount of evidence needed for a decision (boundary separation), the duration of non-decision processes (non-decision time), and bias towards one choice over other (starting point). The illustration of the drift diffusion model is presented in figure 1.1 (from Lerche and Voss (2017)), which demonstrates the evidence acuumulation process, where evidence starts to accumulate at the starting point, and moves towards one of the response thresholds with drift rate. Once the evidence reaches one of the response thresholds, the decision is made. The drift diffusion model accounts for three dependent variables simultaneously - accuracy and the shape of the correct and incorrect reaction time distributions, and therefore helps to avoid inconsistencies from choosing to analyze only one of the dependent variables (B. Forstmann et al., 2016). Application of the drift diffusion model to experimental data has demonstrated boundary settings and drift rates are largely unrelated, and that accuracy mainly correlates with drift rate while reaction time correlates with boundary separation, which could explain the lack of correlation between reaction time and accuracy (Thompson, Ratcliff, & McKoon, 2016; Hedge, Powell, & Sumner, 2018). Moreover, boundary separation parameter in drift diffusion model supposedly represents the level of caution; as boundary separation increases, fewer errors are made, but the responses also become slower. In this way, SAT is implemented in the drift diffusion model via the boundary separation parameter (B. Forstmann et al., 2016). Boundary separation and caution is interesting for impulsivity and response inhibition research, as, in theory, boundary separation is the only parameter that should be under direct influence of the participant. Pote et al. (2016) manipulated the level of caution and response threshold by subthalamic nucleus deep-brain stimulation in patients with Parkinson's disease, and induced impulsive action in patients when they were acting under speed pressure. This provides evidence that speed-accuracy trade-off and boundary separation is important in understanding impulsivity and caution.



.

Figure 1.1: Visual representation of the evidence accumulation in the drift diffusion model. The figure illustrates how evidence starts to accumulate at the starting point z, then moves with drift v towards the upper threshold a. When the evidence reaches the threshold, the decision is made. Figure from Lerche and Voss (2017).

The drift diffusion model has been applied successfully to a wide range of decision making tasks, but it cannot quite capture the patterns of data seen in response inhibition tasks, such as fast errors in incongruent trials. There have been attempts to modify the drift diffusion model to account for such data in the Eriksen flanker task (White, Ratcliff, & Starns, 2011), but this adjustment cannot address other issues, like negative going delta plots (which indicate that experimental effects of congruency are more pronounced in short responses than they are in longer responses) in Simon task (Ulrich et al., 2015). Moreover, it cannot fit other response inhibition tasks. For this reason, diffusion model for conflict tasks by Ulrich et al. (2015) is a good choice to use with response inhibition tasks, as it still retains basic aspects of drift diffusion model but is general enough to account for different response inhibition tasks. This model maps quite well to drift diffusion model by including non-decision time, boundary separation, and drift rate for controlled process, between-trial variability in starting point distribution and variability of non-decision time, but also contains three parameters that describe an impulse function: amplitude of automatic activation, time to peak automatic activation, and shape of automatic activation.

A simple visualization of the diffusion model for conflict tasks is shown in figure 1.2 for congruent and incongruent trials. The model postulates that the evidence accumulation process consists of two processes, one controlled (which would be identical to the accumulation process in the drift diffusion model), and another one that is automatic (which makes the model suitable for inhibitory tasks). The controlled process mimics the evidence accumulation process of the drift diffusion process, with main parameters of drift rate (how fast the evidence accumulates), and the threshold (or boundary separation) - the amount of evidence to activate a decision. The automatic process, pulls the evidence accumulation towards a specific response depending on the congruency of the stimulus: for congruent stimuli, the automatic activation provides a boost towards the correct response, while for the incongruent stimuli, the automatic activation pushes the evidence towards the incorrect response. The automatic activation is influenced by the impulsivity parameters of the models, which define how much of a boost the automatic process provides (amplitude of automatic activation), when does the automatic process peak in amplitude (time to peak automatic activation), and the shape of the automatic process (shape of automatic activation). The non-decision time is a parameter shared between the two models, and corresponds to non-decision parts of the process, such as perceiving the stimulus and making a physical response. Variability in the two models is also encoded with same parameters: variability of non-decision time, and variability of starting point between trials, implemented in the diffusion model for conflict tasks as a general beta distribution that is symmetrically centered around zero (starting point). The shape of this beta distribution is the shape of the starting point parameter (for more details of mathematical implementation, refer to Appendix D in Ulrich et al. (2015)).

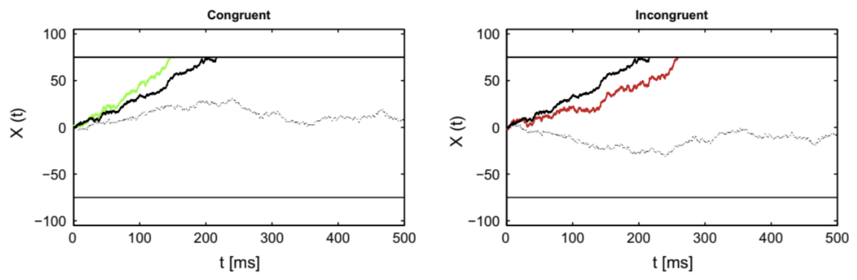


Figure 1.2: Visual representation of the evidence accumulation in the diffusion model for conflict tasks. The figure illustrates how evidence starts to accumulate at the starting point X(0), then moves with drift towards the upper threshold a. The controlled process is displayed in black, while the automatic process is shown in grey. The automatic process is positive in congruent trials (moving towards correct response threshold), while in incongruent trials, it is negative (moving towards the incorrect response threshold). The two processes are superimposed to produce a combined evidence accumulation, shown in green for congruent trials, and in red for incongruent trials. The decision is made when the colored trace reaches the decision boundary. Figure from Ulrich et al. (2015).

1.4 Parameter recovery from non-differentiable models

Sequential sampling models are useful because they provide insights into human behaviour that cannot be obtained from behavioural measures alone. Application of the drift diffusion model has helped to understand how cognitive processing is affected by aging (Ratcliff, Thapar, & McKoon, 2010), or how emotional processing affects decision making in individuals with depressive symptoms (White, Ratcliff, Vasey, & McKoon, 2009). But application of cognitive models to experimental studies is not straightforward, because parameter recovery from models can be complicated. In psychological research, parameter recovery refers to finding the parameters of a model that provides the best fit to the experimental data, because the true parameter values are unknown, however, in ideal cases, true parameters and best fitting parameters should be the same.

In most cases, the fewer parameters the model has, the easier it is to recover those parameters. Some models, like EZ diffusion model (Wagenmakers, Van Der Maas, & Grasman, 2007), are very easy to apply to experimental data because only minor calculations are needed to recover parameters from reaction time distributions. Standard drift diffusion model is a differentiable model, which means that a proven global minimum solution can be found, even if the application of the differential equations are quite difficult. On top of that, there are plenty of ready-to-use packages that allow parameter recovery from drift diffusion model without much difficulty.

Parameter recovery from extension models, like the diffusion model for conflict tasks, becomes quite difficult, because the addition of automatic activation parameters makes the model non-differentiable, meaning that the solution to the model (a set of parameters that fits a reaction time distribution) cannot be calculated using mathematical equations. A lack of analytical solution means that in order to recover parameters from the model, optimization algorithms need to be used. An optimization algorithm, in general terms, is a process that compares solutions to a problem over multiple iterations, until a satisfactory solution is found. Global optimization tries to locate the global minimum (or maximum) of a function over a given set, where multiple local minima might be present. Stochastic global optimization generates and uses random variables to solve optimization problems. Stochastic methods that introduce randomness into the search process can accelerate the optimization process and help the algorithm escape local minima. Even with stochasticity, the application of global optimization algorithms is computationally demanding, and it does not guarantee a global minimum solution. More details on different global optimization algorithms used in this thesis are provided in section 3. There are currently no packages that offer quick parameter recovery from diffusion model for conflict tasks. Even though some packages, like Hierarchical Bayesian (Wiecki, Sofer, & Frank, 2013), which recover parameters from the drift diffusion and the linear ballistic accumulator models, could also be extended to diffusion model for conflict tasks as the package is implemented in PYTHON, it would still not be suitable to researchers who are interested in individual differences, as a Bayesian hierarchical approach assumes a general overall underling distribution of the sample.

There have been studies that looked into parameter recovery from non-differentiable decision making models like the diffusion model for conflict tasks (White, Servant, & Logan, 2018) or leaky competing accumulator (Miletić, Turner, Forstmann, & van Maanen, 2017). Both studies found that parameter recovery worked well when parameters were constrained, but both studies also heavily constrained the search space. In real-word situations, where experimental data comes from unknown parameter ranges, it is difficult to predict how these methods would perform. This problem is exaggerated in cases where human behaviour is pushed to the edges, like in SAT experimental designs - when the behaviour becomes more extreme, the search space should in theory also increase, which in turn could hinder parameter recovery.

There are many global optimization algorithms, broadly divided into four families by their underlying mathematical, philosophical, and inspirational principles: evolutionary algorithms, swarm intelligence algorithms, Bayesian optimization algorithms, Markov Chain Monte Carlo based algorithms. Each of these families has a global optimization algorithm that is the most suitable for numeric optimization problems, which is the problem with parameter recovery from decision making models. There has not been much work done to evaluate how different global optimization algorithms perform in parameter recovery, especially in non-differentiable models. Hawkins, Forstmann, Wagenmakers, Ratcliff, and Brown (2015) suggested that differential evolution seemed to perform better than particle swarm optimization and simplex algorithms in their study that compared the drift diffusion model and some extensions, however the researchers did not provide any results or benchmarking. Moreover, the algorithms from different families are never evaluated against the most simple black box optimization algorithm, which is random search, so it is unclear whether more complex mathematical implementations add anything to parameter recovery.

The issue with global optimization algorithms is that irrespective of the algorithm, the comparison between reaction time distributions needs to be made in order to assess the goodness of fit between sets of parameters. We do not know whether there are multiple sets of different parameters that produce the same looking reaction time distributions. Moreover, the process of making distributions is stochastic, in a way that there is a randomness element in producing a single reaction time trial. Due to this, two reaction time distributions computed with the same parameters could not look identical, so the statistic to evaluate them will suffer. As the number of trials per distribution becomes smaller, this problem increases. Finally, as already mentioned, applying global optimization algorithms is very computationally expensive. This is the case because to evaluate the goodness of fit of a single set of model parameters, a full reaction time distribution from the model needs to be computed. Even with CYTHON (Behnel et al., 2011) optimizations, which compile PYTHON code to run in C, using global optimization algorithms for parameter recovery can take tens of hours. If researchers do not have access to significant computational resources, then parameter recovery from a multitude of participants becomes too time costly, as was emphasized by Ambrosi, Servant, Blaye, and Burle (2019), who could not recover the parameters from diffusion model for conflict tasks for each individual child in their study (from a sample of 53 children), because they did not have enough computational resources.

1.5 Deep learning

Traditional machine learning algorithms have been used for decades to predict outputs from sample data by building mathematical models. Machine learning has been used for various tasks, both classification and regression, and also unsupervised clustering, with specific algorithms tailored to each task and also to input sample characteristics. Even though machine learning methods are very powerful in specific task performance, they suffer from two major drawbacks: first, task performance in heavily influenced by characteristics of input samples, which means that researchers need to be very skilled at feature engineering; and second, as the number of input samples increases, the computational load of the algorithm also increases. Both of these drawbacks can be overcome by deep learning methods, which are capable of performing supervised, unsupervised, classification, regression, clustering, and generative problems.

Even though traditional shallow machine learning methods have been applied in research for a few decades now, deeper methods have only gained interest in the past decade. Deep learning (or deep neural networks) is becoming a very popular machine learning algorithm for learning from and making predictions on data in a variety of fields, such as computer vision (Gebru et al., 2017; Cruz, Luvisi, De Bellis, & Ampatzidis, 2017), natural language processing (Li, 2017), bioinformatics (Uziela, Menéndez Hurtado, Shu, Wallner, & Elofsson, 2017), among many others, due to beating records in many artificial intelligence problems (LeCun, Bengio, & Hinton, 2015). Deep learning methods can automatically identify the optimal representation from the raw data without requiring prior feature selection, which makes application of deep learning in different research fields very straightforward.

Deep neural networks, as the name suggests, are based on artificial neural networks. Artificial neural networks are loosely based on live neural architectures. Natural neurons receive input signal from synapses. The signal from multiple synapses is then combined.

If the combined signal is above a certain threshold, the neuron is activated and emits an output signal. The output signal could be sent to another synapse, and could activate other neurons. Similarly, artificial neural networks consist of an input layer, which is a set of units (like synapses on a live neuron) that represents the inputs into the model, then a hidden layer of units that takes the inputs in specific proportions and combines them, and then applies a non-linear function to produce a signal that gets sent to the output layer. Artificial neural networks are not impressive at solving complex machine learning problems, because the limited architecture with a single hidden layer cannot learn representations well. When artificial neural networks are expanded with multiple hidden layers, where one hidden layer provides the input into another hidden layer, they become deep neural networks. The hierarchical structure of the deep neural networks, which involves the application of consecutive nonlinear transformations to the raw data via the activation function, allows the models to solve quite complicated problems (Vieira, Pinaya, & Mechelli, 2017).

The real power of deep learning comes not from the neural inspired deep architecture, but from its ability to learn. Learning is achieved through an iterative process of adjustment of the connections between the artificial neurons within the network (Bengio et al., 2009). This ability to backpropagate the error of the performance, and to adjust the weights of the network in such a way that it reduces the size of the error, implemented via stochastic gradient descent (Bottou, 2012), allows deep neural networks to learn how to solve very complicated problems. Because the networks are deep (have many hidden layers), they can extract features from input data that are necessary for task performance. The ability of the neural networks to learn feature representations by itself, instead of using input features engineered by the researchers, makes the algorithm very universal and applicable to almost all fields of research. However, the same aspect also makes deep learning extremely difficult to interpret, which makes it challenging to use in fields where decisions made by the algorithm have to be understood by humans.

Deep neural networks are usually quite large, as they consist of many hidden layers, with each hidden layer having tens, hundreds, or even thousands of units. This results in the models having a very large number of trainable parameters, as the weight of each individual unit in each of the hidden layers needs to be learned for the solution of the given problem. For example, the ImageNet classification model by Krizhevsky, Sutskever, and Hinton (2012) contains over 60 million trainable parameters. Due to such a large number of trainable parameters, the deep learning models requite huge amounts of training data, which can be a big problem in fields where abundant amounts of data are hard to come by, such as health care (Miotto, Wang, Wang, Jiang, & Dudley, 2017). However in decision making modelling, lack of data is not an issue as samples can be created very easily at will. This makes deep learning extremely suitable to solving problems in decision making modelling, such as parameter recovery.

Deep learning methods are famed for computational complexity, which is well illustrated by the fact the method only took off in the past 10-12 years, when the cost of computational hardware sharply decreased (LeCun et al., 2015). However there are two stages of application of deep learning: first, the models need to be trained to solve the problem, then they can be applied to unseen data. Only the training stage is computationally expensive, as it depends heavily on the number of trainable parameters and the number of training samples, which usually go hand in hand. Application of the models is, however, computationally cheap, provided that the inputs are not large (e.g., real time video streams in ultra-high definition), and that the application of the models is not pushed to edge devices with very limited computational resources. For this reason, applying deep learning to parameter recovery from the diffusion model for conflict tasks could be beneficial as it would solve the biggest problem with the application of the model to experimental data: the amount of time it takes to recover the parameters using conventional global optimization methods.

Deep learning seems to be the most fashionable machine learning algorithm at the moment, probably due to its very impressive and heavily publicized achievements, such as the success of AlphaGo (Silver et al., 2017), with an increasing number of fields trying to achieve the same level of success. However, it is often stated that classical machine learning algorithms can achieve similar results, without having to use huge computationally excessive architectures. However shallow methods rely heavily on feature engineering, which is specific to each field and takes considerable skill of the researcher to master. Also in order for shallow machine learning methods to perform well, it is necessary to know a priori whether the problem would benefit from linear or non-linear methodology. This knowledge is hard to come by in the field of decision making models, as machine learning, as of the time of the writing, has not been applied to parameter recovery from any decision making model. Deep learning, although computationally excessive, does not care about either feature engineering, or linearity/non-linearity of the problem, therefore it can be applied to parameter recovery with few complications.

1.6 Thesis outline

The work presented in the following chapters will cover two main topics: parameter recovery from diffusion model for conflict tasks, and investigating stability of performance in response inhibition tasks using SAT and modelling approaches. The first half of the thesis will start by investigating the solution space of the diffusion model for conflict tasks, in order to determine what effect do individual parameters of diffusion model for conflict tasks have on reaction time distributions from two-choice response inhibition tasks, and to establish whether each parameter can subsequently be recovered by global optimization algorithms. In order to investigate how well optimization algorithms can recover parameters from diffusion model for conflict tasks, five different optimization algorithms, each from a different stochastic optimization algorithm family, will be compared. Irrespective of how well global optimization algorithms perform in parameter recovery, they are still exceptionally limiting in application time, as recovery of parameters from a single reaction time distribution takes hours if not days. For this reason, the two subsequent chapters will investigate whether deep neural networks can learn to predict parameters from conflict diffusion model using distributions with large number of trials, and then try to adapt deep learning models to work with data from experiments with human participants, which tend to have much lower trial number per reaction time distribution.

The second half of the thesis will apply the deep learning models from the first half in experimental studies of response inhibition aimed to investigate the whether response inhibition is stable within two different tasks, and also over a period of time, when participant behaviour is measured with SAT in mind. As reaction time and error rates are not optimal measures of performance in response inhibition tasks, the diffusion model for conflict tasks (Ulrich et al., 2015) is going to be applied to the results of the response inhibition tasks, using trained deep learning models to recover parameters from experimental data. Are model parameters from diffusion model for conflict tasks stable across different response inhibition tasks and over a period of time? Finally, the same analysis is going to be applied to an experiment in which participants performed a response inhibition task and a perceptual decision making task without impulsivity elements. This will allow to establish whether the relationship between behavioural measures and model parameters between different response inhibition tasks result from decision making, rather than inhibitory processes. The differentiation between general decision making and inhibitory processes can be achieved, because the use of the diffusion model for conflict tasks allows direct comparison with drift diffusion model, as they share same parameters that underlie general decision making, while the diffusion model for conflict tasks contains inhibition-specific parameters.

2 Solution space of diffusion model for conflict tasks

2.1 Introduction

The diffusion model for conflict tasks (Ulrich et al., 2015), an adaptation of drift diffusion model (Ratcliff et al., 2016) that allows better fits to response inhibition tasks, has a major disadvantage in using non-differentiable equations to compute responses from two-choice reaction time (RT) tasks. When functions are differentiable, in majority of the cases it is relatively easy to find a global minimum of those functions using differential equations. For decision making models, having differentiable functions translates into straightforward parameter recovery from experimental data. When non-differentiable models, like diffusion model for conflict tasks, are fitted to experimental data, global optimization algorithms have to be utilized instead, without any guarantees that they are able to find a global minimum of the objective function.

The goal of this chapter is therefore to investigate what effect individual parameters of diffusion model for conflict tasks have on RT distributions from two-choice RT tasks, and to establish whether each parameter can subsequently be recovered by global optimization algorithms. If a change in one parameter from diffusion model for conflict tasks has unique effect on RT distribution, it should not be confused with changes in other parameter or combination of parameters. Conversely, if effects of one parameter can be mimicked by some other parameters, then recovery of that parameter might be difficult.

Some research has already investigated parameter recovery from non-differentiable decision making models like diffusion model for conflict tasks (White et al., 2018) or leaky competing accumulator (LCA; Miletić et al. (2017), but they employ global optimization algorithms for this task. The approach in this chapter instead takes a step back and considers the objective function response surface that the global optimization functions later explore. Therefore instead of presuming that if there are global minimum in response surface, the global optimizer should find it, the work presented here will look at the response surface and establish whether global minimum does indeed exist, and whether global optimization of parameter recovery from diffusion model for conflict tasks is a worthwhile pursuit.

The most intuitive and arguably the most widely used approach in global optimization when objective function is non-differentiable is grid search. In grid search, a set of values is chosen for each parameter comprising the function, and the function is evaluated at each combination of all the chosen parameter values. Grid search allows to equally cover the response surface by all parameters. Choosing the values of parameters, however, involves some manual optimization. Grid search has some big advantages, including low technical overheads, simple to implement parallelization, reliable performance, and having no approximations or underlying assumptions about the response surface. However, grid search has a huge disadvantage of the curse of dimensionality (Bellman, 1961): as the number of parameters increases, and as the sampling rate within each parameter increases, the number of evaluations increases exponentially. There are also arguments that lack of underlying assumptions and the systematic covering of the search space can be a disadvantage in global optimization due to its inefficiency compared to other approaches (Bergstra & Bengio, 2012).

Even though grid search is not the most advanced or the most efficient approach to global optimization, it allows evaluation of the response surface of objective function and provides insight into whether parameter recovery is possible from diffusion model for conflict tasks given current optimization procedures. Using any optimization algorithms to recover parameters from non-differentiable models like diffusion model for conflict tasks involves computing an RT distribution with given set of model parameters and comparing that distribution to an "experimental" RT distribution (the RT distribution from which the parameters need to be recovered). The comparison can involve any function, like maximum likelihood estimation (Myung, 2003), Kolmogorov-Smirnov test of equality (Voss, Rothermund, & Voss, 2004), or any other deviance-based method. This process continues until a set of parameters is found that produces an RT distribution closely matching that of the "experimental" RT distribution. How the optimization algorithm chooses a set of parameters to evaluate is the main difference between different optimization algorithms, however they all include comparison of RT distributions. Due to this, if two or more sets of parameters from diffusion model for conflict tasks produce RT distributions that are similar enough for a comparison function to judge them as same, no optimization algorithm can discriminate between these sets of parameters to fit "experimental" data.

Grid search is therefore a good method to inspect whether diffusion model for conflict tasks objective function response surface has a global minimum that can be later detected by optimization functions, or whether the response surface is comprised of a number of similar-looking local minima with no discernible global one. As diffusion model for conflict tasks has a high number of free-varying parameters, evaluating how each combination of parameters can trade-off in grid search approach is computationally unfeasible, therefore only changes in single parameters will be inspected in this chapter. This will be done by taking a set of parameters called "benchmark" parameters, varying one parameter at a time, and seeing whether the resulting change in RT distribution can be mimicked by changing remaining six parameters while keeping the former one the same as in benchmark set. If single parameters can be mimicked by combinations of other parameters, it is highly likely that changes in two or more parameters will be equally hard to distinguish from other sets of parameters, because that would mean that the effects individual parameters have on reaction time distributions is not unique. If, however, individual parameters do have distinguishable effects on reaction time distributions, changes in two or more parameters would be harder to mimic than changes in single parameters.

The drawback of this approach is that because diffusion model for conflict tasks has seven free-varying parameters (due to shape of atomatic activation being fixed to the same value throughout this thesis, as in Ulrich et al. (2015)), using only seven values of each parameter creates a huge number of grid intersections, but the step sizes between consecutive values of each parameter are relatively big. Due to the crudeness of the constructed grid, a finding that change in one parameter cannot be matched by other parameters is not a definite conclusion that no matching distributions exist. Conversely, finding a few matching distributions might mean that there are hundreds or thousands of others, given that the overall response surface is continuous while grid search only evaluates discrete points on the response surface.

2.2 Methods

2.2.1 Parameter space of diffusion model for conflict tasks

In order to see how parameter values and resulting RT distributions translate to response surface, seven values of seven main parameters of the diffusion model for conflict tasks were selected, equally spaced between chosen limits. The values for each parameter are given below:

- Upper threshold (β_1): $[20, 26.667, 33.333, 40, 46.667, 53.333, 60]$

- Non-decision time (μ_R): $[200, 233.333, 266.667, 300, 333.333, 366.667, 400]$
- Standard deviation of non-decision time (σ_R): $[20, 25, 30, 35, 40, 45, 50]$
- Drift rate for controlled process (μ_C): $[0.4, 0.5833, 0.7667, 0.95, 1.1333, 1.3167, 1.5]$
- Amplitude of automatic activation (A):$[10, 16.667, 23.333, 30, 36.667, 43.333, 50]$
- Time to peak automatic activation (τ): $[40, 73.333, 106.667, 140, 173.333, 206.667, 240]$
- Shape of starting point distribution (α): $[1, 1.333, 1.667, 2, 2.333, 2.667, 3]$

The shape of automatic activation was fixed to 2 as that was the case in Ulrich et al. (2015). The diffusion constant was set to 4. The parameter limits were chosen manually to cover a large space while maintaining plausible parameter values. Every possible combination of parameter values was used, which resulted in $7^7 = 823,543$ unique set of parameters. Each of these sets was then used to compute an RT distribution comprising of 5,000 trials per condition, for three conditions: congruent, incongruent, and neutral. The difference between conditions was determined by the value of the amplitude of automatic activation parameter; in neutral condition, this value was set to 0, while in incongruent condition it was set to -A. This resulted in RT distributions of $15,000$ trials each. The decision process was limited to 1,000 milliseconds, and sampling was done ten times every millisecond, which resulted in 10,000 evidence accumulation points in every trial. If the evidence has not reached either boundary in this time, the trial outcome was recorded as NaN. This corresponds to data collected in chapters section 6 and section 8, where decisions were limited to 1.5 seconds. The CYTHON code to compute a single trial of the conflict diffusion model for conflict tasks with given model parameter values is provided in https://github.com/SolVG/pyCDM/blob/master/maketrial.pyx [1].

One distribution in the "middle" of the parameter space was chosen as the benchmark distribution (with parameters $\beta_1 = 40, \mu_R = 300, \sigma_R = 35, \mu_C = 0.95, A = 30, \tau = 140, \alpha = 2$). Then, 42 distributions were chosen in such a way that each distribution deviated from the benchmark distribution in one parameter only by a certain step. There were six possible steps that the distribution could make, noted as $[+3, +2, +1, -1, -2, -3]$ in following text and figures. The sign indicates whether the parameter in question was higher [+] or lower [$-$] compared to the same parameter of

[1]Clickable links in PDF

the benchmark distribution. The number indicates the size of the difference between the parameters in comparison distribution and the benchmark distribution. As the parameters were chosen from a selection of seven possible values, $+3$ indicates the largest parameter value, -2 indicates the second to smallest parameter value and so on. So for example, $\beta + 2$ distribution means that the upper threshold parameter was moved two steps up from 40 in the benchmark distribution to 60 in the comparison distribution. As each parameter could take six possible steps, and there were seven parameters, this resulted in 42 comparison distributions.

Then, each of the comparison distributions was compared against every other distribution in the parameter space, except for the ones that deviated in the same parameter as the comparison distribution. For example, $\beta + 3$ distribution was compared to every other distribution in which the upper threshold parameter was the same as the upper threshold parameter value of the benchmark distribution. This allowed to investigate whether a change in one parameter could be mimicked by a change in other parameters but the parameter in question, for example, can a change in the upper threshold parameter be mimicked by one or multiple other parameters other than upper threshold?

2.2.2 Reducing the number of trials per distribution

In order to see what influence does the trial number per condition has on recovery of parameters, subsampling was performed on distributions with 5,000 trials per condition to create distributions with lower trial numbers. This was accomplished by iterating through the distributions in the search space (823,543 in total), separating each one into congruent, incongruent, and neutral conditions, and randomly subsampling a smaller number of trials per condition with replacements. The smaller number of trials were chosen as 50, 75, 100, 125, 150, 200, 250, 500, 750, 1000, 1500, 2000, 2500, 3000, 3500, 4000, and 4500. Each subsample was chosen independently for each number of trials, condition, and distribution in the search space. This method was chosen over creating new distributions of smaller numbers of trials as it was computationally more efficient while providing sufficient randomness in data.

2.2.3 Objective function

In order to make comparisons between distributions, Kolmogorov-Smirnov test (KS) (*Encyclopedia of Mathematics, Kolmogorov–Smirnov test*, n.d.) was employed. KS test of equality of distributions is a good choice for decision making modelling as it allows comparing whole distributions without splitting them into bins, and also allows combining correct trials and error trials into a single distribution by giving the errors a negative RT (Voss et al., 2004).

Three separate KS values were calculated for congruent, incongruent, and neutral conditions, and then summed to produce a single KS output. In order to determine what KS test value could represent no difference between two distributions, a thousand distributions sharing the same parameters as the benchmark distribution was computed, and KS test was performed between the benchmark distribution and each of the thousand distributions. This resulted in one thousand KS values that compare the same distribution. The maximum of these values was chosen. Then, the process was repeated one hundred times to account for randomness in computations, resulting in one hundred maximum KS values. The 95th largest value was chosen as a cutoff value to call two distributions as matching.

Similarly, for distributions with lower number of trials, the KS statistic was computed by generating the benchmark distribution once with the lower number of trials, and then the same benchmark distribution 1,000 times with 5,000 trials per condition, then the process repeated one hundred times. This was done because number of trials in experimental data are hard to increase in certain study designs, while the number of trials in simulated data can be selected as needed. Moreover, as fitting "experimental" data to theoretical distributions presumes that the theoretical distribution is known, simulating large numbers of trials allows for more stable, theoretical-like distributions.

Finally, for low number of trials in experimental conditions, fitting with large theoretical distributions might not be the most computationally efficient if very large numbers of trials (say 5,000) do not offer any advantage over fitting with just a large numbers of trials (say 1,000). For this reason, for distributions with trial numbers below 1,000, two other sizes of simulated distributions were used: 1,000 trials per condition (fixed size) and 4× the number of trials per condition in "experimental" data (variable size). Four was chosen as the multiplier to have a reasonably large effect on the resulting size of the distribution. The cutoff KS values for all these conditions are presented in table 2.1.

Table 2.1: KS value cutoffs to evaluate whether RT distributions are the same for varying number of trials per condition in the benchmark distribution (1st column, and different number of trials in comparison distributions.

Trials in benchmark distribution	5,000 trials in comparison distribution	1,000 trials in comparison distribution	$4x$ trials in comparison distribution
50	0.5074	0.6460	0.7600
75	0.4121	0.4684	0.5534
100	0.3726	0.3700	0.4425
125	0.3232	0.3320	0.3880
150	0.3129	0.3397	0.3634
200	0.2592	0.3010	0.3138
250	0.2510	0.2870	0.2780
500	0.1902	0.2180	0.1970
750	0.1748	0.1770	0.1387
1000	0.1546	-	-
1500	0.1323	-	-
2000	0.1173	-	-
2500	0.1136	-	-
3000	0.1040	-	-
3500	0.1019	-	-
4000	0.0969	-	-
4500	0.0954	-	-
5000	0.0930	-	-

2.3 Results

2.3.1 Do diffusion model for conflict tasks parameters have unique effects on RT distributions?

A KS test statistic was computed between each comparison distribution and the rest of the parameter space, as described in section 2.2.1. If the KS value was equal to or below the KS value for 5,000 trials in benchmark distribution and 5,000 in comparison distribution as reported in table 2.1, the distribution in parameter space was considered as a matching distribution to the comparison distribution. The parameter values for all matching distributions are reported for each diffusion model for conflict tasks parameter separately in figure 2.1 to figure 2.5. The figures show the number of matching distributions on the y axis that were found for each comparison distribution (horizontal panels) and the difference between the parameter values of the matching distributions and the benchmark distribution on the x axis, in facets for each model parameter. Non-decision time and drift rate for controlled process had no matching distributions, therefore they have no figures. If there is no panel in a figure for a specific comparison distribution, that means that there were no matching distributions found for that comparison distribution.

These figures do not display all the combinations of parameters from the matching distributions, but some insights about parameters can be made. For example, in figure 2.1 upper facets, it can be observed that all the matching distributions for $\beta - 1$ comparison distribution had non-decision time that was the same as in the benchmark distribution (upper leftmost facet, single bar histogram at 0), but the values for amplitude of automatic activation (4^{th} upper panel from the left) and time to peak automatic activation (5^{th} upper panel from the left) were approximately uniformly distributed.

Out of seven diffusion model for conflict tasks parameters, changes in two parameters were always unique: the non-decision time and drift rate for controlled process. For each comparison distribution that varied either one or the other parameter, there were no matching distributions in the parameter space. Therefore it is likely that these two parameters should be easily recovered by global optimization algorithms. Other five parameters did have some matching distributions, as shown in figure 2.1 to figure 2.5. Upper threshold and variability of non-decision time had matching distributions in the smallest step conditions (+1 and -1), while other three parameters had matching distributions in more extreme steps as well. Even though the number of matching distributions are small given possible search space (exact number of matching distributions are reported in table 2.2), the proportion of possible matches is not that important given that this was done in a grid space with discrete parameter values, whereas in parameter recovery situation the search space is continuous, therefore it is likely that each matching distribution would in fact produce clouds of thousands of distributions with very similar parameter values. Therefore recovery of these five parameters could be problematic for

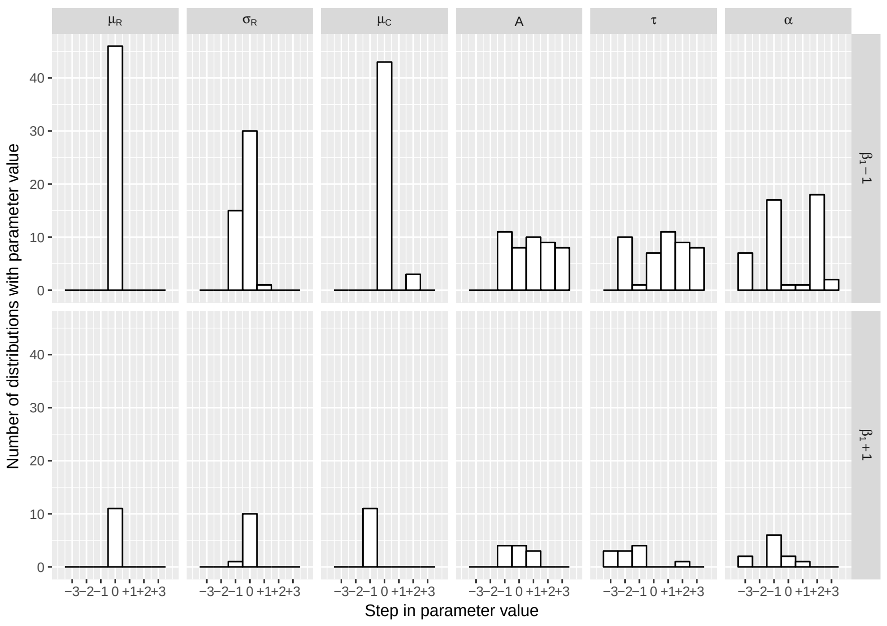


Figure 2.1: Parameter values of matching distributions for upper threshold comparison distributions.

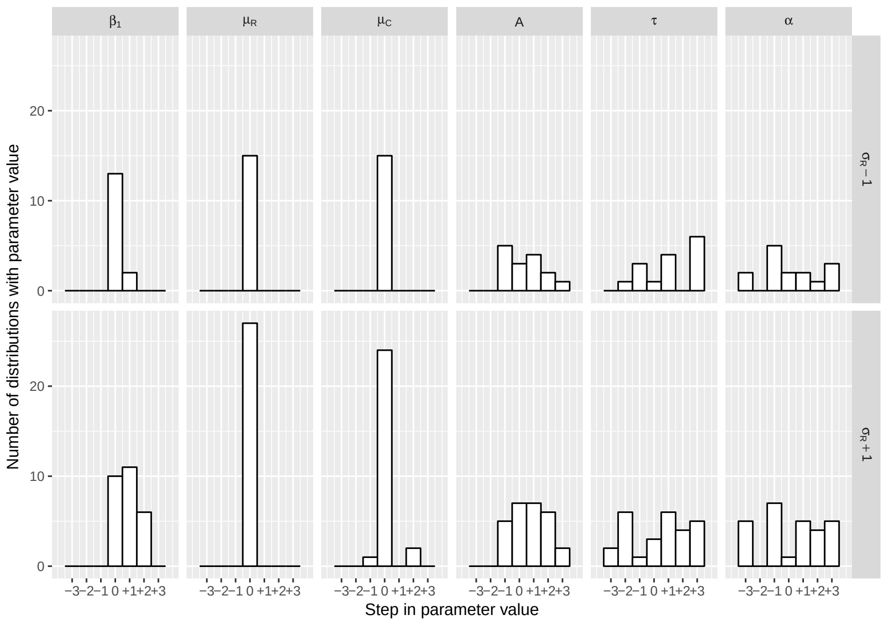


Figure 2.2: Parameter values of matching distributions for non-decision time variability comparison distributions.

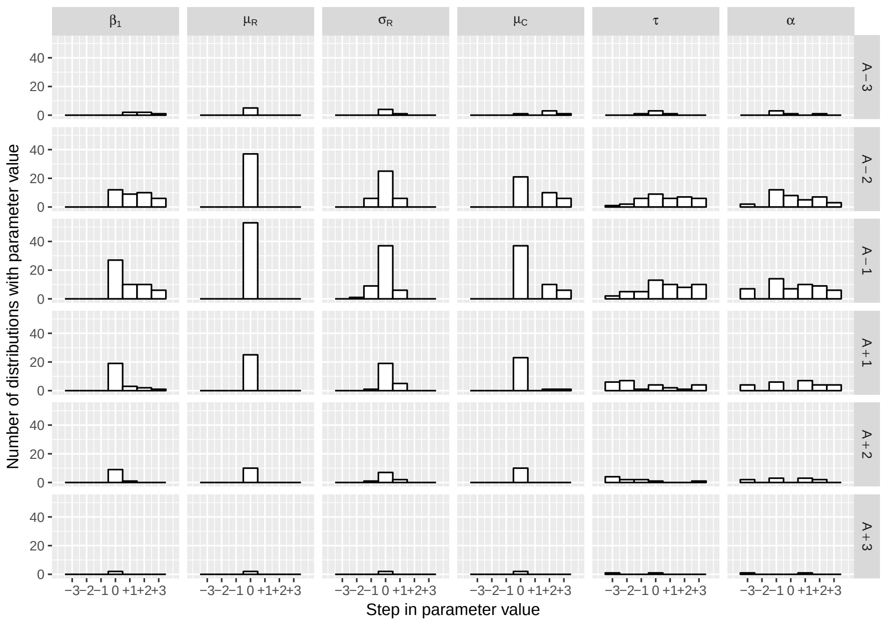


Figure 2.3: Parameter values of matching distributions for amplitude of automatic activation comparison distributions.

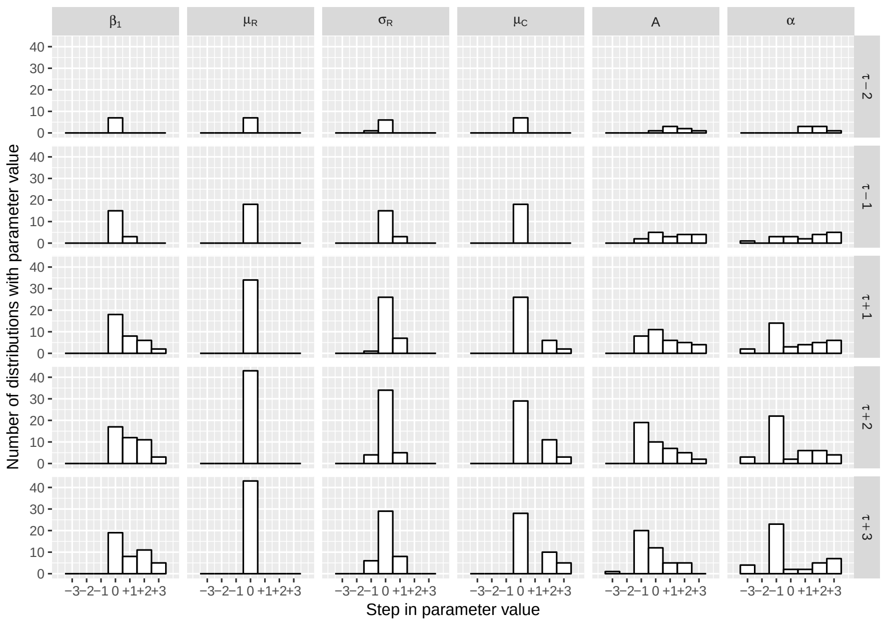


Figure 2.4: Parameter values of matching distributions for time to peak automatic activation comparison distributions.

global optimization algorithms.

The figure 2.1 to figure 2.5 also display the values that the parameters in matching distributions take. As evident in the vertical panels representing each parameter that was allowed to change, some parameters always have to take certain values (e.g. non-decision time always has to be the same as in the benchmark distribution), some have minimal variability across all comparison distributions (like variability of non-decision time and drift rate for controlled process), while others can take multiple values spread across search space to produce matching distributions (e.g. time to peak automatic activation, shape of starting point distribution). This could point to the possibility that parameters with the most uniform distributions (e.g. time to peak automatic activation, shape of starting point distribution) could be the most difficult to recover for global optimization algorithms.

Table 2.2: Number of matching distributions per comparison distribution.

Comparison distribution	Number of matches	Comparison distribution	Number of matches	Comparison distribution	Number of matches
β-3	0	σ_R-3	0	A-3	5
β-2	0	σ_R-2	0	A-2	37
β-1	46	σ_R-1	0	A-1	53
β+1	11	σ_R+1	0	A+1	25
β+2	0	σ_R+2	0	A+2	10
β+3	0	σ_R+3	0	A+3	2
total β_1	57	total σ_R	0	total A	132
μ_R-3	0	μ_C-3	0	τ-3	0
μ_R-2	0	μ_C-2	0	τ-2	7
μ_R-1	15	μ_C-1	0	τ-1	18
μ_R+1	27	μ_C+1	0	τ+1	34
μ_R+2	0	μ_C+2	0	τ+2	43
μ_R+3	0	μ_C+3	0	τ+3	43
total μ_R	42	total μ_C	0	total τ	145
α-3	7				
α-2	17				
α-1	15				
α+1	34				
α+2	34				
α+3	33				
total α	140				

One issue with these results is that RT distributions of 15,000 trials each represent close to “perfect” data. In reality, RT distributions obtained from human studies are very unlikely to have more than 500 to 1,000 trials per condition (given an average 1.5s per trial for speeded reaction time tasks, 1,000 trials would take 25 minutes with no breaks to administer), and clinical samples usually have even smaller trial numbers (e.g. J. Zhang et al. (2015) had 300 trials of Go/NoGo task per session with Parkinson’s patients; Dillon et al. (2015) had 350 trials in total for two conditions of flanker task in patients with clinical depression; in Karalunas and Huang-Pollock (2013) study children with ADHD performed 200 trials of Logan Stopping Task). Therefore the problem of matching distributions is likely to be more severe when parameter recovery from real experimental data is attempted. For this reason, constraining some parameters is necessary in order to make the rest recoverable to global optimization algorithms.Section 2.3.3 is going to discuss this issue for distributions with smaller number of trials per condition.

2.3.2 Randomness in computation and variability in distributions

It must be noted that there is inherent variability in the computation of the distributions that comes from randomness in computing individual trials. This could have an effect on the results of matching distributions as they might be due to chance because each comparison distribution was only calculated once. This should not be an issue of the parameter space distributions, as there were 823543 of them, therefore a few random extreme distributions should not have an impact on the overall outcome. However, there were only 42 comparison distributions, and any statistical deviation in them could have big subsequent effects on the results. Due to this, the comparison distributions were computed five times to see whether resulting matching distributions were different, and consequently whether randomness could introduce enough variability to make results unstable. Figure 2.6 to figure 2.10 shows that fortunately randomness in trial computation does not have big effect on the results, as all five distributions match almost exactly. Therefore only one of the comparison distribution sets was used in the following sections.

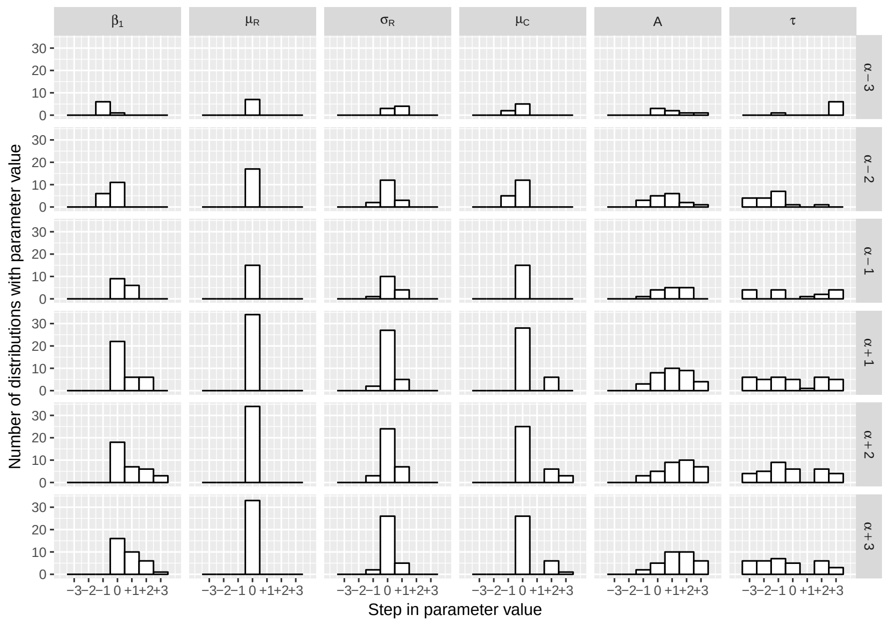


Figure 2.5: Parameter values of matching distributions for shape of starting point distribution comparison distributions.

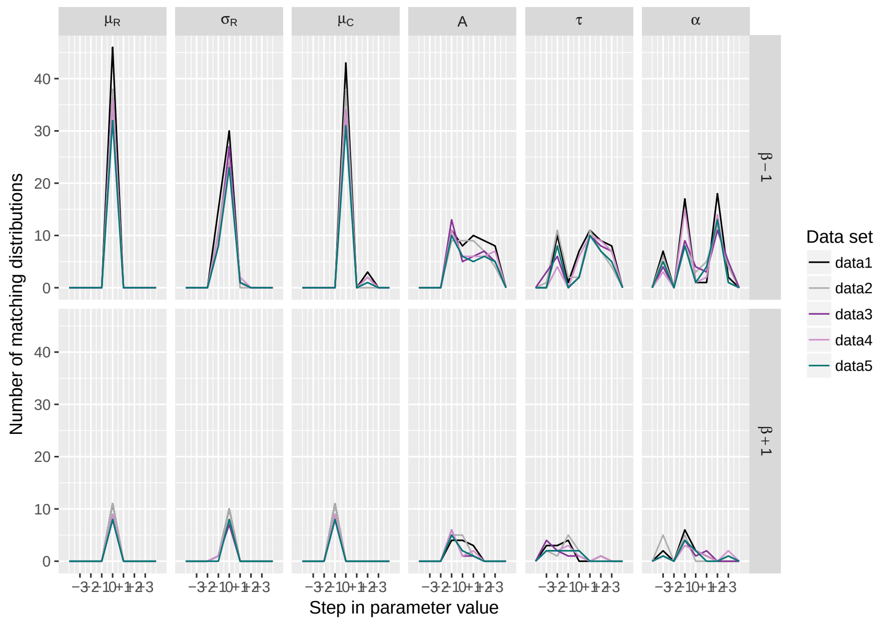


Figure 2.6: Parameter values of matching distributions for upper threshold comparison distributions over five independent samples of comparison distributions.

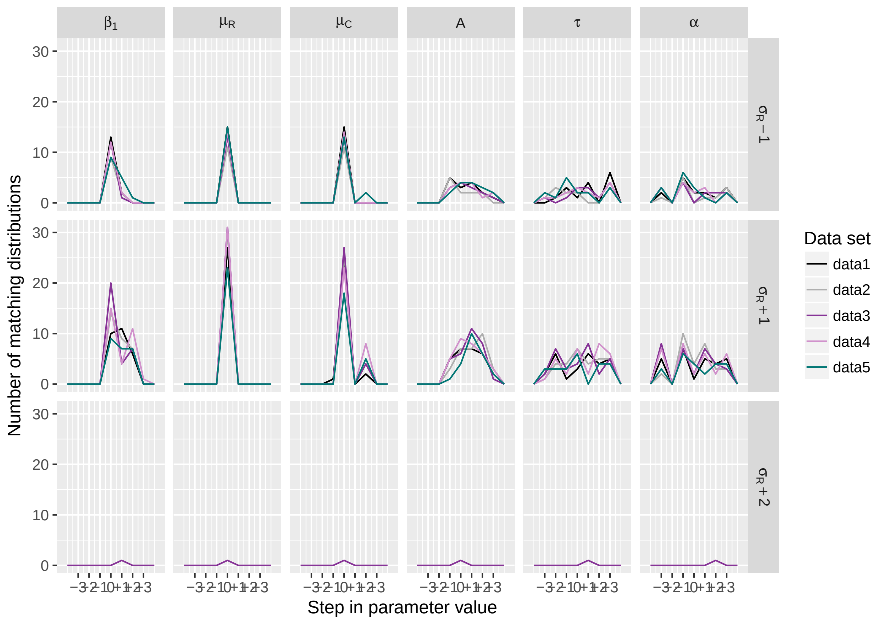


Figure 2.7: Parameter values of matching distributions for non-decision time variability comparison distributions over five independent samples of comparison distributions.

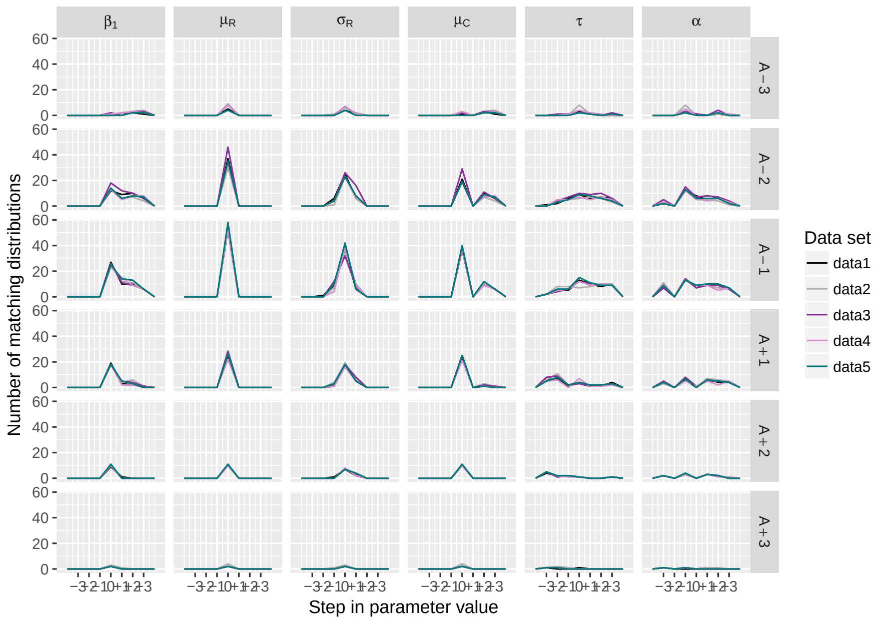


Figure 2.8: Parameter values of matching distributions for amplitude of automatic activation comparison distributions over five independent samples of comparison distributions.

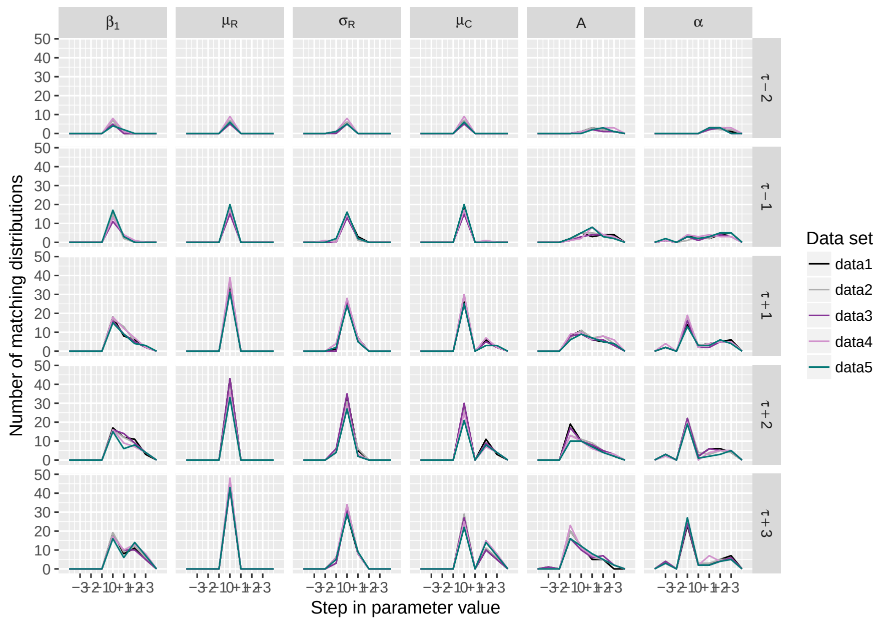


Figure 2.9: Parameter values of matching distributions for time to peak automatic activation comparison distributions over five independent samples of comparison distributions.

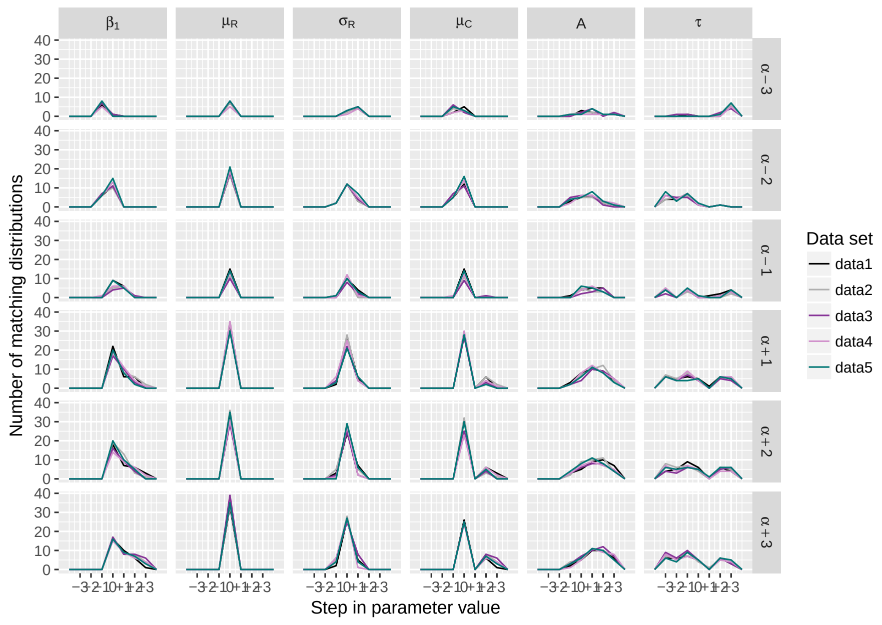


Figure 2.10: Parameter values of matching distributions for shape of starting point distribution comparison distributions over five independent samples of comparison distributions.

2.3.3 Constraining parameters

In order to compare the number of matching distributions for distributions with varying number of trials per condition, sampling with replacements was performed from the RT distribution with 5,000 trials per condition to obtain RT distributions with 50, 75, 100, 125, 150, 200, 250, 500, 750, 1,000, 1,500, 2,000, 2,500, 3,000, 3,500, 4,000, and 4,500 trials per condition. Sampling was done independently for each condition. The KS values chosen as cutoffs to judge whether two distributions were matching or not are shown in table 2.1.

In order to choose which diffusion model for conflict tasks parameters to constrain, the relative importance of each parameters needs to be considered. Upper threshold, non-decision time and drift rate for controlled process are the three main parameters underlying any DDM-based model. The amplitude of automatic activation is one of the parameters that discriminates the diffusion model for conflict tasks from DDM, and therefore very important when considering response inhibition tasks. Time to peak automatic activation is also quite important, as it is the parameter that discriminates between different response inhibition tasks (Ulrich et al., 2015). However, when only one task is considered, this parameter could be fixed to a value that approximates the mean for that task. Standard deviation of non-decision time provides better fit of real data, however is not a very interesting parameter in itself to describe the decision making process. Finally, the shape of starting point distribution is difficult to interpret as a cognitive process, and therefore should probably be the first parameter considered for constraining.

Three parameters were chosen for constraining: shape of starting point distribution, time to peak automatic activation, and standard deviation of non-decision time. When each of the parameters was constrained, the parameter search space was reduced in such a way that each comparison distribution was evaluated against every distribution in parameter space where the varied parameter in comparison distribution matched the benchmark distribution, and the constrained parameter matched the benchmark distribution. For example, to evaluate the effects of constraining the shape of the starting point distribution on upper threshold recovery, $\beta - 1$ comparison distribution was evaluated against every other distribution in the parameter space where $\beta_1 = 40$ and $\alpha = 2$ (same values as benchmark distribution).

To examine the effects of constraining multiple parameters, two combinations were chosen: constraining the shape of the starting point distribution and time to peak automatic activation, and the shape of the starting point distribution and standard deviation of non-decision time. The calculations were similar to constraining one parameter, for example, to evaluate the effects of constraining the shape of the starting point distribution and time to peak automatic activation on upper threshold recovery, $\beta - 1$ comparison distribution was evaluated against every other distribution in the parameter space where $\beta_1 = 40$, $\alpha = 2$, and $\tau = 140$ (same values as benchmark distribution). Figure 2.11 displays results of constraining parameters (x axis) on the the number of matching distributions per comparison distribution (y axis) for varying number of trials per condition (colours in the figure). As the results are difficult to see for higher number of trials per condition, they are displayed in figure 2.12. The figures show that as the number of trials per condition decreases, the number of matching distributions increases. This is true when no parameters are constrained (leftmost group in figure 2.11) and when either one or two parameters are constrained (all remaining groups). Exponential decay functions fitted to the data to demonstrate that as trial number per condition increases, the number of matching distributions per comparison distribution decays to zero for all constraint conditions. This is shown in figure 2.13.

It is also evident from the figure 2.11, figure 2.12, and figure 2.13 that the number of matching distributions per comparison distribution is higher when no parameters are constrained compared to when one or two parameters are constrained. This means that recovery of parameters should become harder with decreasing number of trials per condition and when no constraints on parameters are imposed.

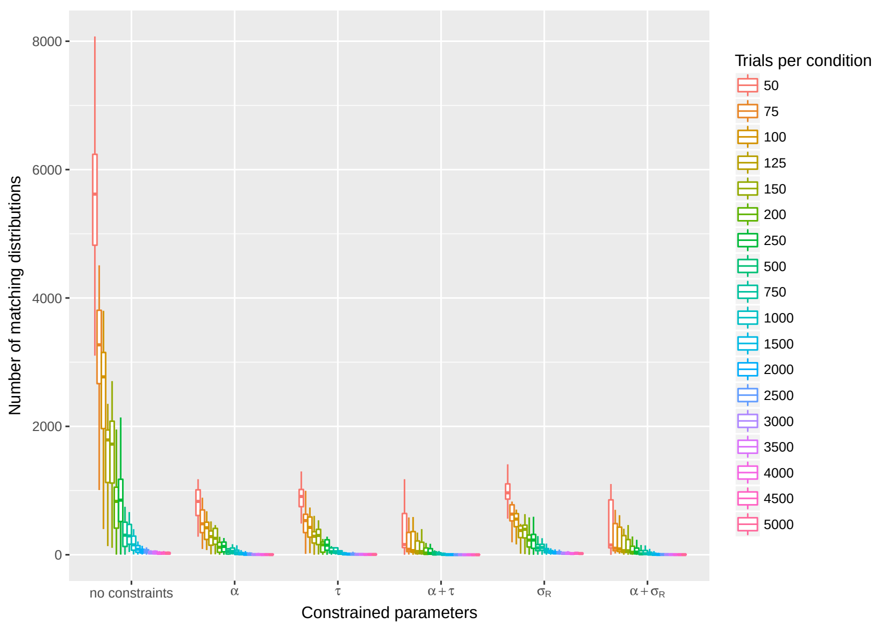


Figure 2.11: Proportion of matching distributions for different constraints and trial types. Note that outliers are not displayed.

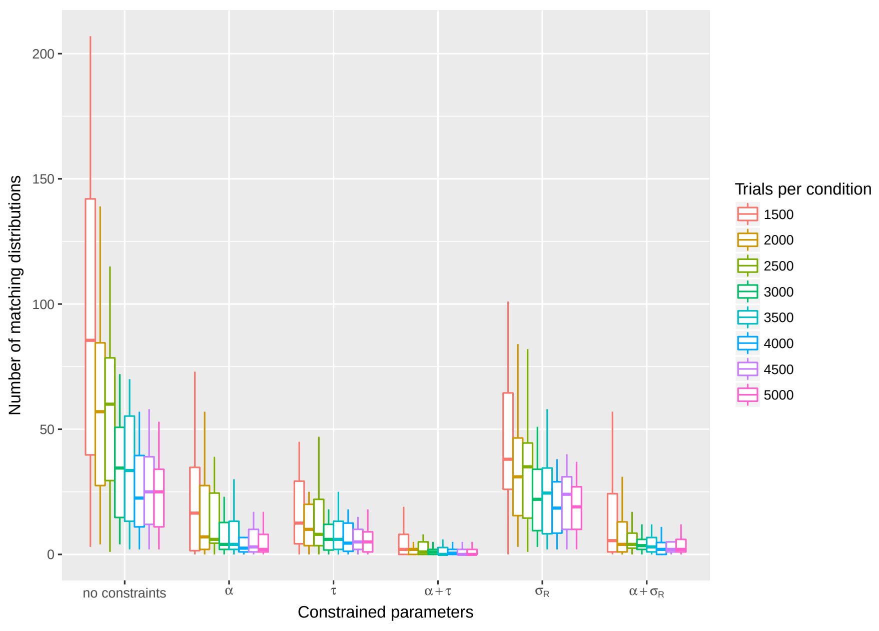


Figure 2.12: Number of matching distributions for different constraints and trial types, shown for the larger number of trials only to see details. Note that outliers are not displayed.

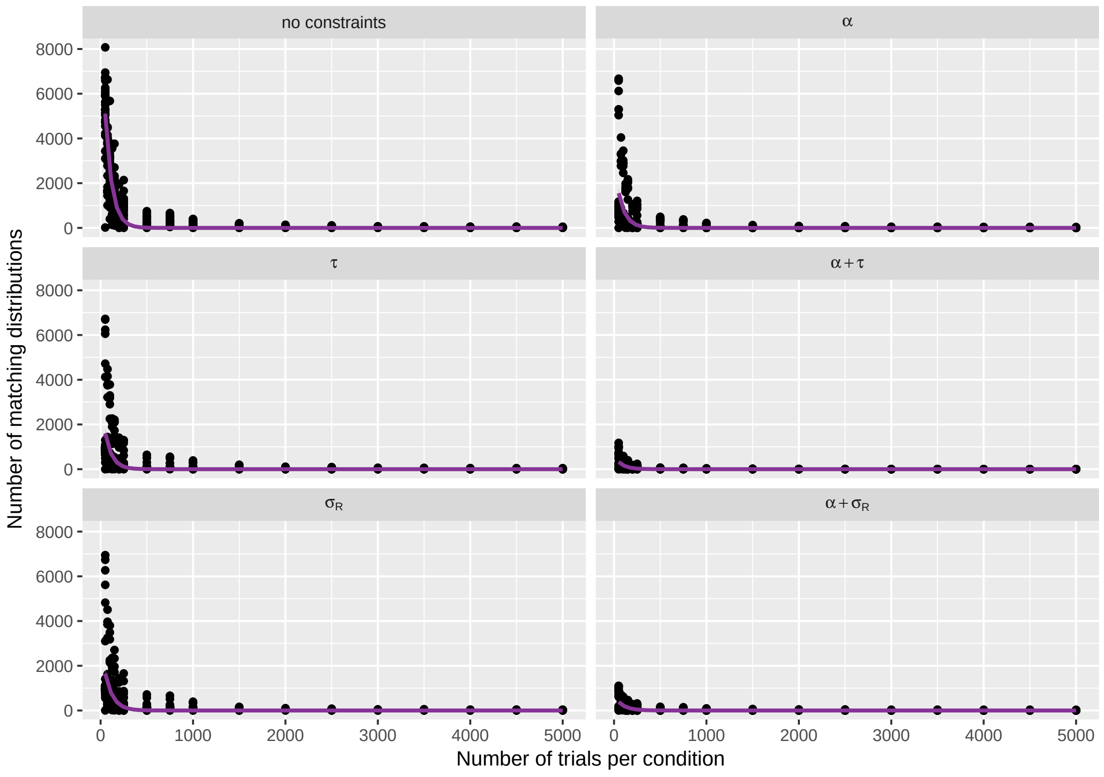


Figure 2.13: Exponential decay functions fitted to number of matching functions as number of trials per condition increases for each constraint.

Note that comparison between constraining one and two parameters versus no constraints is difficult because the parameter space is larger with no constraints compared to one parameter or two parameters constrained, therefore figure 2.11 and figure 2.12 can be scaled by the number of distributions in the parameter space. The results of this are shown in figure 2.14 for all data and figure 2.15 for higher number of trials per distribution.

Figure 2.14 and figure 2.15 show that when the proportion of matching distributions to the size of parameter space is considered, constraining one or two model parameters does not reduce the proportion (median bars in boxplots), and appears to increase variability of proportion of matching distributions per comparison condition (size of boxplots and size of whiskers). Whether the proportion of matching distributions or the actual number of matching distributions is more relevant for global optimization algorithms is discussed in section 2.4.

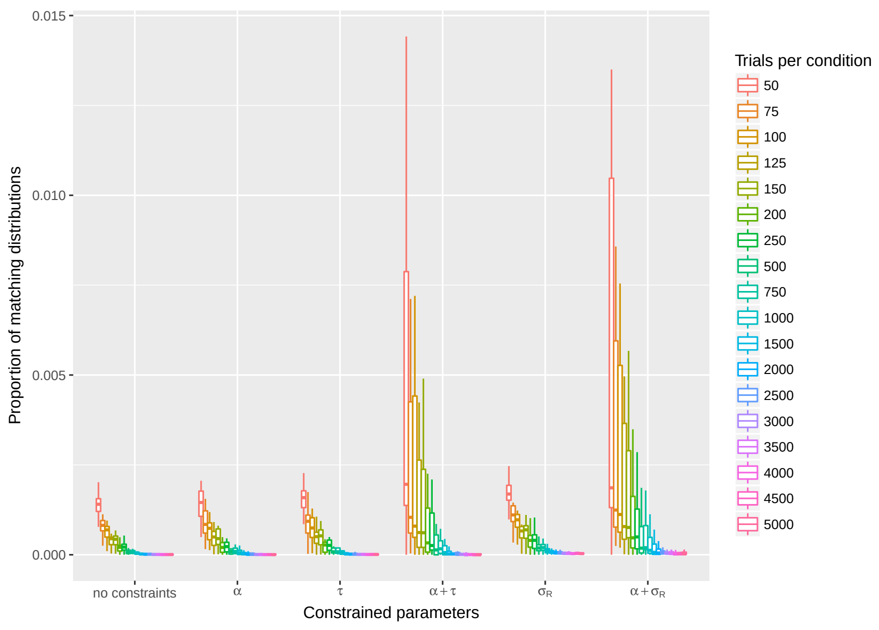


Figure 2.14: Number of matching distributions for different constraints and trial types, scaled per size of available distribution pool

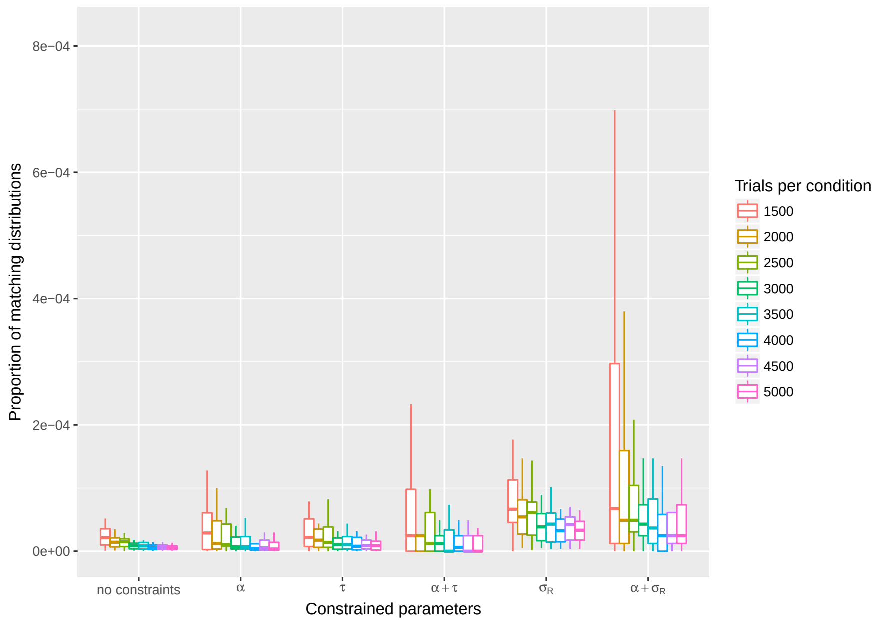


Figure 2.15: Number of matching distributions for different constraints and trial types, shown for the larger number of trials only to see details, scaled per size of available distribution pool

2.3.4 Is high number of trials needed for small distributions?

When diffusion model for conflict tasks parameters need to be estimated from data with small numbers of trials per condition, simulating distributions in the solution space with 5,000 trials per condition might not be computationally most efficient if such large distributions do not offer any advantage over smaller distributions. Therefore in this section three different sizes of simulated distributions, 5,000 and 1,000 trials per condition, and four times the number of trials per condition in “experimental” data, were compared to investigate whether the size of the simulated distribution has an effect on the number of matching distributions for ”experimental” data with small number of trials ($< 1,000$). This effect was inspected in conditions with no parameter constraints, single parameters constrained, and two parameters constrained. The results are displayed in figure 2.16.

Figure 2.16 demonstrates that for all constraint conditions, when “experimental” data has low trial numbers per condition, simulating distributions during fitting procedure with 5,000 trials per distribution offers little advantage over simulating distributions with 1,000 trials per condition. On the other hand, determining simulated distribution size by multiplying the number of trials from ”experimental” data by four seems to increase the number of matching distributions for datasets with very low trials per condition (< 250). This suggests that there might be computational advantages to reducing the size of simulated distributions from 5,000 trials per condition to any number up to 1,000 trials per condition when the “experimental” data contains low trial numbers, without much detriment to the accuracy of the global optimization algorithms.

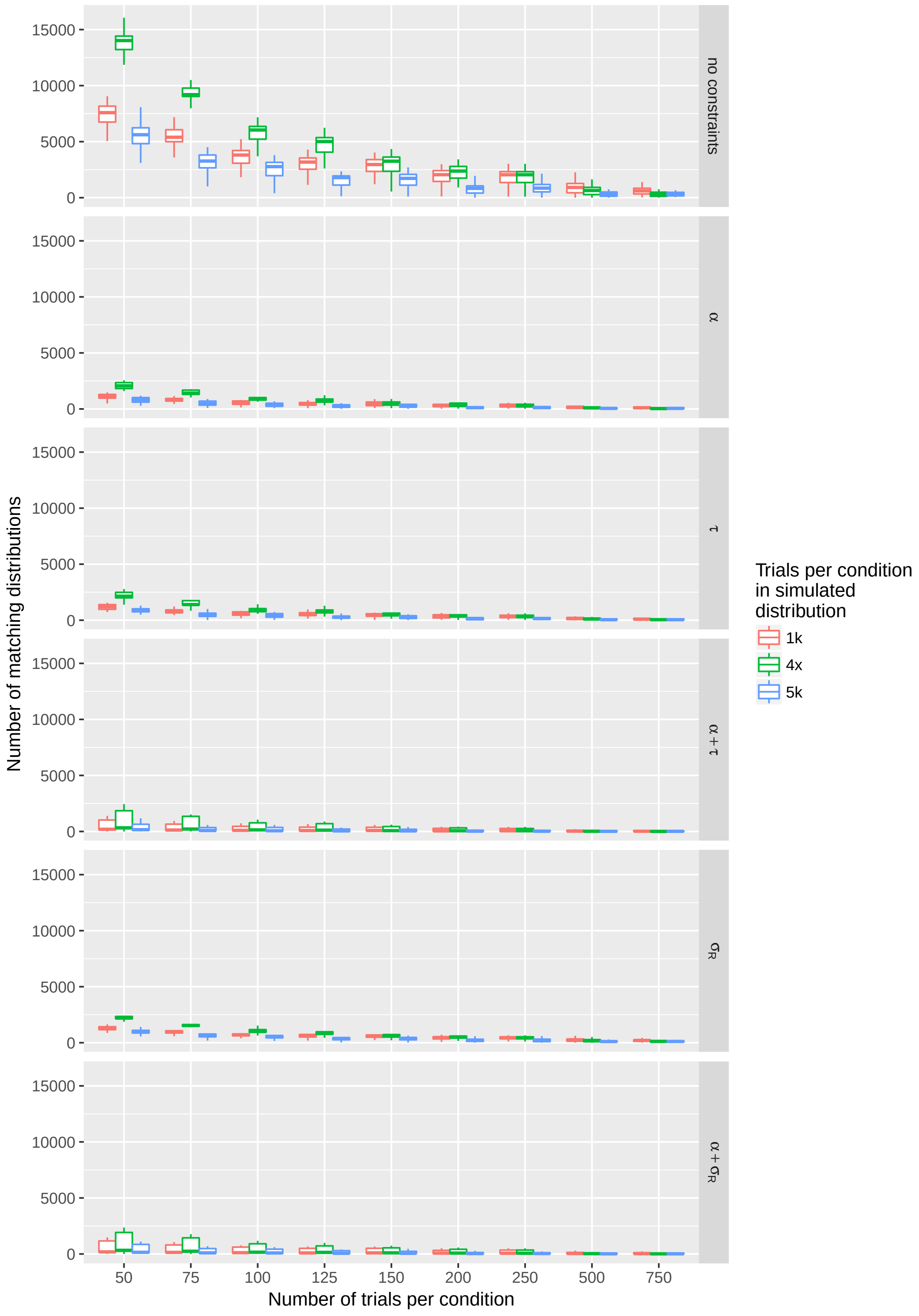


Figure 2.16: Effect of reducing simulated distribution size on number of matching distributions for comparison distributions with low trial numbers.

2.4 Conclusions

This section has used grid search methodology to inspect the solution surface of the diffusion model for conflict tasks parameter space and found that changes in five out of seven diffusion model for conflict tasks parameters can be mimicked by changes in other parameters. This means that any global optimization algorithm will struggle to find a solution to the objective function of the diffusion model for conflict tasks fitting procedure.

Two out of the seven diffusion model for conflict tasks parameters, non-decision time and drift rate for controlled process, had no matching distributions when each of these parameters was varied. This could imply that both of these parameters have unique effects on RT distributions that are impossible to mimic with combinations of other parameters, and hence they should be easily recoverable by global optimization algorithms. Alternatively, no matches for these two parameters could have resulted from the crudeness of the grid space itself that was used: with only seven values per parameter, and relatively large parameter limits, there could be matching distributions if the resolution of the grid was finer.

Moreover, it is also unclear to what extent changes in one parameter can be confused with different changes in the *same* parameter plus some others. This is one of the main differences between the grid search procedure used in this chapter to evaluate matching distributions, and that of any global optimization algorithm. Therefore in reality when parameters need to be estimated from data, the solution surface of the diffusion model for conflict tasks objective function is likely to be even harder for finding the global minimum.

In order to see whether limiting the dimensions of the solution space by constraining one or two parameters of the diffusion model for conflict tasks to a fixed value might aid recovery of other parameters, several single parameters (shape of starting point distribution, time to peak automatic activation, standard deviation of non-decision time) and combinations of two parameters (shape of starting point distribution and time to peak automatic activation, or shape of starting point distribution and standard deviation of non-decision time) were fixed in the parameter space. At the same time, to investigate the effects of the size of the "experimental" data on parameter recovery, the above investigation was repeated on distributions with trial numbers that varied from 50 to 4,500 per condition. The results showed that constraining at least one parameter was beneficial to recovery of other parameters as the number of matching distributions decreased when parameters were constrained. At the same time, reducing the trial number per condition of the "experimental" data has severely increased the number of matching distributions, meaning that studies with low trial numbers would have difficulties in estimating diffusion model for conflict tasks parameters due to the surface of the solution space.

Selection of which parameters to constrain should be at the discretion of the researcher. Time to peak automatic activation parameter (τ), although hard to recover, might be of interest to some researchers who are investigating the differences between two or more response inhibition tasks, as this parameter was found to discriminate between flanker and Simon tasks in the paper by Ulrich et al. (2015). Moreover, it could be of interest to researchers investigating individual differences in inhibition, due to its' effect on delta plots, which are used in examining the effects of inhibition and impulsivity (Ambrosi et al., 2019; Suarez et al., 2015; de Bruin & Della Sala, 2017). Therefore, if time to peak automatic activation is fixed, the value that should ideally correspond to the values comparable with the response inhibition task in mind. Similarly, the variability of non-decision time parameter (σ_R) might be of interest to researchers who are investigating the variability of responding, but are not that concerned with true estimations of time to peak automatic activation. Therefore either of these parameters can be chosen to be fixed. Alternatively, shape of starting point activation does not seem to offer any meaningful interpretations of cognitive processes, and appears to be the most straightforward choice for constraining.

An interesting result that came from this section was that although the actual number of matching distributions decreased with constraining parameters, this was not the case when proportion of matching distributions to the size of the parameter space was considered. This could have resulted from the fact that when high-dimensional spaces are considered, the importance of some parameters to the surface of the solution space is much more pronounced than others (Bergstra & Bengio, 2012). Nevertheless, it is still unknown whether the proportion of matching distributions is as important as the actual number of matching distributions when it comes to parameter recovery from global optimization functions. Response surface of the objective function in diffusion model for conflict tasks is continuous. Given that the most commonly implemented precision for floating point numbers is 64 bits, each parameter of diffusion model for conflict tasks can be coded in 16 digits, meaning 1,000,000,000,000,000 values that each parameter can take. Allowing one less digit to vary to account that some values of the parameters need to fall within specific limits to make sense, means that at least 10^{97} possible unique combinations of parameters exist in the solution space, which is more than the estimated number of atoms in the known universe. Constraining that space in a few axes, even though it would reduce the total number of possible solutions, would not make much actual difference.

Finally, the chapter showed that when parameters are estimated from data with low trial numbers, reducing the number of trials in simulated data does not have a big detrimental effect to the solution space and hence parameter recovery. Therefore to increase the computational efficiency of global optimization algorithms, smaller simulated distributions could be considered.

The next chapter will use the findings from this section and will further investigate the parameter recovery from the diffusion model for conflict tasks in a continuous space. Multiple global optimization algorithms will be compared for precision and speed of performance when trying to estimate parameters from large and small datasets, with and without constrained parameters.

3 Optimization algorithms for model parameter recovery

3.1 Introduction

To apply mathematical models of decision making in psychological research, researchers need ways to extract model parameters from behavioural studies. The process of parameter extraction from drift diffusion class of models is difficult. Even though the classical drift diffusion model (Ratcliff, 1978) is differentiable (which means that a solution can be found using analytic-, or calculation-, based means), approximation, or optimization methods are used instead, due to the complexity of the analytic equations. The expansion of the drift diffusion model to conflict tasks (Ulrich et al., 2015) resulted in the model being non-differentiable, which means that the parameters of the diffusion model for conflict tasks can only be estimated using optimization algorithms. Optimization algorithm, in general terms, is a process that compares many solutions to a problem over multiple iterations, until a satisfactory solution is found.

There are many optimization algorithms, each with its own strengths and weaknesses. Choosing which optimization algorithm to apply for a given problem is difficult, because suitability for a specific problem needs to be weighed against computational time (how long does it take to execute the algorithm), and also against researcher time (how long does it take for the researcher to implement the algorithm): even if an algorithm is excellent at finding global minimum of a problem, but takes weeks, months, or even years to run the total calculations to find that solution, the runtime is going to be unacceptable for practical applications. Likewise, researcher time is very costly, so even if an algorithm is good at finding the global minimum of a problem, and runs in an acceptable timeframe, but takes researchers months or years to implement in production pipelines, it will not be practical to implement it over another, less well performing algorithm that is much cheaper in implementation time.

The diffusion model for conflict tasks does not have an analytic solution, because the derivative of the objective function cannot be calculated. Therefore we cannot rely on classical optimization techniques (which are proven to find a global minimum or a maximum of the problem) to recover parameters from the model, and instead need to apply metaheuristic algorithms, which are used in solving non-linear non-differentiable objective functions to find suboptimal solutions (because is it yet to be proven that the solution found is a global minimum or maximum). The metaheuristic optimization algorithms balance two strategies to search for the global minimum of a function, exploration and exploitation (Civicioglu & Besdok, 2013). Exploration is the process of an algorithm visiting new regions of a search space, and exploitation refers to the process of an algorithm visiting those regions of a search space that are in the neighborhood of previously visited points (Črepinšek, Liu, & Mernik, 2013). In order to be successful, a metaheuristic optimization algorithm needs to establish a good balance between these two processes.

In order to investigate how well optimization algorithms can find parameter solutions from diffusion model for conflict tasks, five different optimization algorithms were selected in such a way as to cover different stochastic optimization algorithm families. First, random search was chosen to serve as the baseline for the optimization solution. Random search is arguably the most basic algorithm, as all it does is pick a random point in the search space, check the objective function value for that particular point, and repeat the process over multiple iterations. The point in solution space with the lowest value of the objective function is then the overall global solution. Random search is incredibly cheap in implementation time, but on the other hand it becomes computationally expensive with increasing solution space.

Evolutionary programming algorithms were first proposed by Fogel, Owens, and Walsh (1966); in these algorithms the principles of biological genetic evolution are applied to solve optimization problems. One of the algorithms belonging to the class of evolutionary algorithms is differential evolution, first introduced by Storn and Price (1997). Differential evolution algorithm was developed for the solution of real-valued numerical optimization problems (Storn & Price, 1997), therefore it is very well suited for parameter recovery from diffusion model for conflict tasks, which is also a real-valued numerical optimization problem. Differential evolution works by initializing a population of solutions, and then creating a new generation of solutions by combining the members of the initial population that have the lowest objective function values. The population of initial solutions is combined into subsequent generations using mathematical expressions for mutation, crossover, and selection strategies of the genetic algorithm. The most important difference of the differential evolution algorithm from the standard genetic algorithm is the strong mutation strategies it has (Storn & Price, 1997). Differential evolution algorithm treats the optimization problem as a black box and therefore does not need the gradient of the problem, which makes it very suitable for non-differentiable problems, such as diffusion model for conflict tasks. Due to its simple mathematical structure, differental evolution algorithm is relatively simple to implement. The algorithm has a great balance between exploitation and exploration, through the implementation of crossover and mutation strategies respectively.

Swarm optimization belongs to category of Swarm Intelligence algorithms (J. F. Kennedy, Kennedy, Eberhart, & Shi, 2001), and was first developed by James Kennedy and Russell Eberhart in 1995 (J. Kennedy, 2011). The Particle Swarm Optimization algorithm is a population-based, stochastic, and multi-agent parallel optimization algorithm (Civicioglu & Besdok, 2013), similarly to the differential evolution algorithm. However, unlike the differential evolution algorithm, particle swarm optimization has no crossover or mutation. Instead, the particle swarm optimization algorithm has mathematical implementation of cognitive components and social components of a "swarm" of solutions, which are modelled on the behaviours displayed by social collections of living creatures, such as swarms of insects and flocks of birds. Such collections display a trade-off between the individual knowledge, as each individual knows the best locations for discovering food from its own personal experience, but also the social knowledge of the swarm as a single system, where the whole swarm or flock can move in a space to locate the best food sources. The mathematical implementation of how both social and cognitive aspects move the swarm over a solution space allows the particle swarm optimization algorithm to cover both exploration and exploitation aspects of metaheuristic optimization.

Another class of optimization algorithms is Markov Chain Monte Carlo (MCMC) based algorithms, which use sampling from a probability distribution. Basin-hopping is one of the most advanced algorithms in the MCMC class as it combines a stochastic global algorithm with deterministic local searches. Basin hopping belongs to a class of stochastic optimization methods, which unlike many other global optimization algorithms, not only finds a local minimum with ease, but is also able to hop between the local minima. The most significant feature of this method is the allowance for the system to navigate between local minima, in particular, from a lower energy minimum to a higher one, enabling the system to hop among them (Zhan et al., 2004). Therefore it should be well suited for parameter recovery from diffusion model for conflict tasks, which has a solution space with many local minima (as described in section 2). However, basin hopping is quite difficult to implement in production pipelines.

Finally, Bayesian global optimization (Močkus, 1975) is a black box global optimization algorithm that is built upon Bayesian inference and Gaussian processes. Bayesian global optimization is particularly suited to optimization problems where the cost function is high, so that the optimum balance is required between exploration and exploitation of the search space. Bayesian optimization algorithm starts with no assumptions about the solution space, and initiates random sampling of the space. Random sampling has been shown to be more efficient than grid search for parameter optimization by (Bergstra & Bengio, 2012). As the sampling progresses, the posterior is altered as some samples result in a lower loss function value than others, which has an effect on the selection of the subsequent samples. Once the exploration stage is completed, regions of the function which are worth to explore for the optimum solution are identified, at which point the exploitation of the suggested solution space can begin. Bayesian optimization is very well suited to problems where the objective function is costly, such as the diffusion model for conflict tasks, as the calculation of a reaction time distribution of hundreds of trials is a very computationally costly process. Bayesian optimization computes a probability model of the objective function that maps the samples to a probability of a loss, which is easier to optimize than the actual objective function.

It is difficult to predict which one of the five chosen algorithms would return the best parameters for the diffusion model for conflict tasks, because performance of optimization algorithms does not transfer between different optimization problems. This was proposed by Wolpert and Macready (1997) as "No free lunch" theorem, which postulates that for both static and time-dependent optimization problems, the average performance of a single optimization algorithm across all possible problems is identical to another optimization algorithm.

This chapter will investigate how well these five different optimization algorithms perform in recovering parameters from diffusion model for conflict tasks, when considering reaction time distributions with high number of trials, and reaction time distributions with low number of trials. In particular, we are interested in whether any of the five algorithms performs better than the rest, and whether the performance of any of the more advanced algorithms is better than simple random search. The performance of the algorithms will be assessed in the context of constrained computational time: each of the algorithm will be allowed the same time frame (72 hours for distributions with high number of trials and 24 hours for distributions will low number of trials). Three different scenarios will be considered: recovering either seven, six, or five parameters from the diffusion model for conflict tasks, with either zero, one, or two parameters constrained to a fixed value respectively. This will allow investigation of how the performance of the optimization algorithms changes as the dimensionality of the solution space decreases.

3.2 Methods

3.2.1 Distributions to recover parameters from

Twenty distributions were computed with parameters from the diffusion model for conflict tasks, which are displayed in table 3.1. These parameters were obtained from published studies that used the diffusion model for conflict tasks (Ulrich et al., 2015; Servant, White, Montagnini, & Burle, 2016), and from our own lab, obtained from fitting experimental data from the flanker and Simon tasks. Two reaction time distributions were computed for each set of parameters, one with 5000 trials per condition, and another with 200 trials per condition, as the former represents ideal reaction time distributions, while the latter is a realistic dataset from a (relatively long) experiment with human subjects. The two distributions will later be referred to as distributions with large (high) and small (low) number of trials respectively. Three conditions were used for the datasets, congruent, incongruent, and neutral; the only difference between the conditions being the value of the amplitude of automatic activation parameter, with positive value for congruent trials, negative one for incongruent trials, and zero for neutral trials. The additional parameters used for computations were set as 2 for the shape of automatic activation, 4 for the diffusion constant, 700 milliseconds for decision making process (how long the evidence is allowed to accumulate until a decision threshold is reached), and the sampling rate was 10 per millisecond.

The reaction time distributions were then passed individually to the optimization algorithms to recover the parameters of diffusion model for conflict tasks. The optimization was run three times, each time trying to recover different number of parameters: all seven, six parameters (excluding the shape of the starting point distribution), and five (excluding the shape of the starting point distribution and the standard deviation (variability) of non-decision time).

Table 3.1: Values of the parameters from diffusion model for conflict tasks used to compute the reaction time distributions for parameter recovery studies. β_1 - upper boundary, μ_R - non-decision time, σ_R - variability (standard deviation) of non-decision time, μ_C - drift rate for controlled process, A - amplitude of automatic activation, τ - time to peak automatic activation, α - shape of the starting point distribution.

β_1	μ_R	σ_R	μ_C	A	τ	α
75	300	30	0.5	20	30	2
51.3	331.79	36.61	0.69	19.2	118.26	2.15
54.56	322.81	38.58	0.69	15.99	34.94	2.8
62.36	302.55	40.86	0.5	14.7	104.26	2.89
40.05	313.28	40.33	0.77	17.02	192.5	3.02
46.26	287.06	33.89	0.57	16.74	46.9	1.04
39.23	303.45	30.42	0.56	19.27	63.85	3.02
72.13	310.61	22.71	0.77	25.66	111.32	3.58
60.71	321.8	24.21	0.63	27.96	39.62	2.71
48.3	296.38	35.31	0.68	12.65	97.32	2.39
39.86	252.26	44.25	0.59	17.4	131.78	1.06
63.32	349.2	45.12	0.65	22.75	130.34	1.35
54.05	334.99	35.18	0.89	29.37	59.19	2.51
49.01	299.73	35.6	0.74	23.62	155.88	1.14
42.42	328.98	33.31	0.55	30.68	161.43	2.15
61.9	369.28	20.31	0.85	37.15	180.28	3.4
49.44	356.39	29.78	0.67	25.7	111.06	3.15
63.04	303.4	22.1	0.79	26.75	119.69	2.56
52.68	282.9	32.15	0.62	25.99	247.01	1.06
37.18	287.46	32.56	1.06	16.55	125.85	0.73

3.2.2 The search space

In order to implement all of the global optimization algorithms in this chapter, the search space of the solution had to be defined. Previous papers that investigated parameter recovery from sequential sampling models looked at the same search space boundaries as they did to compute the reaction time distributions (White et al., 2018). This approach makes parameter recovery unnaturally easy, especially if the boundaries chosen for the parameters are relatively constrained. Moreover, this approach could hinder the recovery of parameters from distributions that might have more extreme parameter values, such as when investigating the performance in clinical samples or in individual differences research with more extreme behaviour. Another approach is to use a much larger search space and see whether the optimization algorithms are capable of finding the correct parameters. This approach not only allows investigating the true power of the optimization algorithms, but also does not unnecessarily exclude the application of the decision making models from experimental studies with populations that may display extreme values for some of the model parameters. For these two reasons, the search space for global optimization algorithms was constrained with parameter values shown in table 3.2.

Table 3.2: Limits for each of the parameter from diffusion model for conflict tasks to constrain the search space.

Parameter	Parameter abbreviation	Lower limit	Upper limit
Upper boundary	β_1	20	120
Non-decision time	μ_R	150	500
Variability of non-decision time	σ_R	10	60
Drift rate for controlled process	μ_C	0.0001	2
Amplitude of automatic activation	A	0.0001	70
Time to peak automatic activation	τ	10	300
Shape of starting point distribution	α	0.0001	4

3.2.3 Algorithm implementation

Every algorithm was implemented using PYTHON programming language, with the extension of CYTHON to speed up the computation of reaction time distributions from the diffusion model for conflict tasks. Random search was implemented manually, differential evolution and basin-hopping were implemented with native SCIPY package (Jones, Oliphant, Peterson, et al., 2001–). Bayesian optimization was implemented with the Bayesian Optimization package (`https://github.com/fmfn/BayesianOptimization`) [2]. Particle swarm optimization was implemented with the PYSWARM package (`https://pythonhosted.org/pyswarm/`).

Each optimization algorithm selected a potential solution to the problem, which was a set of parameter values selected from the search space. This set of parameter values was then used to compute a reaction time distribution from diffusion model for conflict tasks, with congruent, incongruent, and neutral trials. The number of trials per condition depended

[2]Clickable links in PDF

on the size of the distribution that needed parameters recovered. For distributions with large number of trials, the optimization algorithms computed reaction time distributions with 5000 trials per condition, while for the distributions with small number of trials, the optimization algorithms computed 1000 trials per condition. These numbers were chosen considering the findings from section 2. The distributions with large number of trials were matched in size to emulate ideal conditions for parameter recovery in the level of noise in trials. The number of trials for the distributions with small number of trials was increased to 1000 trials per condition, as it offered a good balance between computational time and level of noise in the distributions. The goodness of the solution was evaluated by comparing the computed reaction time distribution with the problem reaction time distribution. The comparison was performed individually for congruent, incongruent, and neutral trials, using Kolmogorov-Smirnov (KS) test for comparing distributions. The returned KS values were then added together to a single sum value, which was the objective function to minimize (as KS values closer to zero denote distributions that look more alike).

The random search algorithm simply chose a set of parameters randomly from a predefined range, computed a reaction time distribution with those parameters, and then compared the resulting distribution with the initial one. If the value of the objective function was lower than the one from the previous best set of parameters, the previous best set of parameters was replaced with the current ones, otherwise, the previous best set of parameters remained unchanged. This process was iterated 500 times for distributions with large number of trials, and 3500 times for distributions with small number of samples. These values were chosen so that the algorithms would finish in 72 hours and 24 hours respectively, with 72 hours being the upper limit for computations set by the High Performance Computing clusters in Cardiff University. The 24 hour limit for the distributions with small number of trials was chosen to reflect the reduced computational time requirements resulting from computing smaller reaction time distributions.

Unlike the other optimization algorithms used in this chapter that look for a minimum of a function, the Bayesian optimization algorithm, as implemented in the Bayesian Optimization package, is a global maximizer. For this reason, the objective function was inverted, so instead of trying to find the smallest sum of Kolmogorov-Smirnov (KS) values for congruent, incongruent, and neutral conditions, the algorithm tried to find the largest value, defined as $3-$ sum of KS values. Formulated in this way, the algorithm was still trying to minimize the sum of KS values. For distributions with large number of trials, the number of exploratory iterations was 1,500, while the number of exploitatory iterations was 1,500. For distributions with low number of trials, both the exploratory and the exploitatory iterations were set to 750 each. The number of iterations was chosen for the same reasons as described above, to allow the algorithm to complete in 72 hours and 24 hours respectively.

The population size for the differential evolution algorithm was set to 10. The number of iterations for recovery of parameters from distributions with large number of trials was set to 50, while for distributions with low number of trials it was set to 200, again to allow the algorithm to finish within 72 and 24 hours respectively. Similarly, for the particle swarm optimization algorithm, the population size was set to 10, while the number of iterations were set to 350 and 1000 for distributions with high and low number of trials respectively.

Parameters from each distribution were only recovered once by each optimization algorithm. This was done for two reasons. First reason was the time constraint: as each recovery algorithm already took so long, running it multiple times would have increased the runtime costs linearly. Secondly, even though the runtime can be optimized by choosing the best starting points for some algorithms, this strategy is difficult to implement in population-based algorithms, which would make comparison between different algorithms difficult.

3.3 Results

3.3.1 Comparison of the global optimization algorithms

The results from four out of five global optimization algorithms were compared. The basin hopping algorithm was excluded from the comparisons due to sub-par performance, as described in the detailed results on each individual algorithm.

In order to see whether there was a difference between algorithms in the objective function value of recovered parameters, a two-way ANOVA was performed, with the number of recovered parameters and the optimization algorithm as independent within-subjects variables, for the data with 5,000 trials per condition. The ANOVA results suggest that there was no interaction between the number of recovered parameters and the optimization algorithm ($F(4,76) = 1.44, p = 0.230$). Further investigation into the main effects revealed that the number of recovered parameters did not have an effect on the objective function values ($F(2.38) = 2.23, p = 0.121$), while the type of the optimization algorithm did ($F(2,38) = 31.71$, p=7.9×10^{-9}). This difference comes from the differential evolution performing better than the other algorithms. These results are shown in figure 3.1.

When the results from the distributions with low number of trials were investigated, similar pattern emerged: there was no interaction between the number of recovered parameters and the optimization algorithm used ($F(4,76) = 0.25, p = 0.910$). When the main effects where investigated, once again the number of recovered parameters had no effect on the objective function values: $F(2,38) = 0.11, p = 0.893$, while the optimization algorithm had a significant effect: $F(2,38) = 66.41$, p=4.0×10^{-13}. This difference came from the differential evolution algorithm producing lower objective function values (better performance of the algorithm) than the remaining three algorithms.

These results indicate that irrespective of the size of the distribution and the number of recovered parameters, differential evolution results in the best parameter recovery from the diffusion model for conflict tasks.

Another way to compare the algorithms is to evaluate whether any of the algorithms resulted in correlation coefficients between actual and recovered parameters that were consistently higher for all of the model parameters compared to other optimization algorithms. This is different than looking at objective function values, because a small objective function value could result from good recovery of some parameters while other parameters were not recovered well. Inspecting correlation coefficients for each parameter separately allows to investigate whether any algorithm recovered all model parameters better than other algorithms. For this purpose, each model parameter was treated as a sample, the optimization algorithms as independent variable, and the correlation coefficient as dependent variable. Kruskal-Wallis test was performed to see whether the difference in the average correlation coefficient was statistically significant, which is a non-parametric equivalent of ANOVA. This was done due to the violation of assumptions underlying ANOVA. The results of comparison are shown in figure 3.2. Even though it appears that differential evolution provided the highest correlation coefficients with the least amount of variance, the Kruskal-Wallis test performed on this data showed that there was no significant difference in the correlation

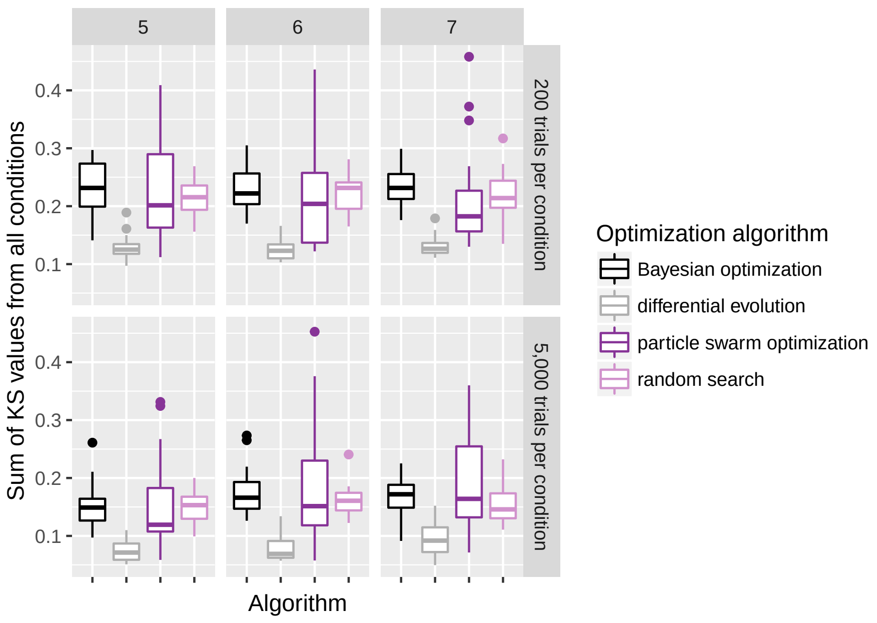


Figure 3.1: Comparison of KS values from recovery of parameters by different optimization algorithms for different number of recovered parameters (columns) for both small (top row) and large (bottom row) reaction time distributions. Dots indicate outliers.

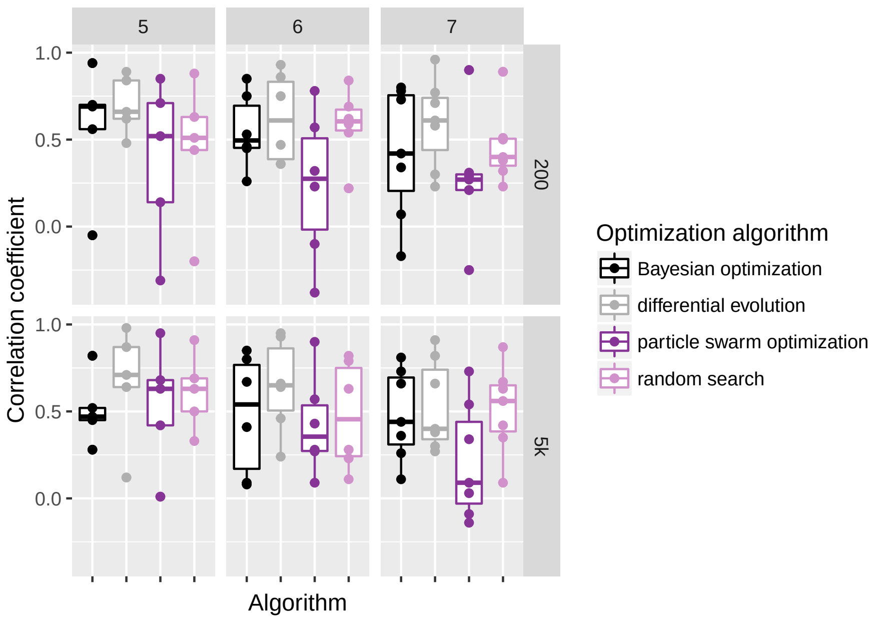


Figure 3.2: Comparison of correlation coefficients for all parameters from four optimization algorithms. Dots indicate the full distribution of data points.

coefficients between the four different optimization algorithms for any combination of the number of recovered parameters and the number of trials in the distributions (all $\chi^2(3) < 4.7, p > 0.19$). This could be the result of a very small "sample size" of five to seven correlations.

3.3.2 Random search

When data was limited to 200 trials per condition, and when trying to recover parameters with distributions with 1,000 trials per condition, the recovered parameters look like displayed in figure 3.4. The Pearson correlation coefficients between actual and recovered parameter values are shown in table table 3.3.

Table 3.3: Correlation coefficients (p values) between actual parameter values and recovered parameter values using the random search algorithm. Dashes indicate the parameters that were not recovered in that condition. Values displayed in bold indicate statistically significant correlations (not adjusted for multiple comparisons).

	5,000 trials		
Parameters	5	6	7
β_1	**0.69** (7.1×10^{-4})	**0.79** (3.7×10^{-5})	**0.67** (0.001))
μ_R	**0.91** (3.1×10^{-8})	**0.82** (8.9×10^{-6})	**0.87** (7.7×10^{-7})
σ_R	-	0.28 (0.230)	**0.63** (0.003)
μ_C	**0.63** (0.003)	0.23 (0.323)	0.35 (0.133)
A	**0.50** (0.024)	**0.63** (0.003)	0.42 (0.063)
τ	0.33 (0.152)	0.11 (0.635)	0.09 (0.694)
α	-	-	**0.56** (0.010)
	200 trials		
	5	6	7
β_1	**0.63** (0.003)	**0.69** (8.4×10^{-4})	**0.51** (0.021)
μ_R	**0.88** (3.5×10^{-7})	**0.84** (3.0×10^{-6})	**0.89** (1.7×10^{-7})
σ_R	-	**0.62** (0.004)	0.40 (0.083)
μ_C	**0.51** (0.021)	**0.54** (0.013)	0.23 (0.325)
A	0.44 (0.050)	**0.59** (0.007)	0.32 (0.166)
τ	-0.20 (0.386)	0.22 (0.357)	0.38 (0.103)
α	-	-	**0.50** (0.025)

The table 3.3 shows that the random search algorithm managed to recover some parameters of the diffusion model for conflict tasks reasonably well in all conditions (upper boundary, non-decision time), while others were recovered less well, as the strength and significance of the correlations for other parameters varied between conditions. Time to peak automatic activation was never recovered appropriately. The correlations between recovered parameters from distributions with large number of trials is shown in figure 3.3. The differences between actual and recovered parameter values are shown in figure 3.6 and figure 3.5 respectively.

Recovering different number of model parameters made no difference to the objective function values for distributions with large number of trials (5,000 trials per condition): F(2,38)=0.0009, p=0.997; and for distributions with low number of trials (200 trials per condition): F(2,38) = 0.02, p = 0.895. This result implies that when using the random search algorithm, focusing on fewer model parameters does not seem to provide any benefit in the recovery of the remaining model parameters.

3.3.3 Differential evolution

Differential evolution performed very well in recovering upper boundary and non-decision time parameters of diffusion model for conflict tasks in all conditions, returning high and significant correlations, which are shown in table 3.4. The algorithm also recovered time to peak automatic activation and the variability of non-decision time well from distributions with low trial numbers. All parameters were recovered well when two parameters were fixed in the distributions with low trial numbers. The correlations between actual parameter values and recovered parameter values are displayed in figure 3.7 for distributions with high number of trials and figure 3.8 for distributions with low number of trials. The differences between actual and recovered parameter values are shown in figure 3.9 and figure 3.10 respectively.

Recovering different number of model parameters with differential evolution algorithm from distributions with large number of trials affected the objective function values: F(2,38)=13.62, p=0.002. The difference in objective function values resulted from the recovery of seven parameters being worse than the recovery of either five or six model parameters (the difference between five and six model parameters: t(19) = -0.32, p=0.750, the difference between five and seven model parameters: t(19) = -3.69, p=0.002, the difference between six and seven model parameters: t(19) = -3.26, p=0.004; p values were not adjusted for multiple comparisons). This result implies that for distributions with high number of trials, recovering fewer parameters with differential evolution results in

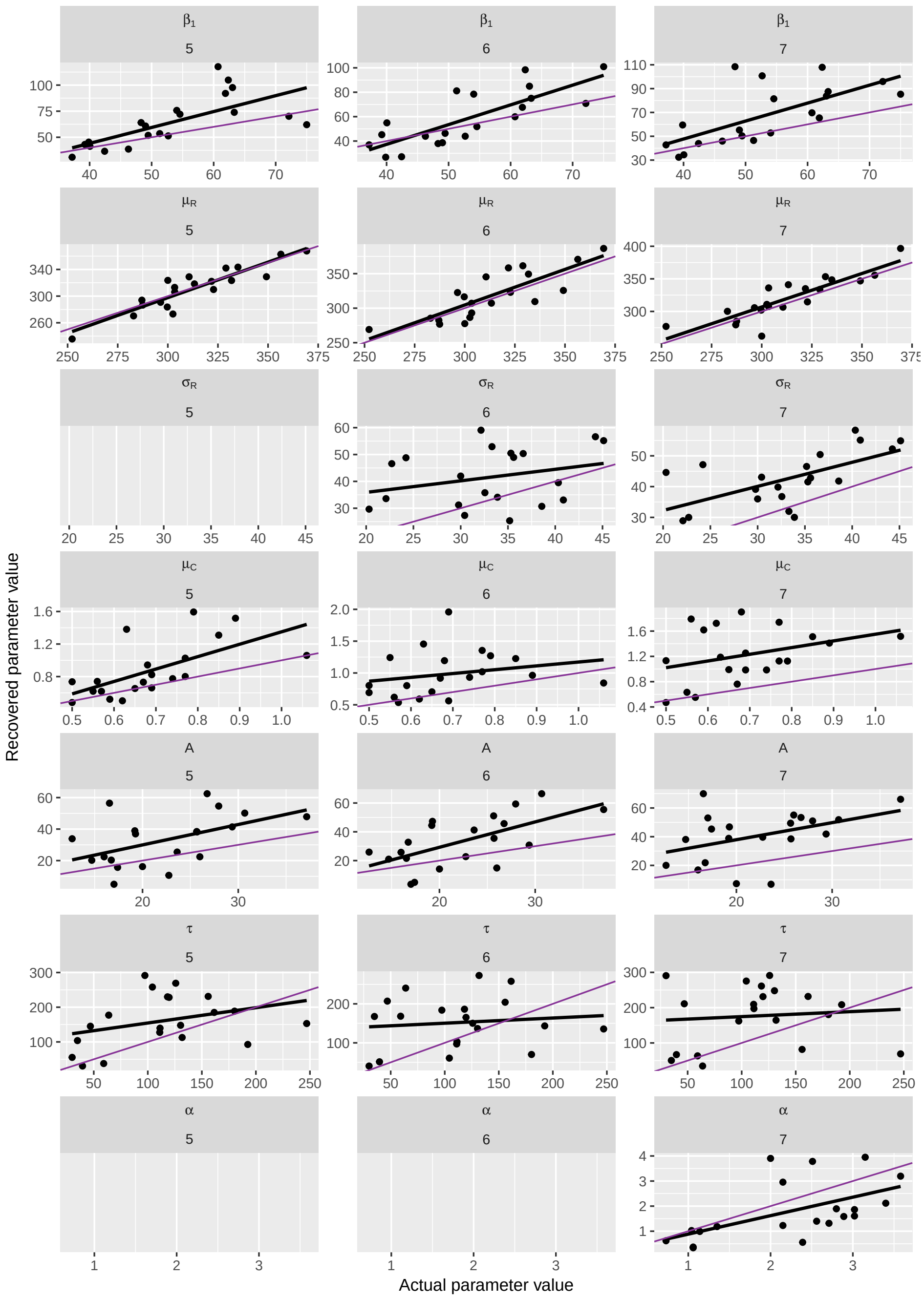


Figure 3.3: Correlations between actual (x axis) and recovered (y axis) parameter values using the random search algorithm from distributions with large number of trials. Purple line indicates identity line, while the black line is the correlation between recovered and actual parameters.

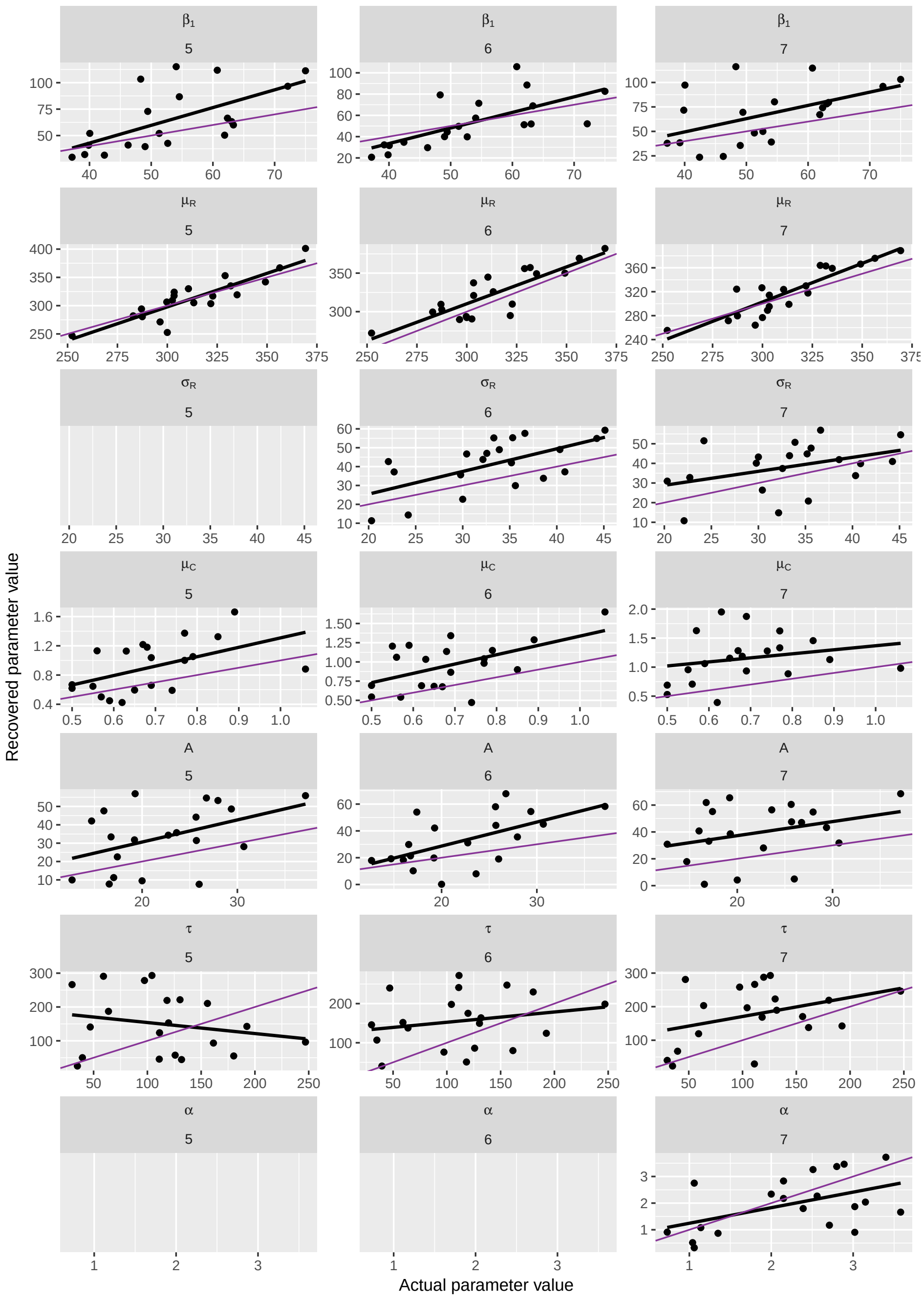


Figure 3.4: Correlations between actual (x axis) and recovered (y axis) parameter values using the random search algorithm from distributions with small number of trials. Purple line indicates identity line, while the black line is the correlation between recovered and actual parameters.

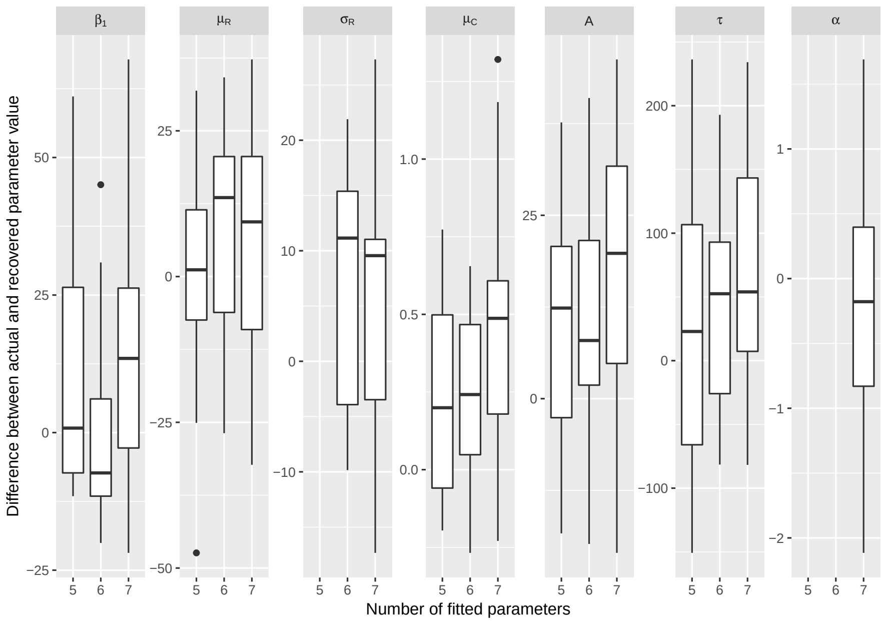


Figure 3.5: Difference between actual and recovered parameter values using the random search algorithm from distributions with small number of trials. Dots indicate outliers.

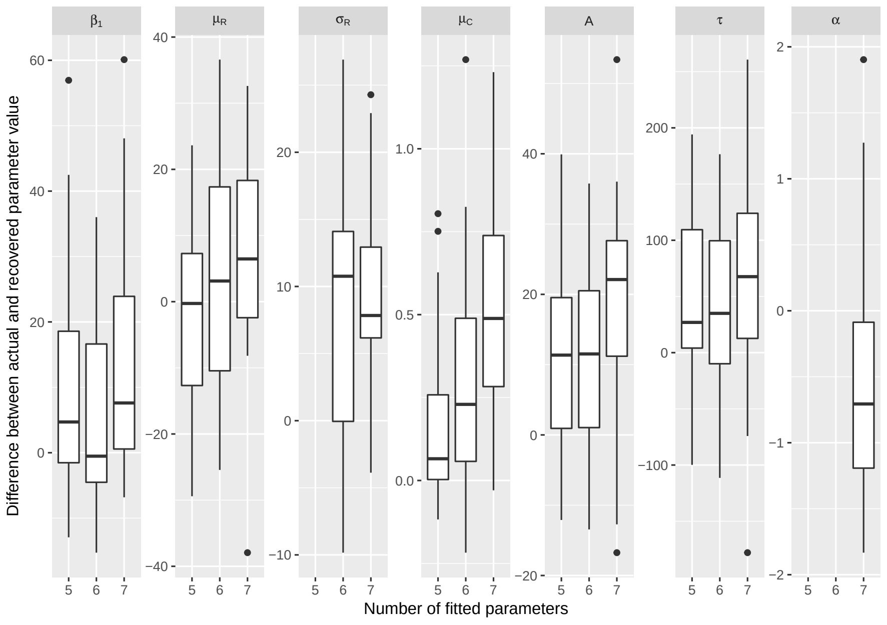


Figure 3.6: Difference between actual and recovered parameter values using the random search algorithm from distributions with large number of trials. Dots indicate outliers.

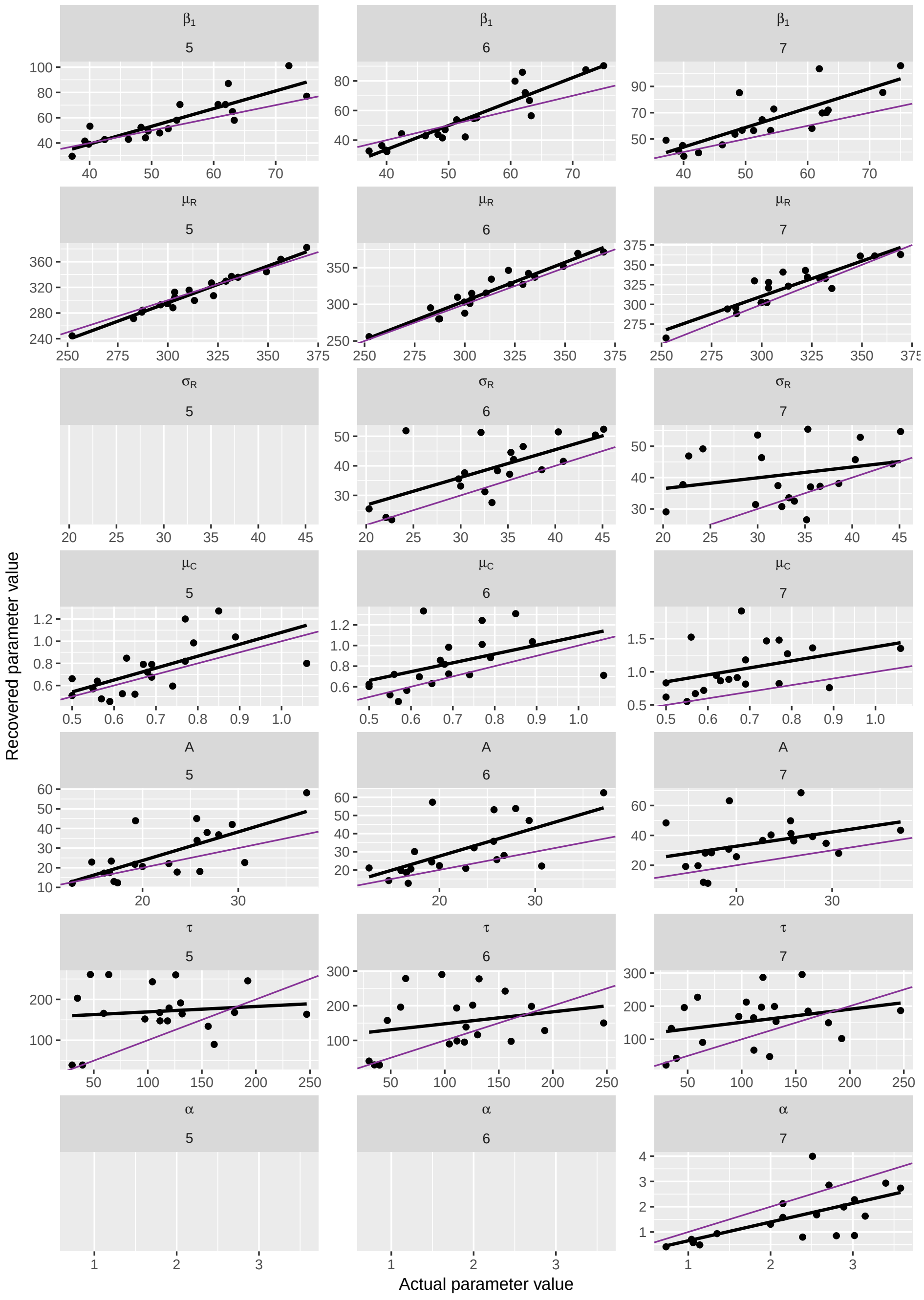


Figure 3.7: Correlations between actual and recovered parameter values using the differential evolution algorithm from large distributions. Purple line indicates identity line, while black line is the correlation between recovered and actual parameters.

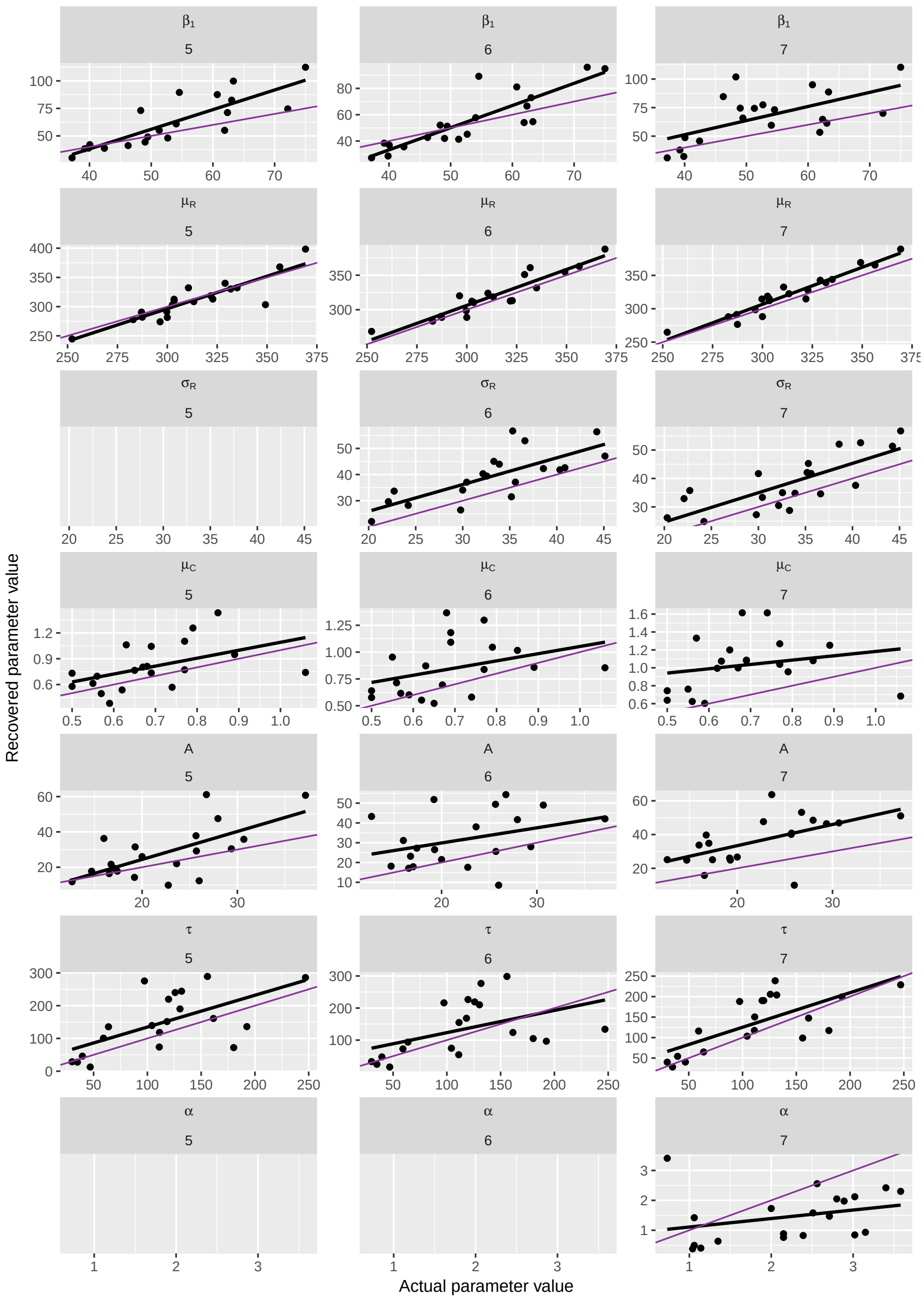


Figure 3.8: Correlations between actual and recovered parameter values for differential evolution of small distributions. Purple line indicates identity line, while black line is the correlation between recovered and actual parameters.

better recovered parameter values.

This was not the case for distributions with low trial numbers, as there was no difference in the objective function values when trying to recover five, six, or seven model parameters: F(2,38) = 0.22, p=0.645.

Table 3.4: Correlation coefficients (p values) between actual parameter values and recovered parameter values using the differential evolution algorithm. Dashes indicate the parameters that were not recovered in that condition. Values displayed in bold indicate statistically significant correlations (not adjusted for multiple comparisons).

	5,000 trials per condition		
Parameters	5	6	7
β_1	**0.87** (7.1×10^{-7})	**0.93** (4.5×10^{-9})	**0.82** (1.1×10^{-5})
μ_R	**0.98** (2.0×10^{-13})	**0.95** (6.2×10^{-11})	**0.91** (4.0×10^{-8})
σ_R	-	**0.66** (0.001)	0.27 (0.259)
μ_C	**0.64** (0.002)	**0.46** (0.039)	0.40 (0.082)
A	**0.71** (4.3×10^{-4})	**0.64** (0.002)	0.38 (0.093)
τ	0.12 (0.625)	0.24 (0.300)	0.30 (0.202)
α	-	-	**0.66** (0.002)
	200 trials per condition		
	5	6	7
β_1	**0.84** (4.6×10^{-6})	**0.86** (8.9×10^{-7})	**0.61** (0.004)
μ_R	**0.89** (1.6×10^{-7})	**0.93** (2.3×10^{-9})	**0.96** (2.4×10^{-11})
σ_R	-	**0.75** (1.3×10^{-4})	**0.77** (7.1×10^{-5})
μ_C	**0.48** (0.031)	0.36 (0.115)	0.23 (0.339)
A	**0.66** (0.001)	0.36 (0.123)	**0.58** (0.008)
τ	**0.62** (0.003)	**0.47** (0.039)	**0.71** (4.0×10^{-4})
α	-	-	0.30 (0.201)

3.3.4 Particle swarm optimization

The particle swarm optimization algorithm recovered the non-decision time parameter well from all conditions, as evaluated by the correlations between the actual and recovered parameter values table 3.5. Otherwise, the algorithm struggled to recover the model parameters well, as shown in figure 3.11 for distributions with high trial numbers and figure 3.12 for distributions with low trial numbers. The differences between recovered and actual parameter values are displayed in figure 3.13 and figure 3.14 re-

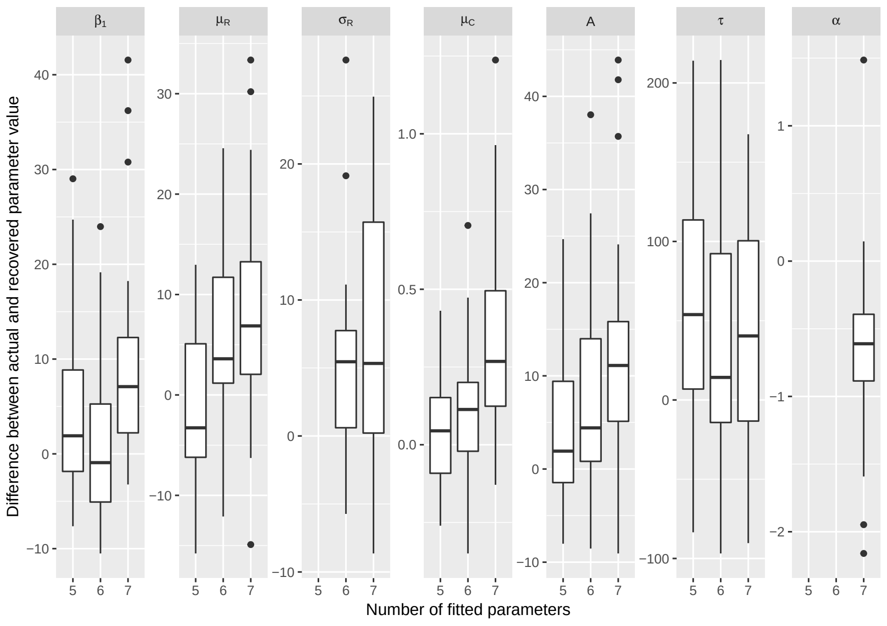


Figure 3.9: Difference between actual and recovered parameter values using the differential evolution algorithms from large distributions. Dots indicate outliers.

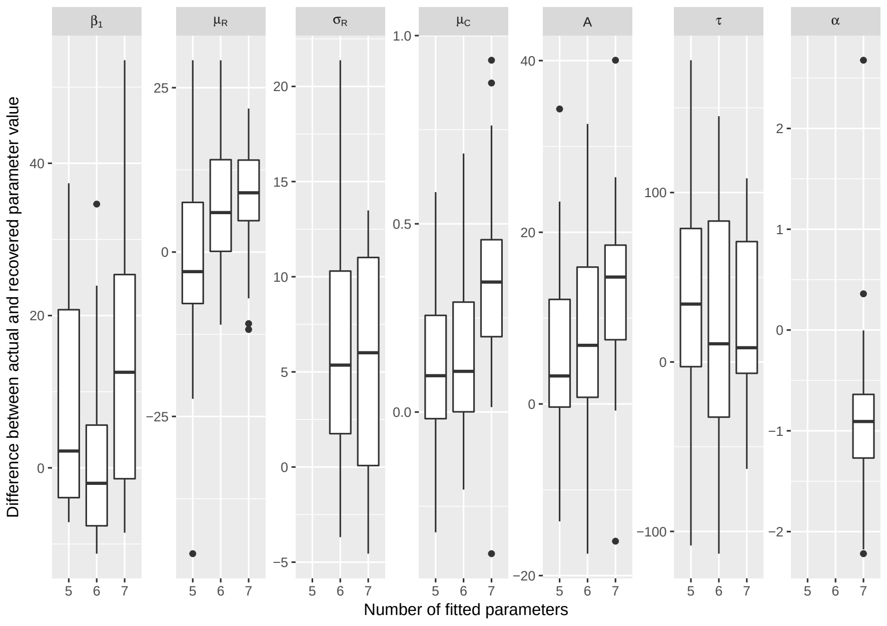


Figure 3.10: Difference between actual and recovered parameter values using the differential evolution algorithms from small distributions. Dots indicate outliers.

spectively.

There was a difference between recovering different numbers of parameters for distributions with large number of trials, but it did not reach significance: F(2,38) = 3.05, p=0.097. This difference resulted from recovery of five parameters being slightly better than recovery of six or seven parameters: the difference between six and seven parameters: t(19)=0.05, p=0.959, the difference between five and seven parameters: t(19)=-1.75, p=0.097, the difference between five and six parameters: t(19)=-1.46, p=0.161.

For small distributions, there was no difference between recovering different number of model parameters: F(2,38)=0.68, p=0.419.

Table 3.5: Correlation coefficients (p values) between actual parameter values and recovered parameter values from the particle swarm optimization algorithm. Dashes indicate the parameters that were not recovered in that condition. Values displayed in bold indicate statistically significant correlations (not adjusted for multiple comparisons).

	5,000 trials per condition		
Parameters	5	6	7
β_1	**0.63** (0.003)	0.28 (0.228)	0.09 (0.712)
μ_R	**0.95** (3.4×10^{-10})	**0.90** (7.7×10^{-8})	**0.73** (2.4×10^{-4})
σ_R	-	0.27 (0.258)	-0.14 (0.544)
μ_C	0.42 (0.065)	**0.57** (0.009)	**0.54** (0.015)
A	**0.68** (0.001)	0.43 (0.060)	-0.09 (0.706)
τ	0.01 (0.960)	0.09 (0.700)	0.03 (0.904)
α	-	-	0.34 (0.148)
	200 trials per condition		
Parameters	5	6	7
β_1	**0.52** (0.020)	**0.57** (0.009)	0.29 (0.208)
μ_R	**0.85** (2.3×10^{-6})	**0.78** (4.8×10^{-5})	**0.90** (4.3×10^{-8})
σ_R	-	0.23 (0.329)	0.31 (0.182)
μ_C	**0.71** (4.6×10^{-4})	-0.10 (0.687)	0.27 (0.241)
A	0.14 (0.570)	0.32 (0.174)	0.21 (0.369)
τ	-0.31 (0.191)	-0.38 (0.103)	0.21 (0.368)
α	-	-	-0.25 (0.292)

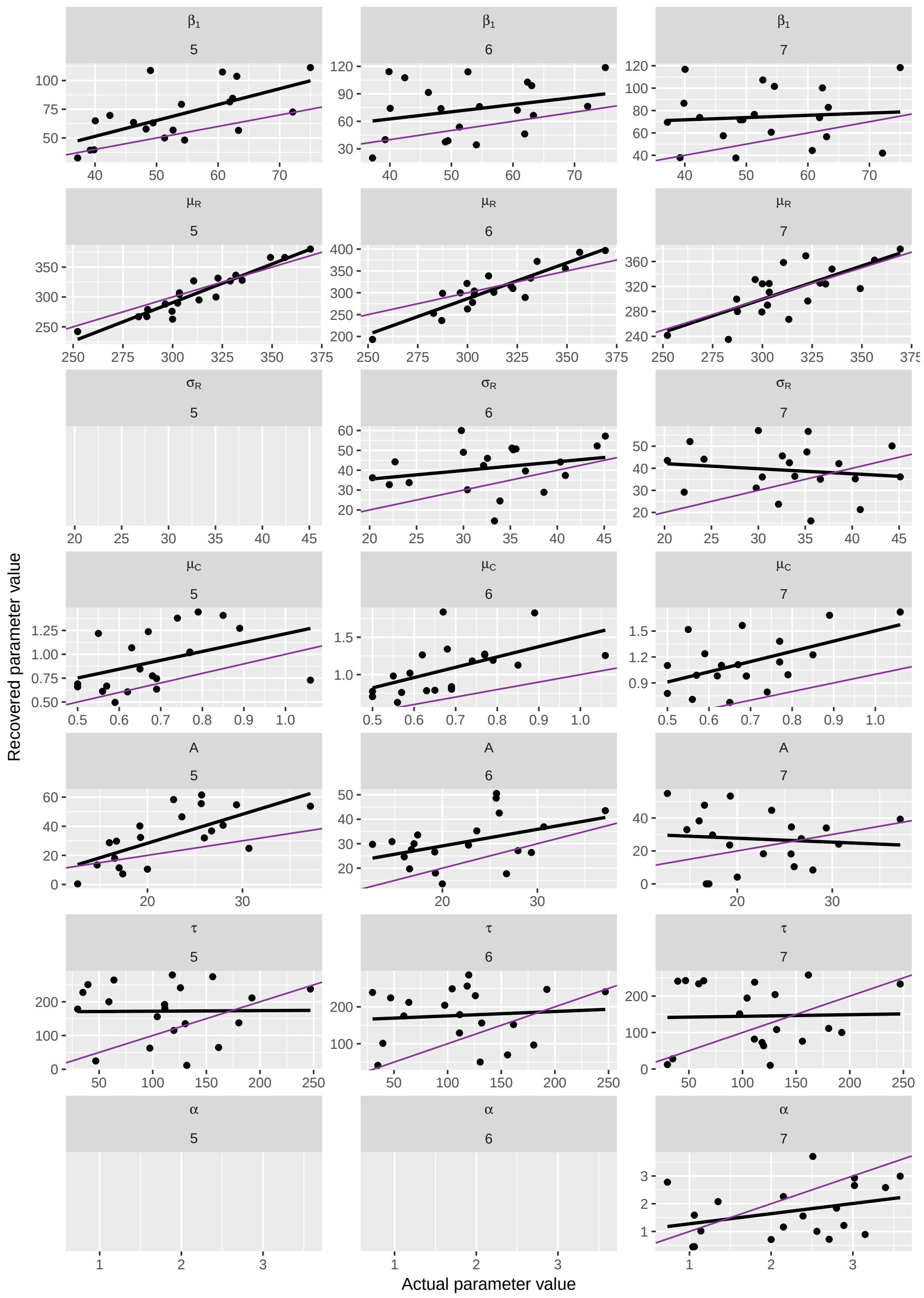


Figure 3.11: Correlations between actual and recovered parameter values for particle swarm optimization of large distributions. Purple line indicates identity line, while black line is the correlation between recovered and actual parameters.

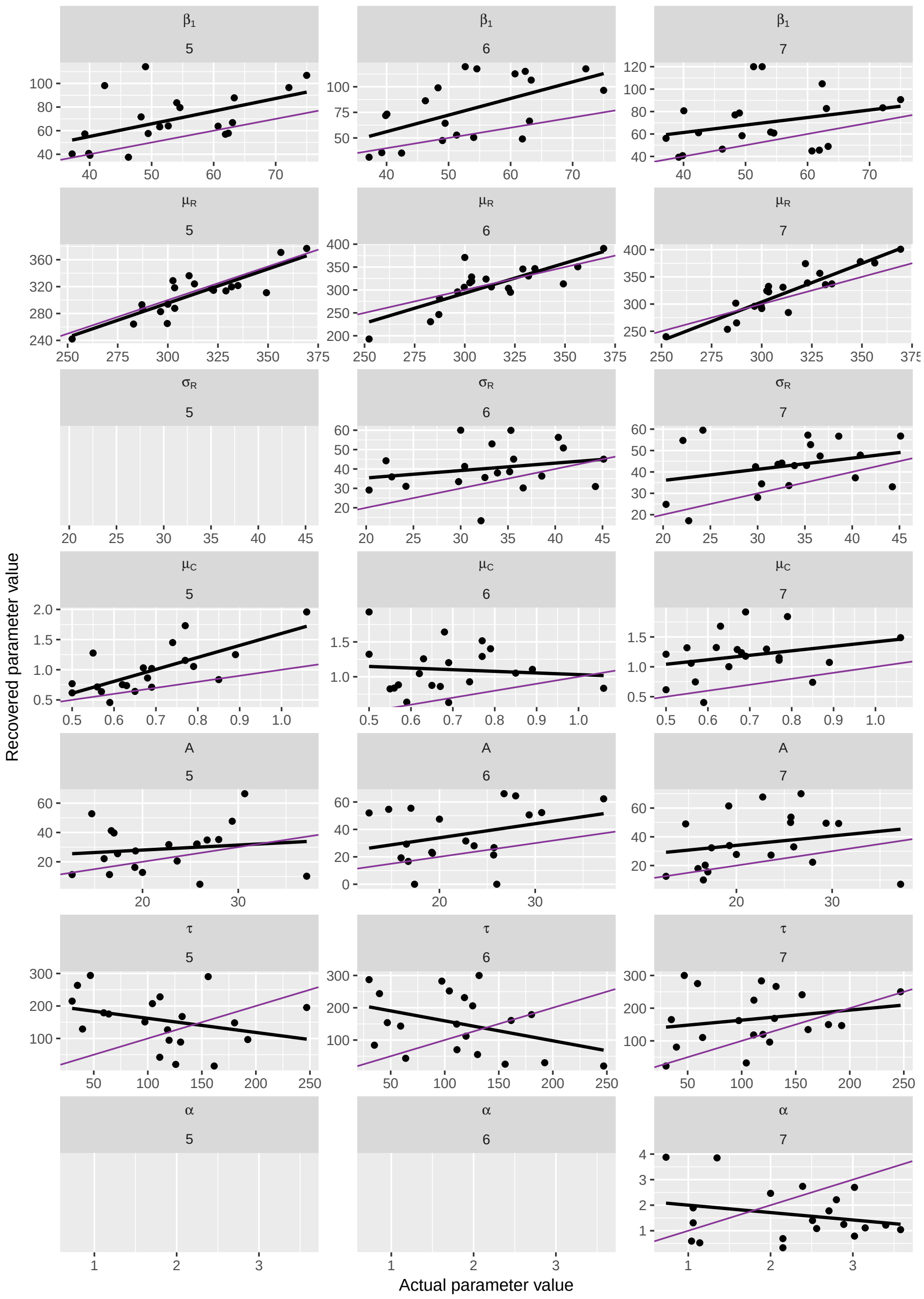


Figure 3.12: Correlations between actual and recovered parameter values for particle swarm optimization of small distributions. Purple line indicates identity line, while black line is the correlation between recovered and actual parameters.

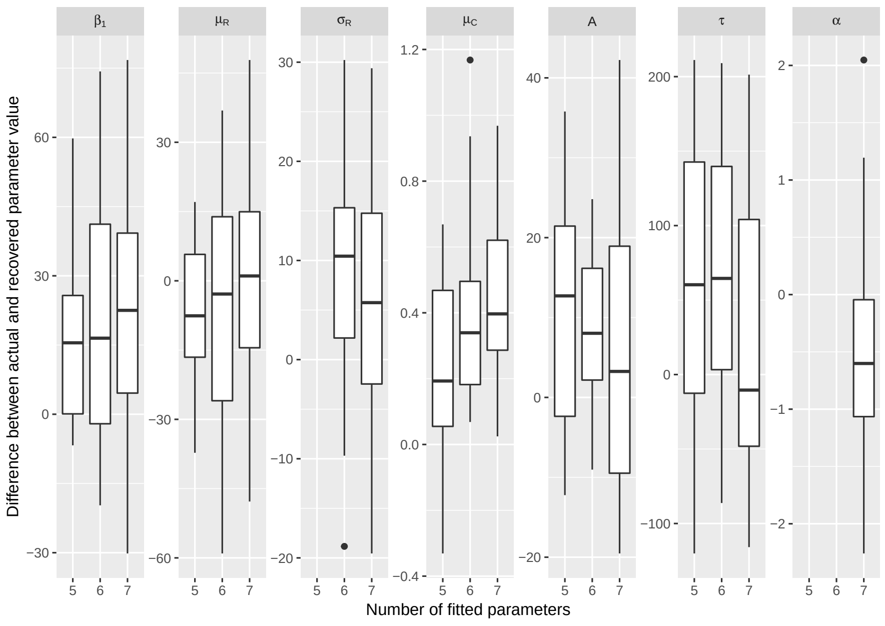


Figure 3.13: Difference between actual and recovered parameter values for particle swarm optimization of large distributions. Dots indicate outliers.

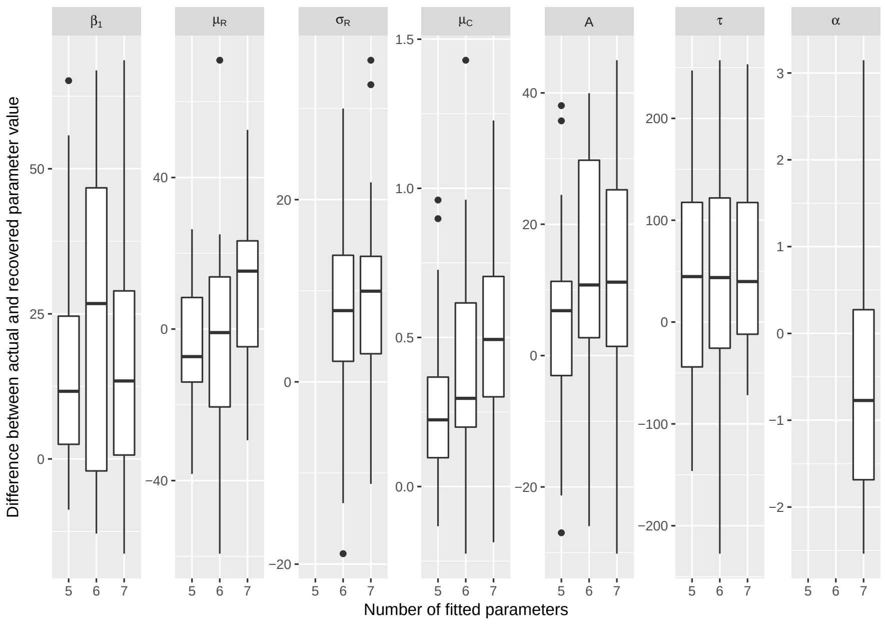


Figure 3.14: Difference between actual and recovered parameter values for particle swarm optimization of small distributions. Dots indicate outliers.

3.3.5 Basin hopping

Basin hopping appeared to be extremely unsuitable for parameter recovery from diffusion model for conflict tasks due to the very long algorithm computation time. As the algorithm managed only 15 iterations over 72 hour period for distributions with high number of trials, the recovery of parameters was very poor, as evident by the very high objective function values (average sum of KS values for distributions with high trial number: 1.29; compare to values in figure 3.1). The correlations between actual parameter values and recovered parameter values are displayed in table 3.6, and illustrated in figure 3.15. As is evident from the table, not a single correlation for any parameter reached significance, irrespective of the number of fitted parameters. There was no difference between fitting five, six, or seven parameters: F(2,38) = 0.62, p=0.433.

Table 3.6: Correlation coefficients (p values) between actual parameter values and recovered parameter values using the basin hopping algorithm. Dashes indicate the parameters that were not recovered in that condition. Values displayed in bold indicate statistically significant correlations (not adjusted for multiple comparisons).

	5,000 trials per condition		
Parameters	5	6	7
β_1	0.12 (0.616)	0.19 (0.432)	-0.13 (0.588)
μ_R	0.15 (0.520)	-0.13 (0.587)	-0.08 (0.752)
σ_R	-	-0.05 (0.840)	-0.04 (0.875)
μ_C	0.06 (0.816)	0.03 (0.891)	0.40 (0.081)
A	0.08 (0.747)	-0.35 (0.129)	0.04 (0.868)
τ	0.09 (0.713)	-0.41 (0.072)	0.03 (0.885)
α	-	-	0.37 (0.104)
	200 trials per condition		
Parameters	5	6	7
β_1	**0.52** (0.020)	**0.57** (0.009)	0.29 (0.208)
μ_R	**0.85** (2.3×10^{-6})	**0.78** (4.8×10^{-5})	**0.90** (4.3×10^{-8})
σ_R	-	0.23 (0.329)	0.31 (0.182)
μ_C	**0.71** (4.6×10^{-4})	-0.10 (0.687)	0.27 (0.241)
A	0.14 (0.570)	0.32 (0.174)	0.21 (0.369)
τ	-0.31 (0.191)	-0.38 (0.103)	0.21 (0.368)
α	-	-	-0.25 (0.292)

Similar results were observed with the parameter recovery from distributions with low trial numbers with the basin hopping algorithm. Even though more function evaluations

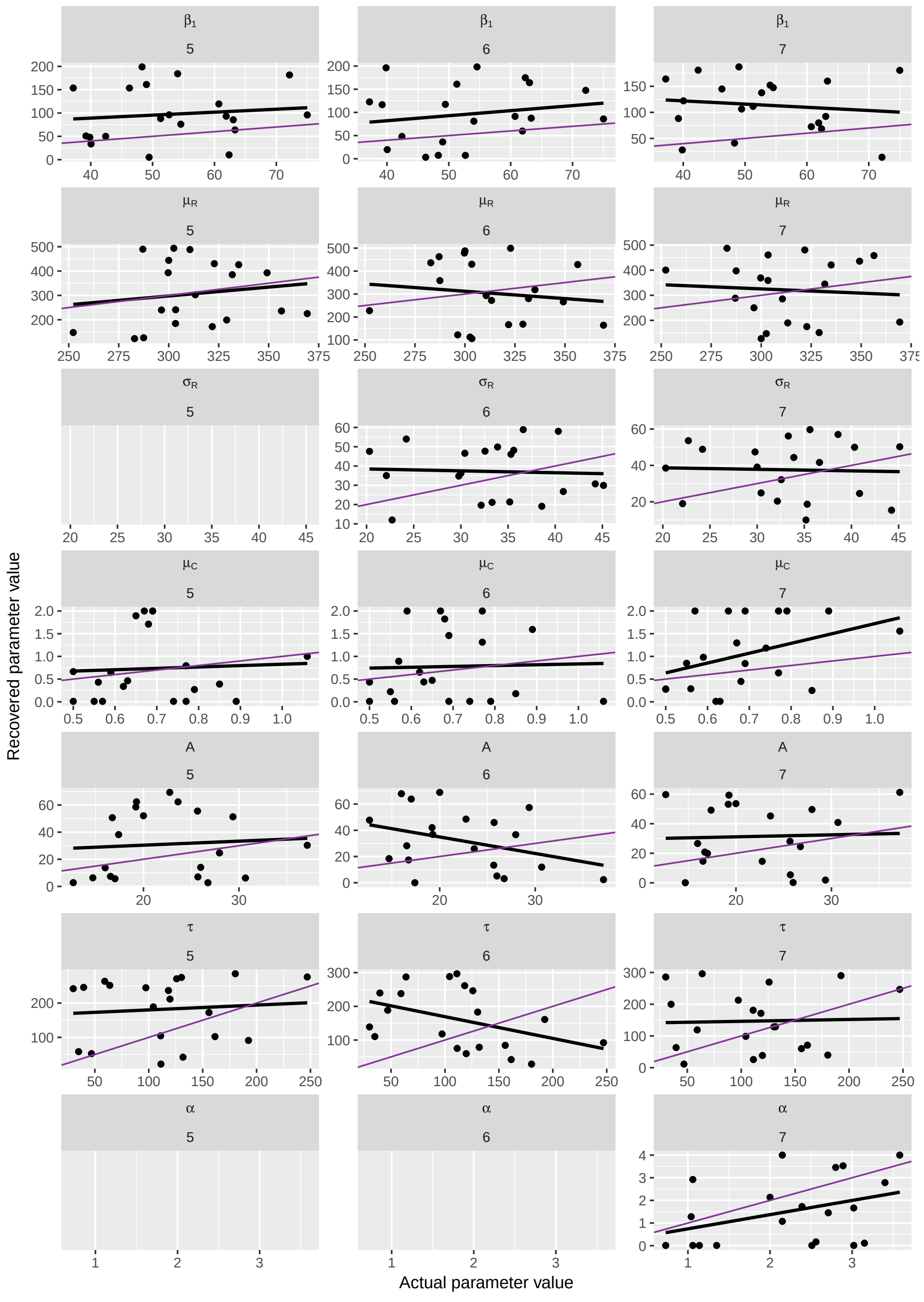


Figure 3.15: Correlations between actual and recovered parameter values for basin hopping of large distributions. Purple line indicates identity line, while black line is the correlation between recovered and actual parameters.

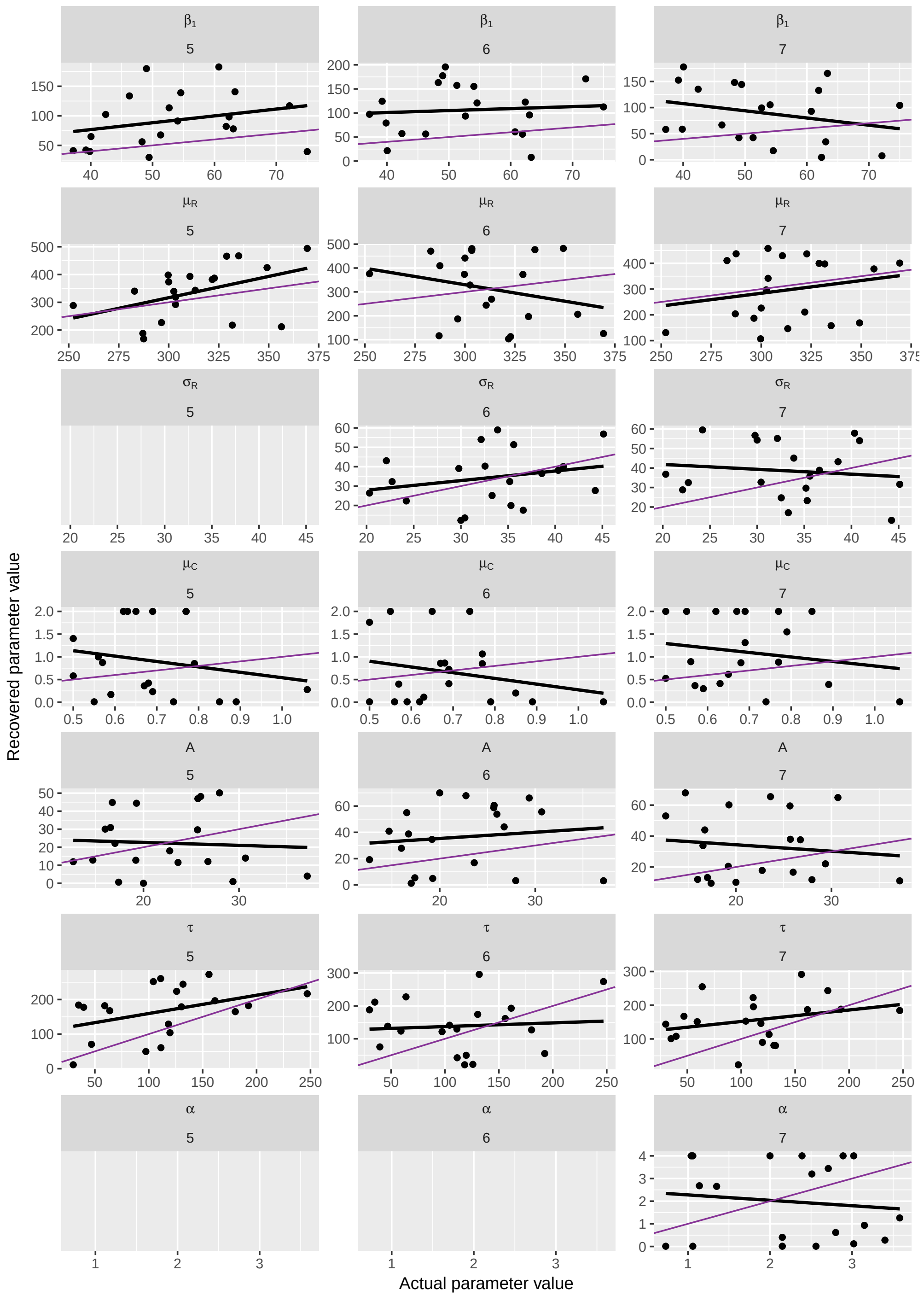


Figure 3.16: Correlations between actual and recovered parameter values for basin hopping of small distributions. Purple line indicates identity line, while black line is the correlation between recovered and actual parameters.

were possible (150 were made within 24 hour window), the algorithm still struggled to achieve objective function values low enough to be comparable to other global optimization algorithms (average sum of KS values for distributions with low trial number: 1.16; compare with KS values in figure 3.1). Some parameters, like the non-decision time, were recovered well by the basin hopping algorithm (reported in table 3.6 and illustrated in figure 3.16). There was a difference between recovering different number of parameters from distributions with low trial numbers: F(2,38) = 3.57, p = 0.04. This difference came from the recovery of seven parameters being worse than the recovery of six or five parameters: the difference between five and seven parameters: t(19) = 1.75, p = 0.10, the difference between five and six parameters: t(19) = -0.97, p = 0.34, the difference between six and seven parameters: t(19) = -2.70, p = 0.01, p values not adjusted for multiple comparisons.

3.3.6 Bayesian optimization

The Bayesian optimization algorithm recovered the upper boundary and the non-decision time parameters well in all conditions, as displayed in table 3.7. The recovery of other parameters depended on condition. The correlations between actual and recovered parameter values are shown in figure 3.17 for distributions with high trial number, while for distributions with low trial number the correlations are shown in figure 3.18. The differences between actual and recovered parameter values are shown in figure 3.20 and figure 3.19 respectively.

There was no difference between recovering seven, six, or five parameters from distributions with small number of trials: F(2,38) = 0.07, p=0.94. There was also no difference in objective function values between recovering different number of parameters from distributions with high number of trials: F(2,38) = 3.12, p = 0.06.

3.4 Discussion

In this chapter, five global black-box optimization algorithms covering different families of global optimization methods were used to recovered parameters from twenty reaction time distributions obtained from diffusion model for conflict tasks. The distributions were computed with large (5000 per condition) and small (200 per condition) number of trials, and the number of parameters recovered from the distributions were either seven

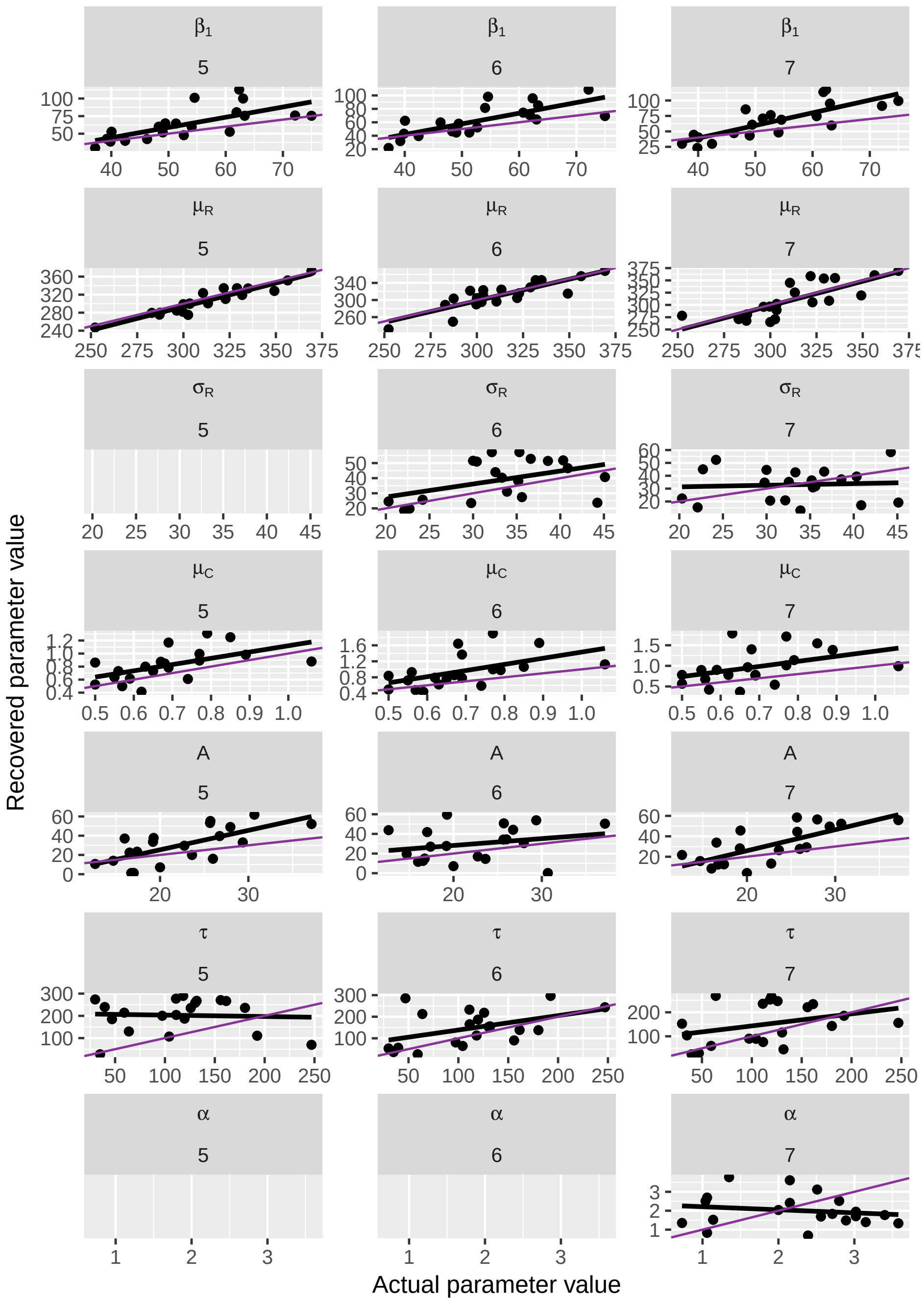


Figure 3.17: Correlations between actual and recovered parameter values using the Bayesian optimization algorithm from distributions with large number of trials. Purple line indicates identity line, while black line is the correlation between recovered and actual parameters.

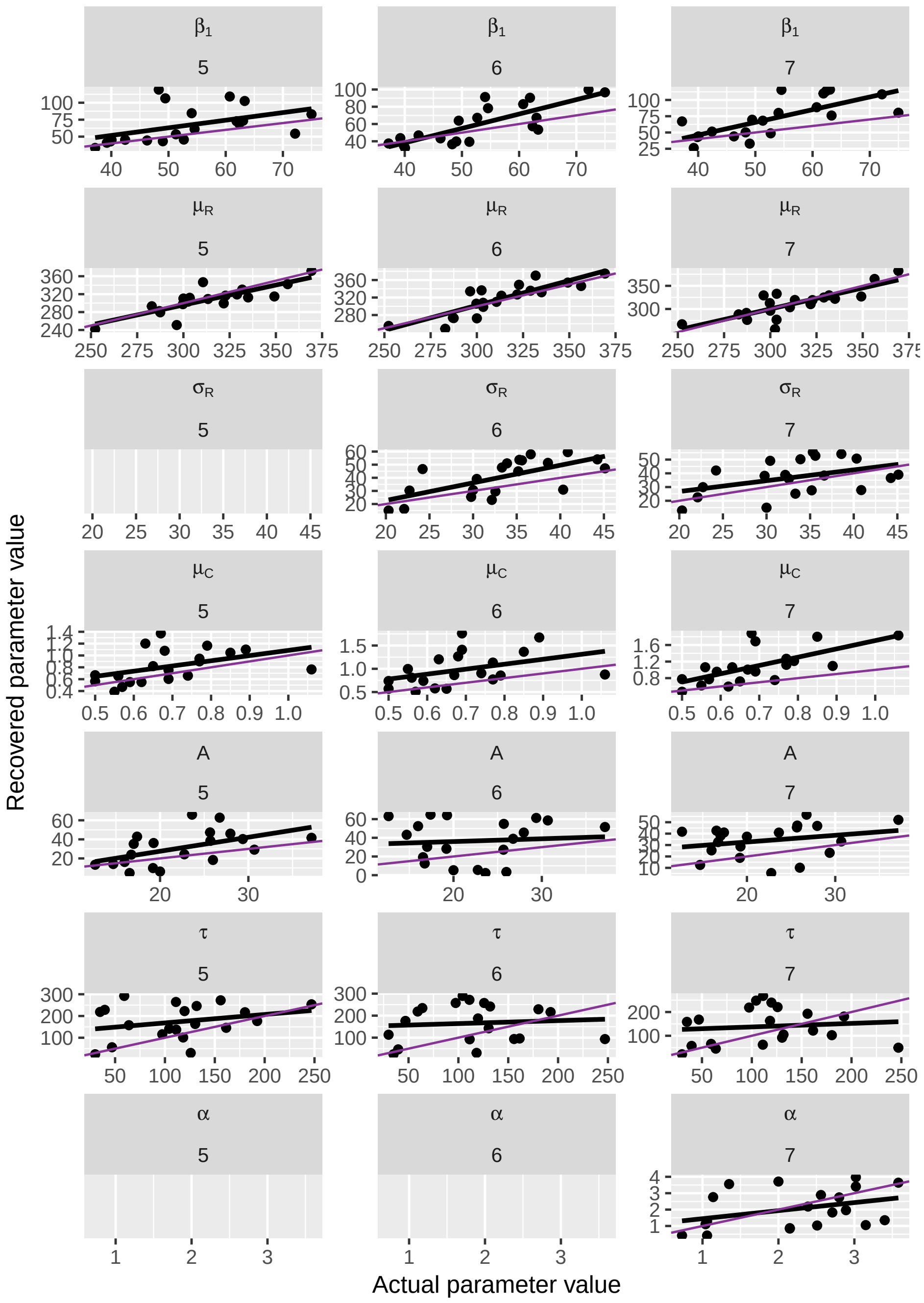


Figure 3.18: Correlations between actual and recovered parameter values using the Bayesian optimization algorithm from distributions with small number of trials. Purple line indicates identity line, while the black line is the correlation between recovered and actual parameters.

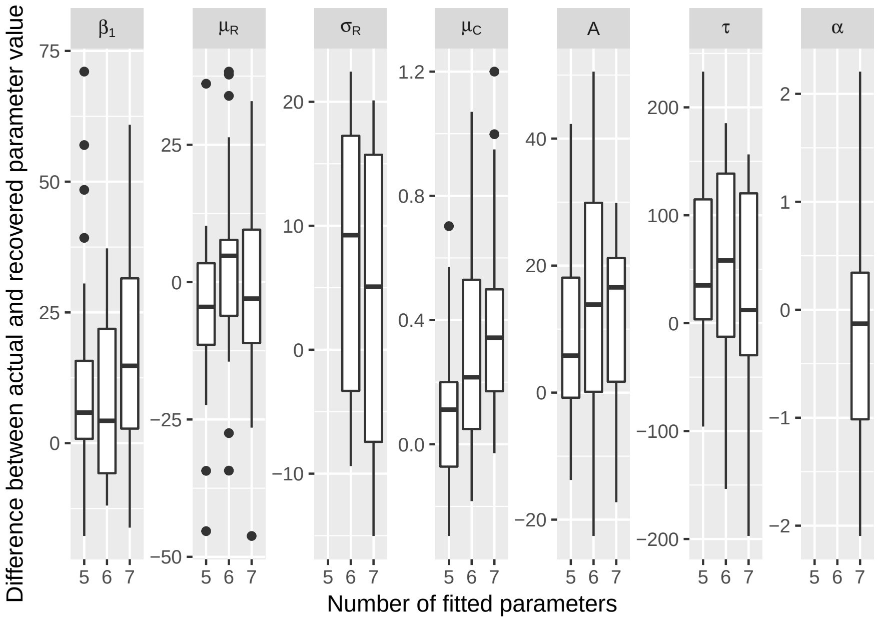


Figure 3.19: Difference between actual and recovered parameter values using the Bayesian optimization algorithm from distributions with small number of trials. Dots indicate outliers.

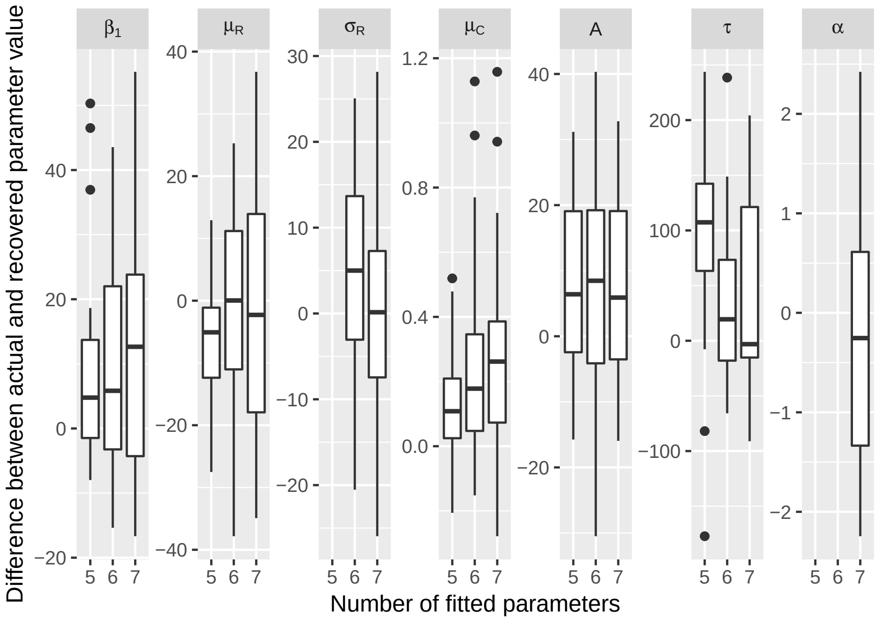


Figure 3.20: Difference between actual and recovered parameter values using the Bayesian optimization algorithm from distributions with large number of trials. Dots indicate outliers.

Table 3.7: Correlation coefficients (p values) between actual parameter values and recovered parameter values using the Bayesian optimization algorithm. Dashes indicate the parameters that were not recovered in that condition. Values displayed in bold indicate statistically significant correlations (not adjusted for multiple comparisons).

	5,000 trials per condition		
Parameters	5	6	7
β_1	**0.70** (5.3×10^{-4})	**0.75** (1.4×10^{-4})	**0.80** (2.1×10^{-5})
μ_R	**0.94** (4.9×10^{-10})	**0.85** (1.8×10^{-6})	**0.78** (4.2×10^{-5})
σ_R	-	**0.45** (0.045)	0.07 (0.773)
μ_C	**0.56** (0.011)	**0.53** (0.017)	0.42 (0.064)
A	**0.69** (7.2×10^{-4})	0.26 (0.276)	**0.73** (2.8×10^{-4})
τ	-0.05 (0.840)	**0.46** (0.040)	0.34 (0.139)
α	-	-	-0.17 (0.470)
	200 trials per condition		
Parameters	5	6	7
β_1	**0.47** (0.038)	**0.80** (2.4×10^{-5})	**0.73** (2.6×10^{-4})
μ_R	**0.82** (9.9×10^{-6})	**0.85** (1.6×10^{-6})	**0.81** (1.7×10^{-5})
σ_R	-	**0.67** (0.001)	0.44 (0.054)
μ_C	**0.45** (0.049)	0.41 (0.073)	**0.66** (0.002)
A	**0.52** (0.018)	0.08 (0.724)	0.26 (0.264)
τ	0.28 (0.236)	0.09 (0.707)	0.11 (0.640)
α	-	-	0.36 (0.114)

(all parameters), six (with the shape of the starting point distribution fixed), or five (with the shape of the starting point distribution and the variability of non-decision time fixed). The optimization algorithms had limited number of time to recover parameters: 72 hours for distributions with large number of trials, and 24 hours for distributions with small number of trials. The results from all of the algorithms suggest that some parameters, like upper boundary and non-decision time, are easier to recover, given that all algorithms (except for basin hopping) managed to recover parameter values that had high and significant correlations with actual parameter values. The recovery of other parameters varied between algorithms and conditions, suggesting that they are harder to recover using global optimization algorithms.

The algorithm that performed the best in recovering parameters of diffusion model for conflict tasks from distributions with both high and low number of trials, was the differential evolution algorithm. This result came from comparing objective function values returned by the different optimization algorithms. When comparing correlation values between recovered and actual parameters, differential evolution seemed to recover more parameters well compared to other optimization algorithms, however this difference was not statistically significant. The finding that differential evolution seems to outperform other global optimization algorithms in parameter recovery is consistent with Hawkins et al. (2015), who found that differential evolution provided better parameter recovery from drift diffusion model than particle optimization and simplex algorithms. The result is also consistent with research outside of the decision making modelling field, as Civicioglu and Besdok (2013) found that differential evolution outperformed other global optimization algorithms, such as particle swarm optimization and artificial bee colony, in 50 benchmark numerical optimization functions.

Constraining the number of parameters to be recovered did not have any effect on the performance of the algorithms. There were some differences when investigating the performance of individual algorithms, but the only significant difference was found in differential evolution algorithm, while recovering parameters from distributions with large number of trials, where performance in recovering all seven parameters was worse than the performance in recovering five or six parameters. There was no difference in performance of differential evolution when recovering parameters from distributions with small number of trials. This result suggests that when trying to recover parameters from experimental studies, which tend to contain lower number of trials, constraining the number of parameters to be recovered is not necessary.

One drawback with comparing different global optimization algorithms in this chapter was the differences in runtimes of the algorithms. Some algorithms were computationally quicker than others, in that they performed more function evaluations in the given time than others (for example, basin hopping only managed 15 function evaluations compared to 750 function evaluations performed by Bayesian optimization in the same time frame). In this chapter, all algorithms were allowed to continue the search of the best solution for approximately 72 hours for distributions with large number of trials, and 24 hours for distributions with small number of trials. This time limit of 72 hours was selected in accordance with maximum time limit imposed by Cardiff University High Performance Computing cluster usage policies. If time is no constraint, or if time is even more constrained, the selection of the best performing algorithm might differ. A clear example is the basin hopping algorithm. Although it is a very powerful algorithm, it only managed to perform fifteen iterations of the algorithm, which was not nearly enough to find a solution on par with the other algorithms, as evident by the very high objective function values.

Another issue that needs highlighting is that all global optimization algorithms in this chapter were run with default hyperparameter settings (except for the number of iterations that was adjusted to fit into specific time frame). Because of this, some or all of the algorithms may not have performed as well as they potentially could, if hyperparameters were tuned appropriately. Every algorithm used in this section has potential adjustments and improvements that might make them better at recovering parameters from the diffusion model for conflict tasks, however, hyperparameter optimization is extremely time-costly due to the fact that each reaction time distribution is fit independently. When different versions of the algorithms are also considered, the task becomes computationally unjustifiable without prior knowledge of what would work best. Therefore the conclusions that differential evolution is the best investigated global optimization algorithm for parameter recovery from diffusion model for conflict tasks should be adjusted to highlight that it is only true when comparing out-of-the-box, default algorithms. If global optimization algorithms were compared with appropriate hyperparamers optimized, differential evolution might not come out on top.

It is worth noting that none of the global optimization algorithms investigated in this chapter did exceptionally well. Even though differential evolution performed the best in recovering parameters from distributions with both large and small number of trials, it needs to be stressed that the correlations between recovered and actual parameters varied significantly for different parameters. Even though the correlations were as high as 0.98 for some parameters like the non-decision time, for others like the time to peak automatic activation they were as low as 0.12. Therefore if differential evolution is used in fitting reaction time distributions to obtain parameters from diffusion model for conflict tasks, caution needs to be taken when interpreting any results relating to parameters that are more difficult to recover.

A huge disadvantage of all the global optimization algorithms is that they do a lot of objective function evaluations to fit a single reaction time distribution, but do not store or remember any of this information when fitting the next one. As computing large reaction time distributions is the most computationally expensive process in all global optimization algorithms, this is a very wasteful approach. Global optimization algorithms are also very slow to run. If hundreds or thousands of distributions need to be fitted, it might take weeks and months of computational time. Even when allowing 72 hours for parameter recovery, if all twenty distributions are evaluated in a serial manner, that would take 60 days. This speed was also achieved by using CYTHON implementations for computing reaction time distributions, which is very expensive in researcher time. Recovering parameters from the diffusion model for conflict tasks is so computationally expensive, that researchers cannot apply the model in their research to individual participants if they do not have access to High Performance Computing facilities (Ambrosi et al., 2019). For this reason, the next chapter will investigate whether deep neural networks can learn representations between reaction distributions and parameters from diffusion model for conflict tasks, and therefore could very quickly and accurately predict parameters from diffusion model from conflict from unseen data.

4 Deep learning for parameter recovery from diffusion model for conflict tasks

4.1 Introduction

Deep learning is becoming a very popular machine learning algorithm for learning from and making predictions about data in a variety of fields, such as computer vision (Gebru et al., 2017; Cruz et al., 2017), natural language processing (Li, 2017), bioinformatics (Uziela et al., 2017), among many others, due to beating records in many artificial intelligence problems (LeCun et al., 2015). In the simplest form, deep learning refers to artificial neural networks with multiple hidden layers to learn representations of data using backpropagation. Even though deep learning is not a new development in machine learning, it has only started gaining popularity in the last decade. This is because artificial neural networks were for many years wrongfully presumed to be exceptionally susceptible to local minima (LeCun et al., 2015), and because deep architectures were hard to implement given limited computational power. Due to increasing access to cheap but powerful specialist computing resources, the methodology has benefited greatly from developments in different architectures like convolutional neural networks (Krizhevsky et al., 2012), recurrent neural networks (Mikolov, Karafiát, Burget, Cernockỳ, & Khudanpur, 2010), long short term memory cells (Hochreiter & Schmidhuber, 1997), generative adversarial networks (Goodfellow et al., 2014), increasingly being applied to new areas. On top of hardware accessibility, artificial intelligence methods are becoming more acceptable due to a move away from explanatory to predictive science, even in psychology (Yarkoni & Westfall, 2016). However, neither machine learning nor deep learning has been applied in decision making models yet.

One area where machine learning could be useful in decision making modelling is parameter recovery. As seen from section 2 and section 3, parameter recovery from diffusion model for conflict tasks (Ulrich et al., 2015) can be tricky, even when imposing constraints on the solution space. Deep learning seems to be particularly suited to parameter recovery from non-differentiable functions, because the power of deep learning algorithms comes from harnessing very large amounts of training data. Application of deep learning can be very problematic in areas where data is either limited or hard to collect, such as clinical health care (Miotto et al., 2017), however, in decision making modelling, unlimited amount of data can be generated very easily. Deep learning could also be advantageous for parameter recovery because parameters can be recovered independently from each other, unlike with global optimization algorithms, where the whole set of parameters is recovered at once. In this way, a set of parameters mimicking change in another parameter, like reported in section 2, would not have an effect on parameter recovery. Moreover, unlike with global optimization algorithms, deep neural networks learn representations of thousands of reaction time distributions at once, therefore local minima are not an issue. Moreover, fitting thousands of distributions at once allows defining the prediction error, so even when single distributions are fitted, it is possible to know the approximate error of the recovered parameters, which would be very time consuming and difficult to achieve with global optimization algorithms.

Finally, the biggest advantage of deep learning over global optimization algorithms in parameter recovery from diffusion model for conflict tasks is that once a deep learning model is trained to predict parameters from diffusion model for conflict tasks, it can be deployed and applied for prediction as a simple calculation, therefore making parameter recovery a task that takes seconds. This would allow researchers who do not have access to significant computational resources to apply the diffusion model for conflict tasks to research where individual differences are important.

There are a lot of differences in how parameters from the diffusion model for conflict tasks are recovered using deep learning and global optimization algorithms. The main difference is that deep learning models need to be trained with engineered features and not reaction time distributions that were used with global optimization algorithms. Even though one of the biggest advantages of deep statistical learning methods over shallow ones is that feature engineering is not necessary as the models can use raw data for inputs (LeCun et al., 2015), treating reaction time distributions as inputs in deep learning models is problematic for multiple reasons. First of all, unlike in convolutional neural networks, reaction time distribution data is not spatially correlated. This means that if a trial sequence in reaction time distributions were reshuffled, the distribution itself would not change. This makes neural networks with convolutions unsuitable. Secondly, even though we know there are temporal dependencies in reaction times in decision making, decision making models like the diffusion model for conflict tasks treat each trial as independent, therefore recurrent network architectures are also unsuitable. For this reason, a network with fixed engineered meaningful inputs needs to be employed. Therefore the reaction time distributions will be converted to input features, as described in section 4.2.

The purpose of this chapter is to see whether deep learning can be applied to parameter recovery of models with non-differentiable objective functions, and if so, whether the use of deep learning brings improvements over using global optimization algorithms. Differently from global optimization algorithms, the deep learning models will be trained on thousands of distributions, and assessed in two ways: first, as in section 3, using the twenty test distributions, and secondly, as is customary in artificial intelligence research, using test data of 50,000 distributions, that are excluded from the model until the final testing phase, to allow evaluating the predictive performance of the models.

The deep learning models in this chapter will be trained with input data from reaction time distributions with a high number of trials (each reaction distribution consisting of 15,000 trials, 5,000 trials in each congruent, incongruent, and neutral trials). This will evaluate whether deep learning is capable of learning representations between input features describing a reaction time distribution, and model parameters from the diffusion model for conflict tasks. The deep learning models will then be evaluated using data with a large number of trials and data with a small number of trials. It is unlikely that models trained with data with a large number of trials will be good at predicting data with a low number of trials, as deep learning models are very data-specific (Najafabadi et al., 2015). If this is the case, the next chapter (section 5) will examine how the predictive power of the deep learning models can be improved from inputs with low number of trials.

4.2 Methods

All deep learning models were designed using the Keras library (Chollet et al., 2015) for python with TensorFlow backend (Abadi et al., 2015). Some data preprocessing required scikit-learn (Pedregosa et al., 2011) functionality.

4.2.1 Data

The training dataset consisted of 150,000 reaction time distributions of 15,000 trials each (5,000 trials per condition, three conditions: congruent, incongruent, and neutral). These were computed from randomly chosen sets of parameters from the diffusion model for conflict tasks, from uniform distributions with the following limits:

- Upper threshold (β_1): $[20, 120]$
- Non-decision time (μ_R): $[150, 500]$
- Standard deviation of non-decision time (σ_R): $[10, 60]$
- Drift rate for controlled process (μ_C): $[0.01, 2]$
- Amplitude of automatic activation (A):$[0.01, 70]$
- Time to peak automatic activation (τ): $[10, 300]$
- Shape of starting point distribution (α): $[0.01, 4]$

These limits corresponded to the search space defined in section 3 for all global optimization algorithms. The data from section 2 could not be reused here as each parameter could take only one of seven values, making the data unsuitable for deep learning models, which need to learn the representation of the whole search space.

Each model parameter was chosen independently from all other parameters, and the probabilities of parameter values were uniform, as is confirmed in figure 4.1. Each reaction time distribution was used to calculate 92 input features that would act as the predictors in the deep learning models: 23 unique features calculated four times for the whole distribution, congruent condition only, incongruent condition only, and neutral condition only. The 23 features were:

- mean reaction time of correct trials
- mean reaction time of incorrect trials
- proportion of correct trials
- mean of the normal distribution for correct trials
- standard deviation of the normal distribution for correct trials
- exponential decay parameter for correct trials

- mean of the normal distribution for incorrect trials
- standard deviation of the normal distribution for incorrect trials
- exponential decay parameter for incorrect trials
- minimum reaction time value of correct trials
- maximum reaction time value of correct trials
- minimum reaction time value of incorrect trials
- maximum reaction time value of incorrect trials
- reaction time of *five* bins of conditional accuracy function
- accuracy of *five* bins of conditional accuracy function.

A large number of input features were chosen to fully describe a reaction time distribution. The inclusion of all of these features is most likely unnecessary, as they are highly correlated, however, the number of features is not large enough to have a noticeable impact on computational speed for the deep learning models. The inclusion of some features, particularly the ex-Gaussian parameters, could be questioned, as there is evidence to suggest that ex-Gaussian parameters have no unique relationship with drift diffusion model parameters (Matzke & Wagenmakers, 2009). However, the diffusion model for conflict tasks has additional "impulsivity" parameters to the standard drift diffusion model, therefore they might still be useful. Moreover, as deep learning models are non-linear, they might be able to use information from ex-Gaussian parameters to find high-dimensional interactions with other input features that are not easily detectable with simple linear correlations.

The test dataset was 50,000 reaction time distributions of 15,000 trials each, computed in an identical way to the training dataset. The test dataset was computed after the hyperparameter optimization was completed, and introduced only for the final evaluations of the predictive performance of the deep learning models [3]. Even though there is a possibility that some samples in the test dataset have appeared in the training dataset as

[3] Please note that the final evaluation means the final training of the model across all epochs. The test data does not have any effect on the training itself, but is used to evaluate the model performance throughout the whole process.

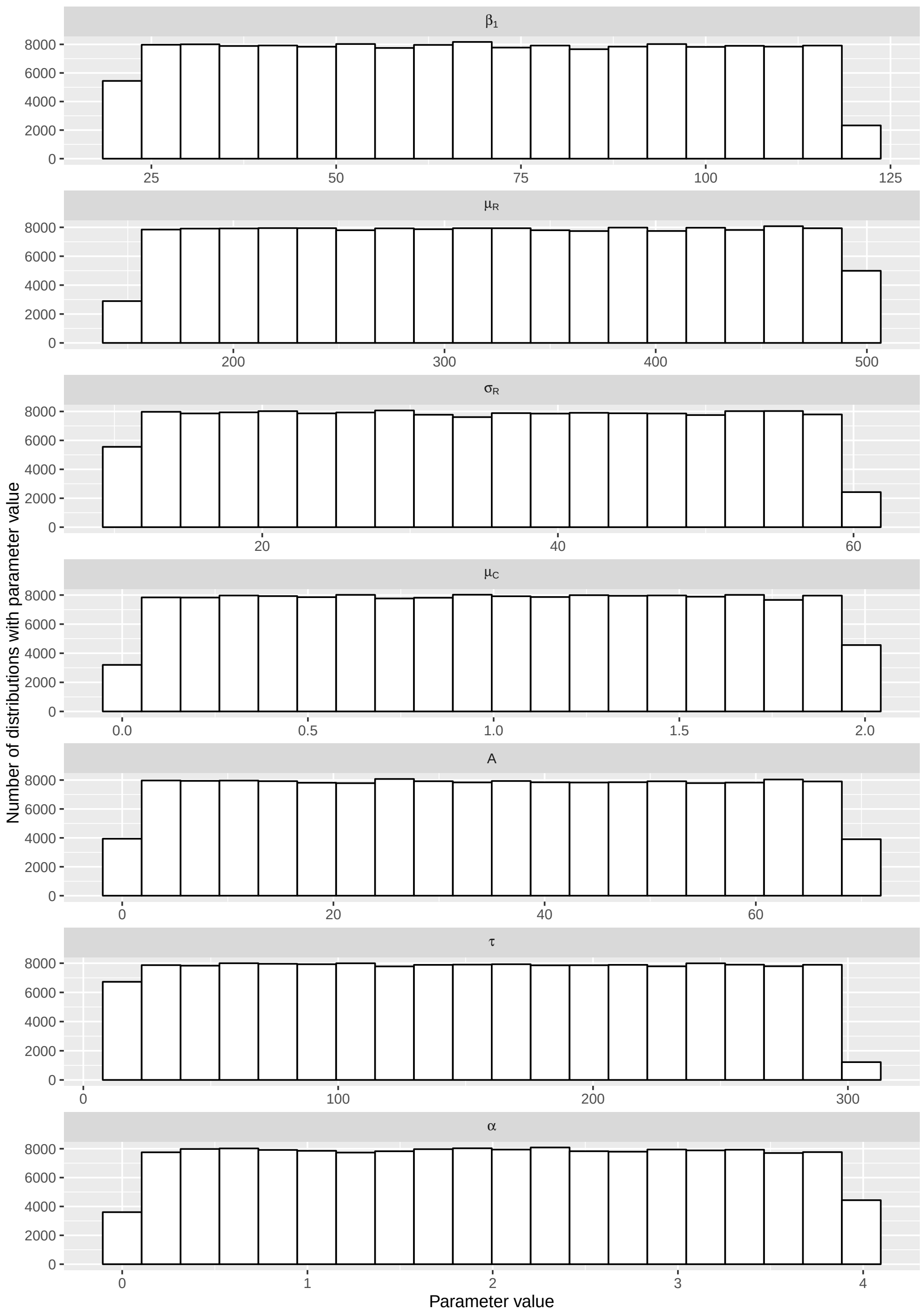


Figure 4.1: Distribution of parameter values in the training dataset

well, the inherent randomness in computation of each trial in the reaction time distributions should make the input features slightly different, even when the sets of parameters to compute the reaction time distributions are the same.

Additionally, to allow comparison with global optimization algorithms, the twenty samples from section 3 were also used in this chapter to evaluate deep learning model performance. The data contained in the twenty samples was identical to that in the section 3, the only difference being that input features were extracted from raw reaction time distributions as described above.

To see whether deep learning models can predict outputs from data with low trial numbers after being trained on data with high trial numbers, smaller datasets of 600 trials each (200 in each conguent, incongruent, and neutral conditions) were computed. The reaction time distributions were created from sets of parameters from the diffusion model for conflict tasks that comprised the training and the test datasets. In this way, the datasets with low trial numbers had exactly the same diffusion model for conflict tasks parameters in the training and the test datasets as the datasets with high trial numbers. For the test dataset of twenty distributions, the data with low trial numbers was identical to that used in section 3.

4.2.2 Hyperparameter optimization

Hyperparameters are parameters of machine learning algorithms that need to be set (as opposed to learned) before the machine learning models are trained. Hyperparameters have a huge effect on how the machine learning algorithm will perform with given input data. In deep learning, the weights of the layers are the parameters that are being learned by the deep learning model, whereas parameters like the number of hidden layers, the size of each hidden layer, the learning rate, any regularization, dropout values, etc, are hyperparameters that need to be set before the model starts training.

As mentioned in section 1.5, deep learning can be decomposed into two main parts: prediction of the outputs (feed-forward) and learning of the weights (backpropagation). Prediction of the outputs consists of taking the inputs into the model, multiplying them by the weights, summing the results, applying a non-linear transformation to the output, and repeating the process over all hidden layers until the output layer is reached. A very simplified illustration of the process is provided in figure 4.2. The weights of the layers are first initialized randomly, and therefore the predictions from such a model are going to be random as well. The difference between the actual and the predicted outputs [4] is called the error. In order for the model to learn the appropriate weights to predict the outputs correctly, a process called backpropagation is needed, which propagates the error between the predicted and the actual outputs and changes the weights in the direction of the gradient of the cost function. The actual computations for this process are quite complex, however in simple terms, a gradient descent optimization algorithm is used to calculate the gradient of the error, and then the weights are moved in the direction of the gradient by an amount specified by the learning rate. A small learning rate means that it is going to take a very long time to train the model, as the weights are changed by very small amounts, however a large learning rate means that the weights can overshoot the minimum and not converge. This adjustment of weights is an iterative process; a single iteration (a complete pass over the whole training dataset) in deep learning is called an epoch. The number of epochs has an effect on how well a deep learning model learns to predict the data: not enough epochs will result in poor performance as the weights will not be set to their optimum values, while too many epochs can result in overfitting, affecting the generalization of the model. There are many different gradient descent optimization algorithms; the overview is provided in Ruder (2016). The gradient can be calculated over the whole set of inputs, however this is a very computationally expensive approach if the number of inputs is substantial. Alternatively, the gradient can be calculated for a subset of inputs, which speeds up the computation time without affecting the model performance significantly. The size of the subset of inputs that is used to calculate the gradient is called batch size. Finally, to prevent the model from overfitting, regularization can be applied to the weights of the model. In simple terms, regularization makes the weights more consistent, therefore reducing the chance that the model will learn direct input-output mappings.

As deep learning has never been applied to parameter recovery and prediction from decision making models, it is unknown what hyperparameter values should be chosen for this problem. Finding the values of hyperparameters for optimal model performance is called hyperparameter optimization (also known as hyperparameter tuning). Hyperparameter optimization evaluates how the deep learning models perform with different hyperparameter values, and then selects the hyperparameter values that give the best

[4]There are multiple ways to calculate the difference between actual and predicted outputs, which will depend on the type of outputs, for example, cross entropy for categorical outputs and mean squared error for numerical outputs. This is called a loss function.

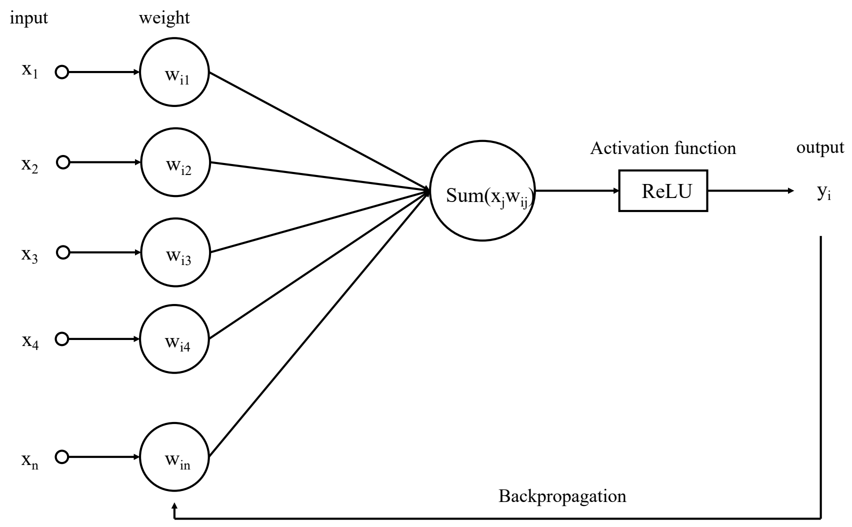


Figure 4.2: Illustration of the deep learning calculations. This illustrates a very simple model with a single hidden layer of size 1 and a single output.

performance by the deep learning model.

Hyperparameter optimization was performed on the training dataset with high trial numbers for each deep learning model individually, to find the most suitable set of hyperparameters. A combination of manual search and systematic optimization was performed, the latter with the help of the HYPERAS library (http://maxpumperla.github.io/hyperas/), a wrapper of HYPEROPT package (Bergstra, Yamins, & Cox, 2013) for KERAS. A few parameters for the model were chosen by manual search, such as root mean square propagation (rmsprop) optimization function, batch size of 75 samples, L1 regularization function, and rectifier linear unit (ReLU) activation in layers. ReLU is a very simple but effective activation function that nulls all negative values, while keeping the positive values as is. The approximate width of hidden layer architecture was also predetermined by manual search. Other parameters, like the learning rate and the L1 regularization value, came from pilot work with automatic hyperparameter optimization. The learning rate of 0.0001 was chosen as the best rate of learning for all of the models every time, therefore it was not optimized further. L1 regularization value was set to 0.0001 as any value between 0 and 0.1 did not affect model performance much, but adding L1 regularization resulted in less erratic performance on the test dataset during model training. Although dropout rate (explained further in the Overall Model Architecture section) was initially considered as a simple way to prevent model overfitting, it was not necessary due to the narrowness of the final deep learning model architectures.

Finally, the number of hidden layers, and the size of each hidden layer, were optimized using HYPERAS. The automated optimization procedure started by defining the possible model architectures with either three of four hidden layers and the size of each hidden layer (chosen independently from 64, 128, 256, or 512 units). The models were evaluated after running for 25 epochs five times for each different testing split (see next section). The number of different evaluations was set to 50 per model.

4.2.3 Cross validation

When hyperparameter optimization algorithms are searching for the best set of hyperparameters to solve a given problem, they are given only the training dataset to evaluate the performance of hyperparameters. Because machine learning models are quite powerful, they can easily learn how to fit noise in the data in order to improve model performance. For this reason, having a training dataset split into fixed parts for training and evaluating the model can produce results that will not translate well to unseen test data. One way to avoid this is to cross-validate the performance of the model on multiple validation datasets.

Cross validation, in simple terms, tries to define a dataset to test the model in the training phase (including hyperparameter optimization) in such a way that the model generalizes well to unseen test data. One approach to this is called K-fold cross validation. K-fold cross validation splits the training dataset into equal K parts (for example, five-fold cross validation would split the training datasets into five equal parts). Then the model is trained on K-1 parts, and tested on one part. As there are K possible parts to act as a test dataset now, this process is repeated K times, each time a different split acting as a test dataset. The model evaluation is then averaged over K splits. In this way, the model that is more likely to fit noise should perform less well than a model that is more likely to fit signal, as noise changes in each test split, while the signal remains the same.

The number of folds in K-fold cross-validation is the choice of the researcher, however there are some issues to consider. The more folds are chosen, the longer it is going to take to train the model, therefore if computational time is an issue, a high number of folds is going to negatively affect the time to train the model. Conversely, lower number of folds increases the chances of model being able to overfit training data.

K-fold cross validation was performed during hyperparameter optimization. Five folds of data were chosen as a good balance between computational time and coverage of data. Therefore the training sample of 150,000 was split into five batches of 30,000 samples for each selection of hyperparameters. The split was random for each set of hyperparameters. Data in four training folds was then used to create a scaler to scale the input features, which was subsequently applied to training and testing splits independently. In this way, the testing fold had no input in creation of the scaled features. Then, the deep learning model was trained five times; each time a different fold of the data acted as the test dataset. The average R^2 of all five test datasets (folds) was then used as the evaluation metric for a set of hyperparameters.

4.2.4 Overall model architecture

Seven different models were trained, one for each of the seven parameters from the diffusion model for conflict tasks as the output. The same input data was used in all seven models. The model architecture consisted of:

- sequential model
- dense layers
- bias term included in all layers
- ReLU activation in all hidden layers
- RMSprop optimization function with parameters $\rho = 0.9$, $\epsilon = 1e - 08$, decay=0.0
- batch size of 75 samples
- L1 regularizer on activation in each layer (set to 0.0001).
- mean square error as loss function
- learning rate set to 0.0001

Dropout is a technique that allows models to control overfitting of the training data (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014). This was not added to the models in this chapter as the chosen network architecture was too shallow to ensure that the networks would recover from dropout layers. It was also less necessary as the models were only trained for 25 epochs until a sufficiently good model performance was reached.

The number of hidden layers and size of each hidden layer was optimized separately for each model. They are reported in table 4.1. Models with fewer hidden layers imply that the parameter is easier to recover, as simpler low level features seem sufficient to map the inputs and the outputs.

Table 4.1: Optimized hidden layer sizes for deep learning models using input data with high trial numbers. If only three hidden layers were used, Layer 4 size is reported as "-". The R^2 values come from evaluating the best performing model chosen by hyperas on 150,000 samples of the training dataset.

Parameter	R^2 value	Layer 1	Layer 2	Layer 3	Layer 4
β_1	0.983	256	512	512	128
μ_R	0.999	512	64	512	-
σ_R	0.998	512	512	256	512
μ_C	0.974	512	256	64	128
A	0.990	256	512	512	128
τ	0.962	256	512	512	256
α	0.976	256	512	512	64

Each model was trained for 25 epochs, with mean square error between actual and predicted parameters as a loss function. To evaluate the performance of the predictive models, R^2 between outputs and targets, mean absolute deviance from the target, and mean error between output and target [5], were also calculated in each epoch. These were not used to train the models in any way.

[5]Mean error can help reveal bias in the model if it does not center around zero, which could indicate issues with the training dataset.

4.3 Results

4.3.1 Predicting the diffusion model for conflict tasks parameters from data with a high number of trials

The performance of all the final deep learning models was evaluated on a test dataset of 50,000 randomly computed reaction time distributions of 15,000 trials. Three performance measures were used: R^2, mean absolute difference, and mean difference between the predicted outputs and the targets. Pearson correlations were also computed between predicted and target outputs. All final trained models are available on https://github.com/SolVG/pyCDM.

Table 4.2: Final values of all four measures used in model fitting for the test dataset.

Parameter	Mean square error	R^2	Mean error	Mean absolute error
β_1	32.19	0.97	0.53	3.03
μ_R	55.41	0.995	-2.69	4.41
σ_R	2.52	0.988	0.86	1.16
μ_C	0.0099	0.971	-0.027	0.063
A	33.30	0.925	0.32	3.37
τ	1656	0.763	4.56	25.54
α	0.14	0.898	-0.13	0.26

Table 4.3: Correlation coefficients and p values between actual parameter values and recovered parameter values from deep learning models trained with and tested on data with high number of trials.

Parameter	Correlation coefficient	p value
β_1	0.986	$p = 2.2 \times 10^{-16}$
μ_R	0.998	$p = 2.2 \times 10^{-16}$
σ_R	0.996	$p = 2.2 \times 10^{-16}$
μ_C	0.987	$p = 2.2 \times 10^{-16}$
A	0.963	$p = 2.2 \times 10^{-16}$
τ	0.878	$p = 2.2 \times 10^{-16}$
α	0.957	$p = 2.2 \times 10^{-16}$

The history of all the models training over 25 epochs is displayed in figure 4.3 for mean square error, R^2, mean error, and mean absolute error. The mean square error, R^2, mean error, and mean absolute error values for predicting test data are also reported in table 4.2. The model fitting history figure shows that all the models converged to a good R^2 value quickly.

The correlations between target and predicted parameter value for each parameter are displayed in figure 4.4. As evident in the figure, the models predicted each parameter very well. The correlation coefficients and p values are reported in table 4.3. All correlation coefficients were above 0.95, except for time to peak automatic activation, which had correlation coefficient of 0.88. This indicates that the model for time to peak automatic activation might benefit from further hyperparameter optimization or changes in model architecture.

For the twenty test distributions that were used in section 3, trained deep learning models were used to predict the parameters from diffusion model for conflict tasks. The resulting correlations between recovered and target parameters are reported in table 4.4 and visualized in figure 4.5. The figure and table show that unlike the tested global optimization algorithms, the deep learning models were able to predict the twenty test distributions with high trial numbers exceptionally well, with all correlations between predicted and target parameters having R values of 0.95 and higher.

Table 4.4: Correlation coefficients (and p values) between actual and recovered parameter values from twenty datasets from section 3 using deep learning models from high and low trial number input data.

Parameter	Correlation coefficient - dataset with large number of trials	p value - dataset with large number of trials	Correlation coefficient - dataset with small number of trials	p value - dataset with small number of trials
β_1	0.995	$p = 2.2 \times 10^{-16}$	0.875	$p = 4.5 \times 10^{-7}$
μ_R	0.996	$p = 2.2 \times 10^{-16}$	0.976	$p = 1.9 \times 10^{-13}$
σ_R	0.998	$p = 2.2 \times 10^{-16}$	0.903	$p = 5.0 \times 10^{-8}$
μ_C	0.958	$p = 3.2 \times 10^{-11}$	0.768	$p = 7.7 \times 10^{-5}$
A	0.969	$p = 2.0 \times 10^{-12}$	0.633	p=0.003
τ	0.947	$p = 2.5 \times 10^{-10}$	0.766	$p = 8.1 \times 10^{-5}$
α	0.981	$p = 2.3 \times 10^{-14}$	0.829	$p = 6.2 \times 10^{-6}$

Since there were seven deep learning models to predict each diffusion model for conflict tasks parameter individually, it is unclear whether the seven parameters as a whole would be a good fit to the reaction time distribution from which the parameters were pre-

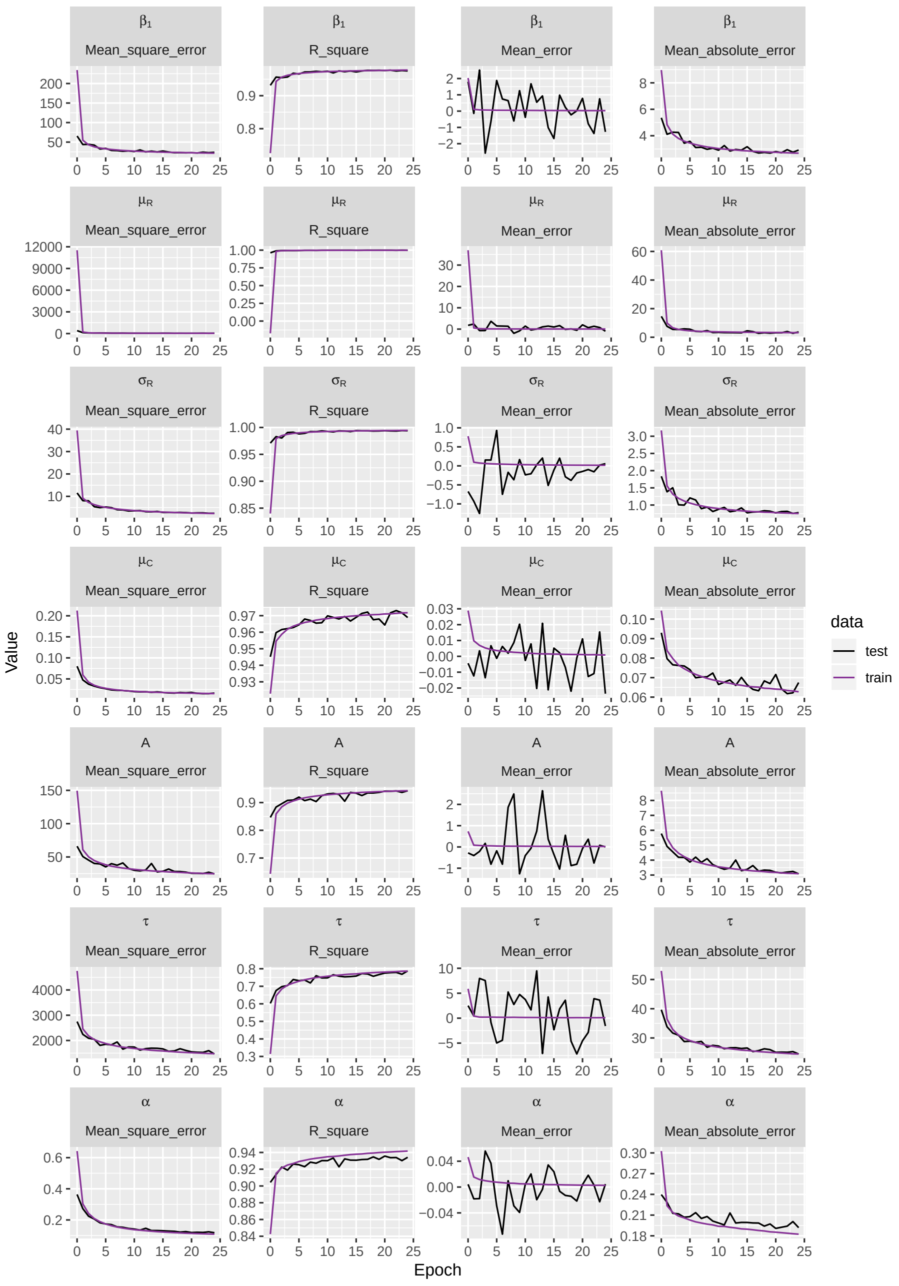


Figure 4.3: History of model training for predicting parameters from the diffusion model for conflict tasks of four measures: mean square error (used as a loss function, outer left column), R^2 (inner left column), mean error (inner right column), and mean absolute error (outer right column). The seven parameters from the diffusion model for conflict tasks are reported in seven rows of the figure. The purple line represents training data, while the black line represents evaluation of the test data.

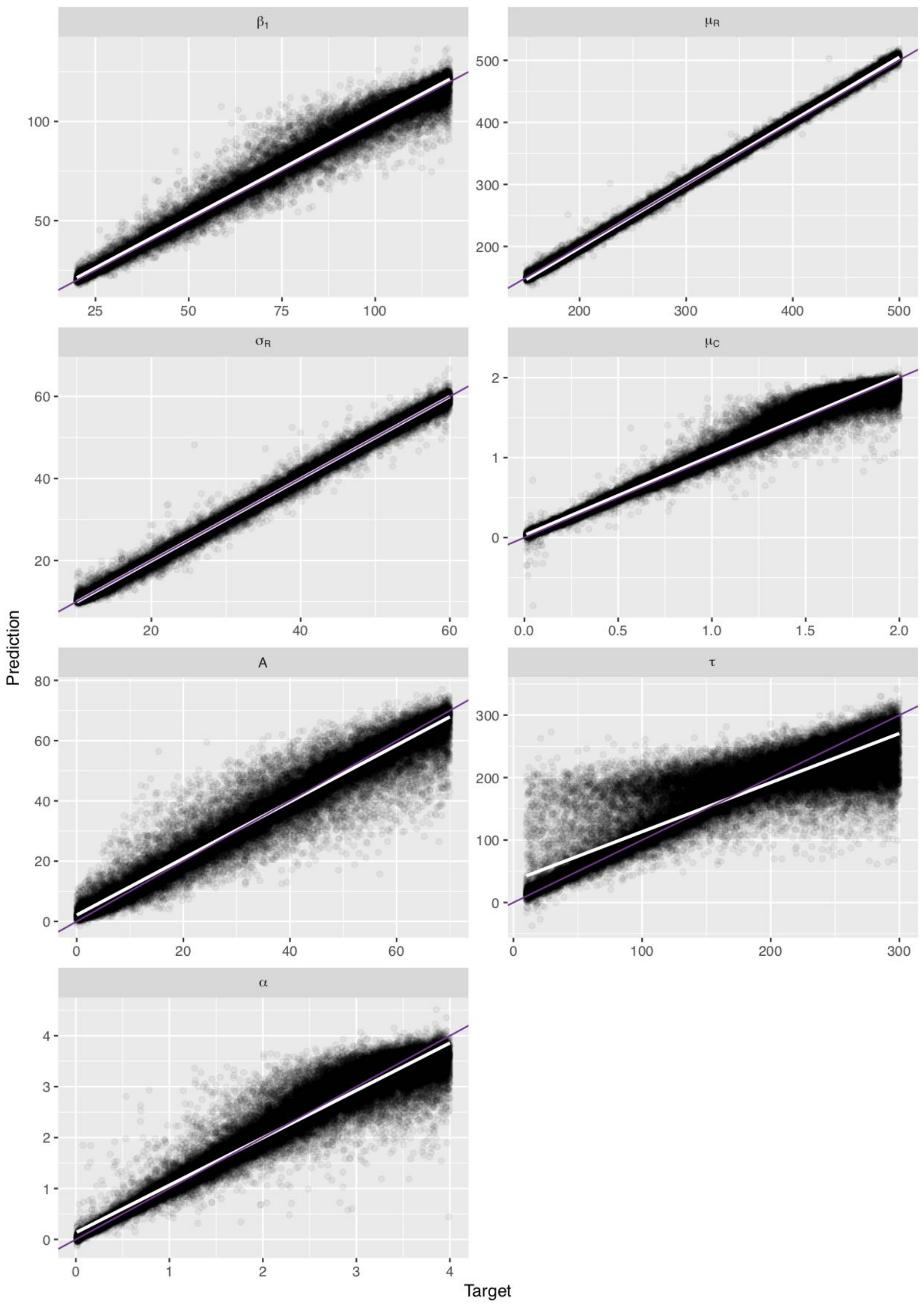


Figure 4.4: Correlation between predicted (y axis) and target (x axis) parameters from diffusion model for conflict tasks for data with large trial numbers (15,000 trials per RT distribution) when the model was trained with data with large trial numbers. White solid lines display fitted regression lines, while solid purple lines display identity lines.

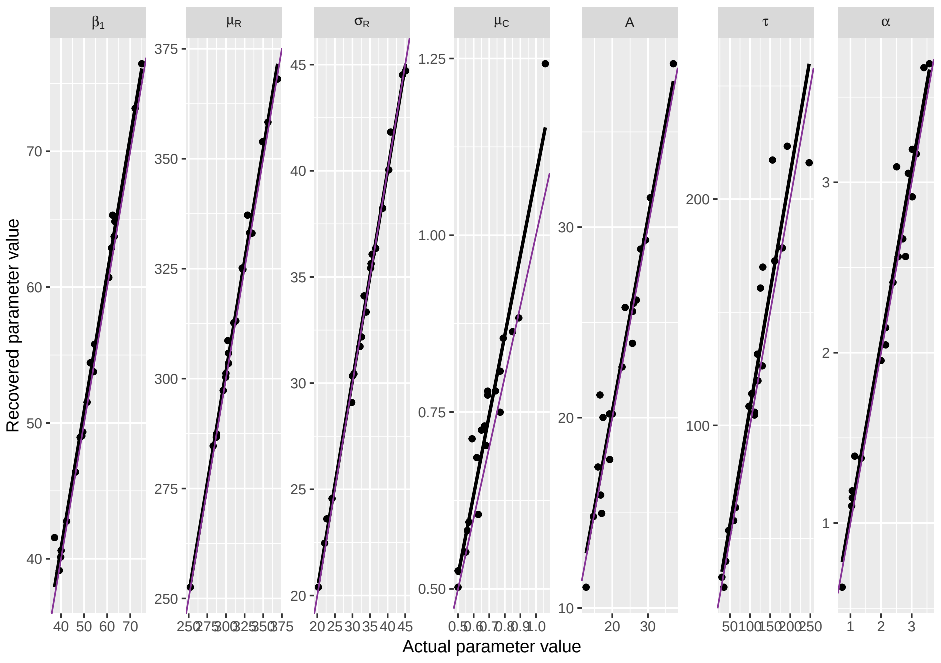


Figure 4.5: Correlation between predicted (y axis) and target (x axis) diffusion model for conflict tasks parameters from twenty datasets from section 3 for input data with large trial numbers (15,000 points per distribution), when the deep learning models were trained on data with large number of trials. Black solid lines display fitted regression lines, while solid purple lines display identity lines.

dicted. In order to evaluate that and whether deep learning models were able to recover parameters of the diffusion model for conflict tasks better than the global optimization algorithms from section 3, the parameters predicted by the deep learning models were used to produce a reaction time distribution consisting of 15,000 trials (5,000 in each congruency condition). Then, the reaction time distribution was compared to the original reaction time distribution from which the parameters were predicted using the Kolmogorov-Smirnov method, as in section 3. The sum of three KS values was reported. The results are displayed in table table 4.5, together with the KS values from each of the global optimization algorithm from recovering seven parameters. The smallest KS value from all optimization algorithms is displayed in bold. The table shows that on average, deep learning recovered parameters better than any optimization algorithm, when looking at the KS values for each of the twenty test distributions, while differential evolution was not far behind (0.091 for deep learning compared to 0.094 for differential evolution). This difference was not statistically significant ($t(19) = -0.26, p = 0.801$). When individual distributions were inspected, twelve out of twenty distributions were recovered the best by deep learning, seven out of twenty by differential evolution, and the remaining one by particle swarm optimization. This shows that in most cases, the highest number of distributions were recovered by deep learning better than any other optimization algorithm tested in section 3, but differential evolution was not far behind.

When comparing correlation coefficients between actual and recovered parameters, deep learning also trumps differential evolution as the correlation coefficients between actual and recovered parameters were higher when recovered by deep learning compared to differential evolution: mean R for differential evolution was 0.53 while mean R for deep learning was 0.98, this difference was statistically significant (Z=28, p=0.016). The difference between KS values and R^2 values from the same models most likely comes from the fast that deep learning models were trained to predict model parameters independently of each other, and KS values in themselves had no effect on model training (whereas they were value that was minimized by the global optimization algorithms).

Table 4.5: Comparison of KS values between actual and recovered RT distributions by global optimization algorithms and deep learning using datasets with large number of trials.

Dataset	Random search	Particle swarm optimization	Differential evolution	Bayesian optimization	Deep learning
1	0.150	0.161	0.129	0.182	**0.083**
2	0.145	0.071	**0.050**	0.165	0.093
3	0.132	0.154	0.102	0.138	**0.093**
4	0.176	0.127	**0.089**	0.196	0.167
5	0.125	0.162	0.074	0.094	**0.066**
6	0.149	0.164	0.103	0.194	**0.057**
7	0.203	0.098	0.111	0.179	**0.078**
8	0.173	0.331	0.129	0.152	**0.057**
9	0.232	0.322	0.152	0.210	**0.140**
10	0.116	0.080	0.087	0.135	**0.058**
11	0.183	0.178	**0.055**	0.153	0.114
12	0.118	0.134	**0.066**	0.179	0.067
13	0.135	0.221	0.119	0.157	**0.058**
14	0.148	0.292	**0.095**	0.137	0.105
15	0.121	0.360	**0.073**	0.188	0.186
16	0.202	0.164	0.114	0.188	**0.058**
17	0.111	**0.072**	0.079	0.091	0.083
18	0.137	0.167	0.117	0.179	**0.113**
19	0.147	0.289	**0.067**	0.161	0.097
20	0.137	0.243	0.068	0.225	**0.052**
mean	0.152	0.190	0.094	0.165	**0.091**

4.3.2 Predicting model parameters from datasets with low number of trials

When models are trained with data with large number of trials, and then predict data with large number of trials, they perform very well. However, it is not known how they would perform with testing data with low number of trials. This section will examine this issue. Firstly, the deep learning models were used to predict the test dataset that was identical to the test dataset with large number of trials in terms of diffusion model for conflict tasks parameter (target) values, but differed in the number of trials in each RT distribution (600 compared to 15,000), thus introducing noise in the input data. The results are displayed in figure 4.6. The correlation coefficients and corresponding p values are reported in table 4.6. The results show that some parameters were predicted quite well (upper threshold, non-decision time, standard deviation of non-decision time, and drift rate for controlled process). The prediction of shape of automatic activation was mediocre, while the remaining parameters, both “impusivity” parameters (amplitude of automatic activation and time to peak automatic activation) could not be recovered at all. The correlations are still reported as statistically significant from zero because of the size of the testing dataset.

Table 4.6: Correlation coefficients and p values between target and recovered parameter values from the diffusion model for conflict tasks from deep learning models trained with data with large number of trials and tested with noisy data.

Parameter	Correlation coefficient	p value
β_1	0.726	$p = 2.2 \times 10^{-16}$
μ_R	0.922	$p = 2.2 \times 10^{-16}$
σ_R	0.827	$p = 2.2 \times 10^{-16}$
μ_C	0.847	$p = 2.2 \times 10^{-16}$
A	0.088	$p = 2.2 \times 10^{-16}$
τ	0.059	$p = 2.2 \times 10^{-16}$
α	0.457	$p = 2.2 \times 10^{-16}$

Secondly, the deep learning models were tested by predicting the diffusion model for conflict tasks parameters from twenty test datasets from section 3 that contained 600 trials per RT distribution. The results are presented in figure 4.7 and table 4.4. They show that the predictive power of the deep learning models has dropped significantly for all parameters from the diffusion model for conflict tasks, though none was below R=0.63.

Since there were seven deep learning models to predict each diffusion model for conflict tasks parameter individually, it is unclear whether the seven parameters as a single group would be a good fit to the reaction time distribution from which the parameters were predicted. In order to evaluate that, and whether deep learning was able to recover parameters of the diffusion model for conflict tasks better than the global optimization algorithms from section 3, the parameters predicted by the deep learning models were used to produce a reaction time distribution consisting of 3,000 trials (1,000 in each congruency condition). These numbers were chosen to maintain consistency with the global optimization algorithms, where 1,000 trials per condition were computed in each comparison. Then, the resulting reaction time distribution was compared to the original reaction time distribution from which the parameters were predicted using the Kolmogorov-Smirnov method, as in section 3. The sum of three KS values are reported

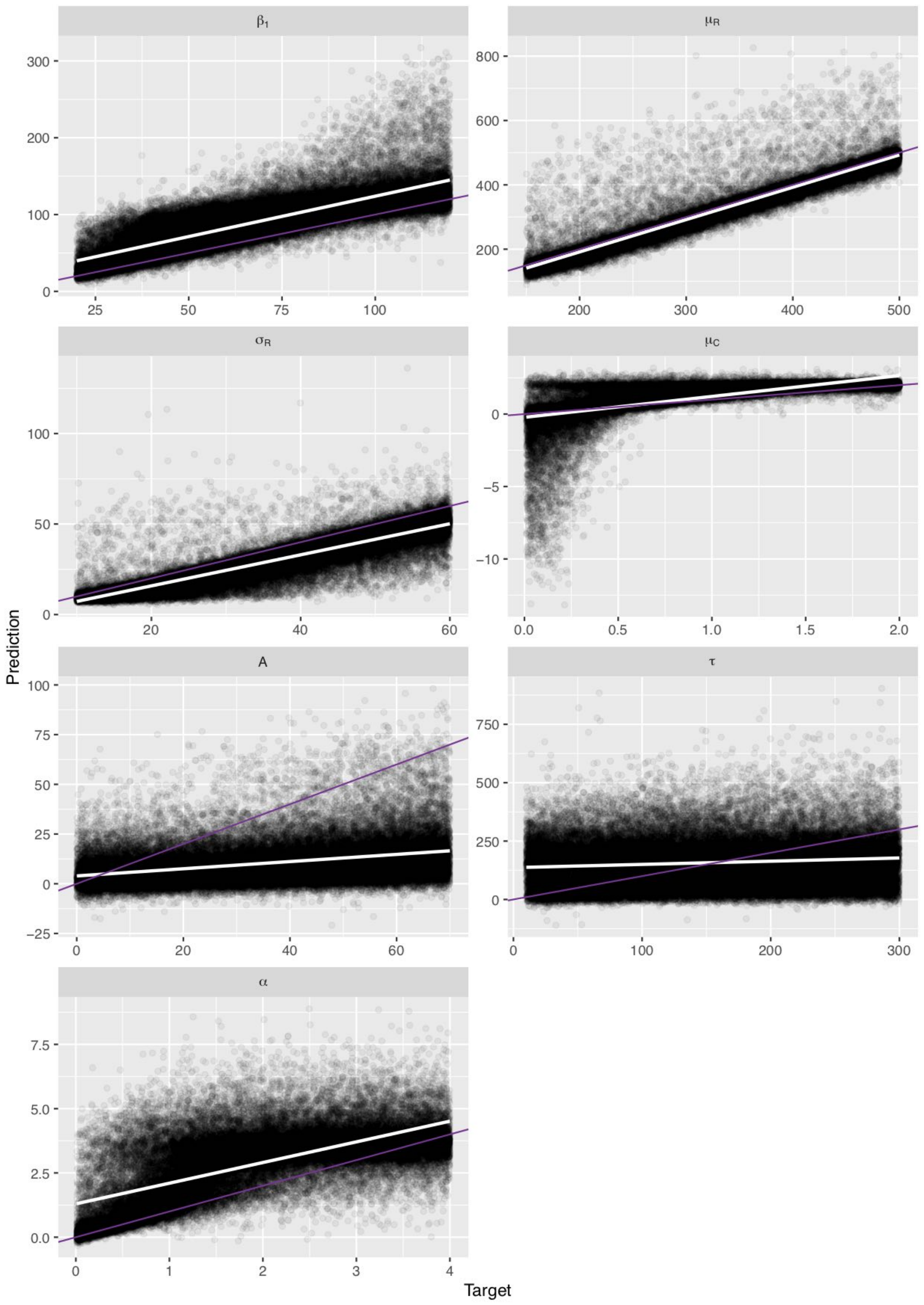


Figure 4.6: Correlation between predicted (y axis) and target (x axis) parameters from the diffusion model for conflict tasks for noisy (600 trials per RT distribution) data when the deep learning models were trained with data with large number of trials. White solid lines display fitted regression lines, while solid purple lines display identity lines.

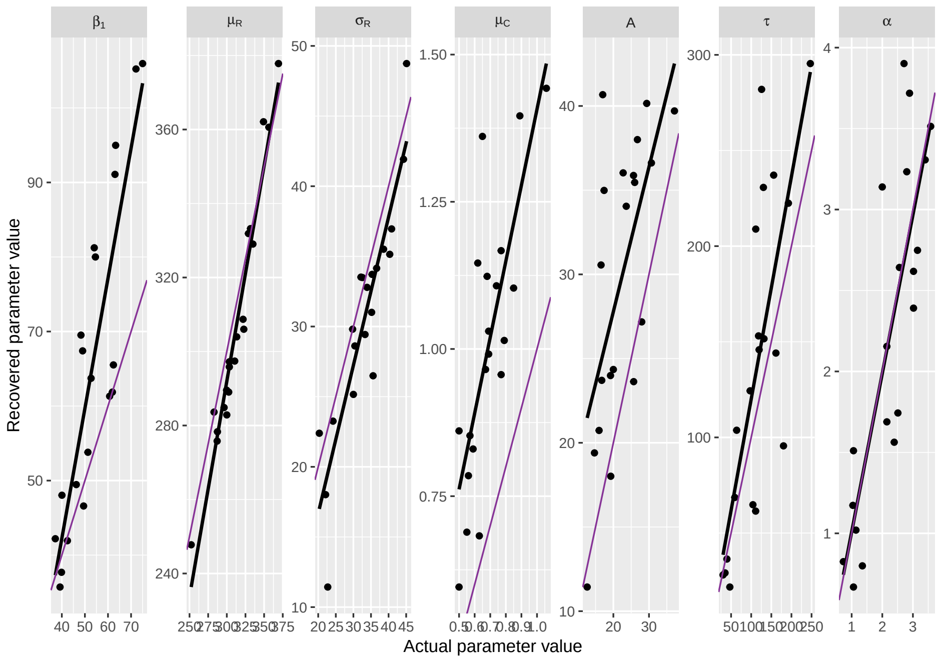


Figure 4.7: Correlation between predicted (y axis) and target (x axis) the diffusion model for conflict tasks parameters for noisy (600 points per distribution) input data, when the model was trained on data with large number of trials. Black solid lines display fitted regression lines, while solid purple lines display identity lines.

in table table 4.7, together with the KS values from each of the global optimization algorithm from recovering seven parameters. The table shows that deep learning was worse at recovering parameters for distributions with few trials when the models were trained on distributions with high number of trials compared to four global optimization algorithms. The average KS value for differential evolution was the smallest (0.131) compared to deep learning's value of 0.340. This difference was statistically significant ($t(19) = 10.37, p = 2.9 \times 10^{-9}$). Out of twenty test distributions, nineteen of them were recovered the best by differential evolution, while the remaining one was recovered best by the particle swarm optimization algorithm. This shows that deep learning models trained on datasets with large number of trials perform worse than optimization algorithms that are non-specific to input data.

When comparing correlation coefficients between actual and recovered parameters, deep learning performed much better at recovering parameters than differential evolution, as the difference between correlation between actual and recovered parameters (0.59 for differential evolution and 0.82 for deep learning) was statistically significant (Z=28, p=0.016). This is an interesting result, as looking at correlations between recovered and actual parameters indicates that deep learning outperforms differential evolution, however when RT distributions are constructed from recovered parameters, the similarities between actual RT distributions and recovered RT distributions are much greater with differential evolution than deep learning for RT distributions with small number of trials.

4.4 Conclusions

This chapter investigated whether deep learning algorithms can be employed to recover parameters from diffusion model for conflicts (Ulrich et al., 2015) when trained on data with large number of trials. The results showed that deep learning models are capable of recovering diffusion model for conflict tasks parameters almost perfectly, both when evaluated on 50,000 testing samples, and the twenty testing distributions from section 3, given that the testing data also contained large number of trials.

One reason why deep learning models performed so much better than any global optimization algorithm could be that seven models were trained in deep learning, each one predicting a different parameter from diffusion model for conflict tasks. In this way,

Table 4.7: KS comparisons noisy data.

Dataset	Random search	Particle swarm optimization	Differential evolution	Bayesian optimization	Deep learning
1	0.206	**0.132**	0.159	0.247	0.491
2	0.241	0.182	**0.154**	0.266	0.409
3	0.176	0.150	**0.113**	0.218	0.301
4	0.190	0.216	**0.118**	0.228	0.380
5	0.203	0.196	**0.138**	0.196	0.370
6	0.260	0.178	**0.132**	0.218	0.408
7	0.211	0.157	**0.120**	0.216	0.482
8	0.135	0.130	**0.127**	0.177	0.208
9	0.273	0.372	**0.120**	0.299	0.325
10	0.204	0.151	**0.126**	0.176	0.439
11	0.224	0.159	**0.117**	0.231	0.328
12	0.217	0.241	**0.141**	0.275	0.245
13	0.233	0.171	**0.130**	0.192	0.258
14	0.242	0.222	**0.179**	0.233	0.282
15	0.317	0.183	**0.136**	0.287	0.250
16	0.256	0.458	**0.125**	0.263	0.270
17	0.195	0.155	**0.122**	0.240	0.473
18	0.198	0.191	**0.116**	0.253	0.221
19	0.250	0.269	**0.128**	0.202	0.393
20	0.157	0.348	**0.111**	0.232	0.268
mean	0.219	0.213	**0.131**	0.232	0.340

there were no trade-offs between parameters from diffusion model for conflict tasks that could influence the performance of the deep learning models. Another possible explanation of the superiority of deep learning could be the sampling rate of the algorithm: deep learning is able to evaluate hundreds of thousands of reaction time distributions, while global optimization algorithms deal with only thousands of reaction time distributions. Moreover, the evaluation of the deep learning performance is not related to any method of comparing RT distributions, and as thousands of distributions are fitted at once, the algorithm can fit a few distributions poorly while not sacrificing the overall good fit. To investigate whether it is the former, a deep learning model could be compiled that predicts all seven diffusion model for conflict tasks parameters at once.

A huge advantage of deep learning over any global optimization algorithm is the speed of estimating parameters. Even with time-consuming hyperparameter optimization, training of the deep learning models takes hours instead of days. However, once the deep learning models are trained, predicting parameters from any number of reaction time distributions takes seconds, while global optimization algorithms take hours or days per reaction time distribution. In theory, once a model is trained, it then can be used by anyone to estimate parameters with a known margin of error.

As mentioned before, hyperparameter search is the most time consuming aspect of deep learning. Computing huge datasets that the deep learning model can use for training is also a computationally expensive task. This is however the result of deep learning having never been applied to parameter recovery from sequential sampling models. As more researchers start training deep learning models in this area, approximate model architectures will become more available, as will training datasets.

The biggest issue with using deep leaning in parameter recovery is that because it has not been done before, the model architecture is quite difficult to determine. The way the model architecture was chosen in this chapter (narrow and long) was because it seemed to produce good enough R^2 values with most parameters from diffusion model for conflict tasks, but this does not mean that better architectures do not exist. For more complicated parameters, like amplitude of automatic activation, and time to peak automatic activation, wider models could be of benefit.

Conversely, it might be that there is not enough signal in input features to predict some parameters from diffusion model for conflict tasks. This could be the result of feature engineering: the signal from parameters like shape of starting point distribution or time to peak automatic activation could just not be captured well by the features that were engineered for deep learning model inputs, and there could be better input features.

Feature engineering is a huge issue for deep learning in itself. The advantage of deep learning over shallow machine learning methods is that deep learning does not require feature engineering, as technically the models should be able to take in raw data to predict outcomes. This is quite complicated with the way the raw data is collected in decision making models. The number of trials (raw data inputs) is variable, the features (raw data trials) do not follow a specific order, so there is no temporal sequence, and that makes using raw trials as input features very difficult. This could be solved by introducing temporal dependencies in the decision making models.

Even though the recovery of parameters from data with large number of trials is nearly perfect, the recovery from noisy data when the model is trained with data with large number of trials is not so good, though still not worse than any global optimization algorithm. Therefore models trained with data with large number of trials will generally not offer good parameter predictions from experimental data, which usually has low trial numbers per condition (and hence noisy input features). Therefore other approaches will need to be used. There are three possible ways to improve parameter recovery from noisy data: first, retraining models, including hyperparameter optimization, using noisy data as deep learning model inputs; second, using transfer learning, adding a few hidden layers on the already trained models, then only training those; third, using autoencoders to de-noise noisy data. All of these will be investigated in the next chapter.

To conclude, there is no reason why deep learning should not be used over any global optimization algorithm for parameter recovery from the diffusion model for conflict tasks, or any other sequential sampling model. Deep learning is able to recover parameters better than global optimizers, it is able to do it quicker, it can take advantage of huge data availability, it can provide error of predictions, thousands of RT distributions can be fitted at once, and finally, as computational power increases, these calculations will become even more sophisticated and fine-tuned. Finally, decision making models can become more complex to account for human behaviour more precisely (like adding time-dependencies that we know exist in human behaviour, like post-error slowing, long-term slow down of behaviour, etc), yet deep learning will be able to adapt to that without much trouble.

5 Deep learning for parameter recovery using noisy input data

5.1 Introduction

Deep neural networks are very powerful at learning representations between input and output data when presented with input data that contains little noise, as seen in section 4. Data that is prevalent in the real world however contains a lot of noise in both the inputs and the output labels. This is also true in mathematical models of decision making. In section 4, we have seen that deep neural networks could recover parameters from the diffusion model for conflict tasks very well when the deep learning models were trained and tested on data with large number of trials (which contain little noise in RT distributions, which are then used to create input features). However, when the same deep learning models were asked to predict parameters from the diffusion model for conflict tasks from data that contains a much smaller number of trials, more akin to data that could be obtained from studies with human participants, the performance of the deep learning models dropped significantly, especially for more difficult to recover parameters like time to peak automatic activation and shape of starting point distribution.

Most of the applications of the diffusion model for conflict tasks and sequential sampling models are done on experimental data where there is considerable noise in RT distributions due to limited number of trials that are collected in experiments with humans. The number of trials per experimental condition is even lower when more challenging populations are tested, like children (Ambrosi et al., 2019) or people with Parkinson's disease (Servant, van Wouwe, Wylie, & Logan, 2018). Therefore in order to consider deep neural networks as a good method for parameter recovery from the diffusion model for conflict tasks, we need to evaluate whether we can train the deep learning models to perform well with input features computed from reaction time distributions that contain much fewer trials (more noise).

Fortunately, this problem is not unique to the diffusion model for conflict tasks. Noisy input features are also prevalent in speech processing (Lu, Tsao, Matsuda, & Hori, 2013), image recognition (Vincent, Larochelle, Lajoie, Bengio, & Manzagol, 2010), biomedical signal processing (Min, Lee, & Yoon, 2017), health care data (Miotto et al., 2017), among other areas, and deep learning has been applied to all of them relatively successfully. There are various methods that can be used to overcome noise in input datasets, and a few will be explored in this chapter to see how successfully can the parameters from the diffusion model for conflict tasks be recovered from reaction time distributions with a small number of trials.

The most straightforward approach to recovering parameters from noisy input data would be to train deep learning models using noisy input data. This is usually a hard task in other areas where deep learning methods are applied, as large samples of labelled data are hard to come by, especially when the input data is noisy. If input data contains more noise, the deep neural networks might need to increase in size in order to separate the signal from the noise in the input data, and subsequently more input samples would be needed to account for the increase in trainable parameters. This however is not a problem for mathematical models of decision making, because true-labelled input data can be computed very easily. The method for training the deep learning models would be identical to section 4.2, except that now we have an idea of approximate areas for the hyperparameter search, therefore the search should be easier. We can also presume that more hidden layers will be needed to account for noise in the input data.

A second approach involves using the information already present in deep learning models trained with clean data in section 4. This approach is called transfer learning, and is very prevalent in computer vision. In computer vision, it is hard to come by a huge amount of training samples for each specific vision problem, given that the architectures usually employed in image recognition and classification tasks, convolutional neural networks, contain millions of trainable parameters. For example, the ImageNet classification model by Krizhevsky et al. (2012), was trained on 15 million labelled images and contained over 60 million trainable parameters. However, irrespective of the specific problem that the deep learning models are trying to solve, low level visual properties extracted by neural networks are the same, therefore they can be transferred from already trained networks to new problems, with only a few additional layers required for problem-specific solution. Transfer learning therefore allows researchers to utilize much smaller sample sizes for training, as shown in Oquab, Bottou, Laptev, and Sivic (2014). Transfer learning is not unique to computer vision, as Google have just recently released a pre-trained network model for natural language understanding (Devlin, Chang, Lee, & Toutanova, 2018), which allows creation of a wide range of models for tasks such as question answering and language inference with fine tuning only a single additional layer, which means that much smaller input samples are sufficient for state-of-the art model performance.

The same principles of transfer learning can be applied to dealing with noisy input data. If we presume that input features that come from reaction time distributions with large number of trials contain both low level and high level features that are extracted from input data and used for predicting parameters from diffusion model for conflict tasks, some of these extracted features may not be affected by noise and thus still valuable when input data is noisy (comes from reaction time distributions with low number of trials). Therefore we can potentially use the models trained with clean input data in section 4 to transfer learning to models when the input data is noisy. In this way, the transfer learning models will have layers with frozen (non-trainable) weights with additional layers with trainable weights to specifically deal with noise in the input data. As the number of trainable weights in transfer models is much lower than in models that are trained from scratch, they should be less prone to overfitting, where the models learn only the mapping between input and output features in the training samples and do not generalize at all to testing samples.

The final approach for dealing with noisy input data involves processing of the input data before it is being passed into the deep learning models. Noise in input data is prevalent in all fields, as the way data is acquired tends to add noise to the inputs. In vision problems, images can include superimposed objects, such as writing, that needs to be ignored in classification problems, while in speech recognition, samples of speech patterns can come with background noise, recording distortions, or accents that need to be dealt with. In order to map noisy data to clean input features, a specific architecture of deep neural networks can be used. Even though unsupervised deep learning methods are not as commonly used in real-world problems, they are very powerful in extracting useful features from unlabelled data, detecting and removing input redundancies, and preserving essential aspects of the data (Masci, Meier, Cireşan, & Schmidhuber, 2011). An encoder is a type of unsupervised deep learning architecture that learns representation (encoding) for a set of data. Most encoders are also stacked with decoders, so that the encoding of the data can be decoded back into input data. Such encoder-decoder architectures are called autoencoders (Huang, Boureau, LeCun, et al., 2007).

Autoencoders consist of encoder and decoder components. In the encoder part, layers of neural networks are stacked in such a way that deeper layers have progressively lower number of neurons compared to the previous layer. Because of this architecture, the input data is forced into a smaller dimensionality of representations, which allows the encoder to ignore small disturbances (noise) in the data. Then the encoded data can be used as input into a decoder model, which translates the data encoded in smaller dimensions (than the original input data) back to the same dimensions as the input data. The decoded data then contains no disturbances (noise) but only useful features (or signal). Even though autoencoder models seem to propose a good generic data compression algorithm, in reality it is not that useful as data-specific compression algorithms perform much faster and require fewer resources than training deep learning networks. However the autoencoders have been applied extensively to cleaning noisy input data, especially in computer vision (Xie, Xu, & Chen, 2012; Vincent et al., 2010; Masci et al., 2011) and speech problems (Lu et al., 2013). In such models, noisy data is used as input into encoders, and cleaned data as output from decoders. This chapter will investigate whether autoencoder models can be used to reduce the noise of reaction time distributions with low number of trials, which could then be used as inputs into deep learning models trained on clean data (reaction time distributions with large number of trials).

As shown in section 4, deep learning models are very capable of recovering parameters from diffusion model for conflict tasks, however they do not perform that well when the input data into the models comes from reaction time distributions with low number of trials. This is a very common situation when trying to apply mathematical models of decision making to experimental data obtained from studies with humans, as the number of trials per condition is limited due to experimental time. Therefore for deep learning to be a useful technique for parameter recovery, the models need to be able to deal well with noisy input data. This chapter will investigate whether transfer learning, autoencoders, or training deep learning models with noisy data from scratch can help improve parameter recovery from noisy data as compared to deep learning models trained with clean data.

5.2 Training deep learning models with noisy data

5.2.1 Data

In order to maintain consistency of methods and data in section 4, the dataset to train models with noisy data was computed in the identical way as the noisy test dataset in section 4.2. This was done by subsampling the trials that comprised reaction time distributions with large number of trials. For each sample (reaction time distribution) with 15,000 trials, a random sample of 200 trials was chosen for each condition (congruent, incongruent, and neutral), thus producing reaction time distributions of 600 trials and introducing variability (noise) in the input features. Due to subsampling of the clean training dataset, the data that made the noisy training dataset was identical to that of the clean dataset, with the exception of variability in reaction time distributions. The targets (output) features remained identical as those in section 4.2. The test dataset was also identical to the one used in section 4.2. As a reminder, the parameters from the diffusion model for conflict tasks were sampled from a uniform distribution, limits and distribution of parameter values are described in figure 4.1.

The inputs into the deep learning models trained with noisy data were 92 input features computed for each RT distribution. All the features are listed in section 4.2. The methods for computing the input features were identical to those described in section 4.2. There was a slight difference between using reaction time distributions with a large number of trials and reaction time distributions with a small number of trials for computing input features, in that some features could result in Not-a-Number/non-valid-numeric (NaN) [6] values when computed from reaction time distributions with low number of trials (e.g. if there were no incorrect responses among the 200 trials in congruent condition, the mean RT for incorrect trials in incongruent condition would be a NaN due to zero-division). NaN features are not acceptable to deep learning models, therefore they were replaced with zeros in input features.

5.2.2 Hyperparameter search

The hyperparameter search was easier for training deep learning models with noisy data, because similar architectures as with training models with clean data could be applied. For this reason, hyperparameter values were inspired by the final models in section 4. The learning rate was not optimized for deep learning models trained with noisy data, and was set to 0.0001, same as in section 4.2. Similarly, L1 regularization was not optimized, but instead set to 0.0001 for all layers. Finally, Dropout was also not included in the new deep learning models trained with noisy data, due to narrow model architectures.

In order to deal with noise in the input data, two more hidden layers were added to the

[6]A common example of this is when an attempt is made to divide by zero

hyperparameter search. Even though it seems like a small number, two extra hidden layers would increase the number of trainable weights by 50%. As the number of training samples would remain the same as in deep learning models trained with clean data, increasing the number of hidden layers even more could potentially hinder the generalization of the deep learning models to test data due to increasing imbalance between number of trainable parameters and number of training samples. The choices for the hyperparameter search were:

- number of hidden layers: [4, 6]
- size of each hidden layer (chosen independently): [64, 128, 256, 512]

The hyperparameter search was performed in the same way as in section 4.2, with five-fold cross-validation over the training dataset. The models were evaluated after running for 25 epochs five times for each different testing split. The number of different model evaluations was 25. The resulting best deep learning model for each parameter from diffusion model for conflict tasks is presented in table 5.1, together with R^2 values for each model evaluated on the full training dataset. As is evident from the table, for all parameters from diffusion model for conflict tasks (except diffusion rate for controlled process and variability of non-decision time), deeper architectures with six hidden layers resulted in better performing deep learning models. What is also evident from the table is preference for larger number of neurons per hidden layer, revealing the need for higher number of trainable weights to improve model performance. Finally, there appears to be a similarity between the architectures of deep learning models for different parameters from diffusion model for conflict tasks, indicating that a single deep learning model might be sufficient at predicting all model parameters at once.

5.2.3 Results

<u>Overall model architecture</u> Seven different deep learning models were trained, one for each parameter from the diffusion model for conflict tasks as the output. The same input data was used in all seven models. The model architecture consisted of:

- sequential model
- dense layers

Table 5.1: Optimized architectures of deep learning models trained with noisy input data. If only four hidden layers were used, Layer 5 and Layer 6 sizes are reported as ”-”. The R^2 values come from evaluating the deep learning model chosen by HYPERAS on 150,000 samples of the training dataset.

Parameter	R^2 value	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6
β_1	0.897	256	512	512	256	512	512
μ_R	0.994	256	512	512	256	512	512
σ_R	0.950	256	512	512	512	-	-
μ_C	0.902	256	512	64	64	-	-
A	0.826	256	512	512	256	512	512
τ	0.600	256	512	512	256	512	512
α	0.788	256	512	512	256	512	512

- bias term included in all layers
- ReLU activation in all hidden layers
- RMSprop optimization function with parameters $\rho = 0.9$, $\epsilon = 1e - 08$, decay=0.0
- batch size of 75 samples
- L1 regularizer in each layer, 0.0001
- mean square error as model error
- learning rate of 0.0001

Each model was trained for 25 epochs. All final trained models are available on https://github.com/SolVG/pyCDM. To evaluate the performance of the predictive models, R^2 values between outputs and targets, mean absolute deviance values from the target, and mean error values between outputs and targets, were also calculated in each epoch. These were not used to train the models in any way. The history of training deep learning models for each parameter from the diffusion model for conflict tasks is presented in figure 5.1. As can be seen in the figure, most deep learning models minimized the loss function well. Some overfitting could be observed for upper threshold, drift rate for controlled process, and shape of starting point distribution parameters, indicating that they may benefit from further regularization of weights, Dropout layers, or early stopping callbacks [7]. Two parameters, amplitude of automatic activation and time to peak automatic activation, could not be recovered at all as the models failed to minimize the loss function. Even though R^2 of the training dataset for these parameters was above zero, the performance of the test dataset was at zero, meaning that the models only learned to map input features to outputs without learning any representations in the data.

The results of evaluating the performance of the deep learning models in recovering parameters from the diffusion model for conflict tasks from the test dataset are reported in table 5.2. The models were evaluated by calculating the Pearson correlation coefficients between actual and recovered parameter values from the test dataset. The correlations are displayed in figure 5.2. The results show that all deep learning models, except for amplitude of automatic activation, and time to peak automatic activation models, had high positive correlations between actual and recovered parameter values. Surprisingly, given the huge generalization errors shown in figure 5.1, even amplitude of automatic activation and time to peak automatic activation resulted in non-zero correlations between actual and recovered parameter values. The results show that all non-impulsivity parameters from diffusion model from conflict tasks can be recovered with deep learning models trained on reaction time distribution data with low number of trials.

Table 5.2: Correlation coefficients and p values between actual parameter values and recovered parameter values from deep learning models trained with and tested on data with low number of trials.

Parameter	Correlation coefficient	p value
β_1	0.909	$p = 2.2 \times 10^{-16}$
μ_R	0.988	$p = 2.2 \times 10^{-16}$
σ_R	0.953	$p = 2.2 \times 10^{-16}$
μ_C	0.861	$p = 2.2 \times 10^{-16}$
A	0.286	$p = 2.2 \times 10^{-16}$
τ	0.280	$p = 2.2 \times 10^{-16}$
α	0.750	$p = 2.2 \times 10^{-16}$

The performance of the deep learning models was also evaluated on 20 test datasets used in section 3. The twenty datasets contained reaction time distributions of 600

[7] in deep learning, not completing the full number of pre-specified training epochs if the learning plateaus is acceptable and widely used, which is commonly implemented by allowing the model itself to stop the learning process via early stopping callbacks

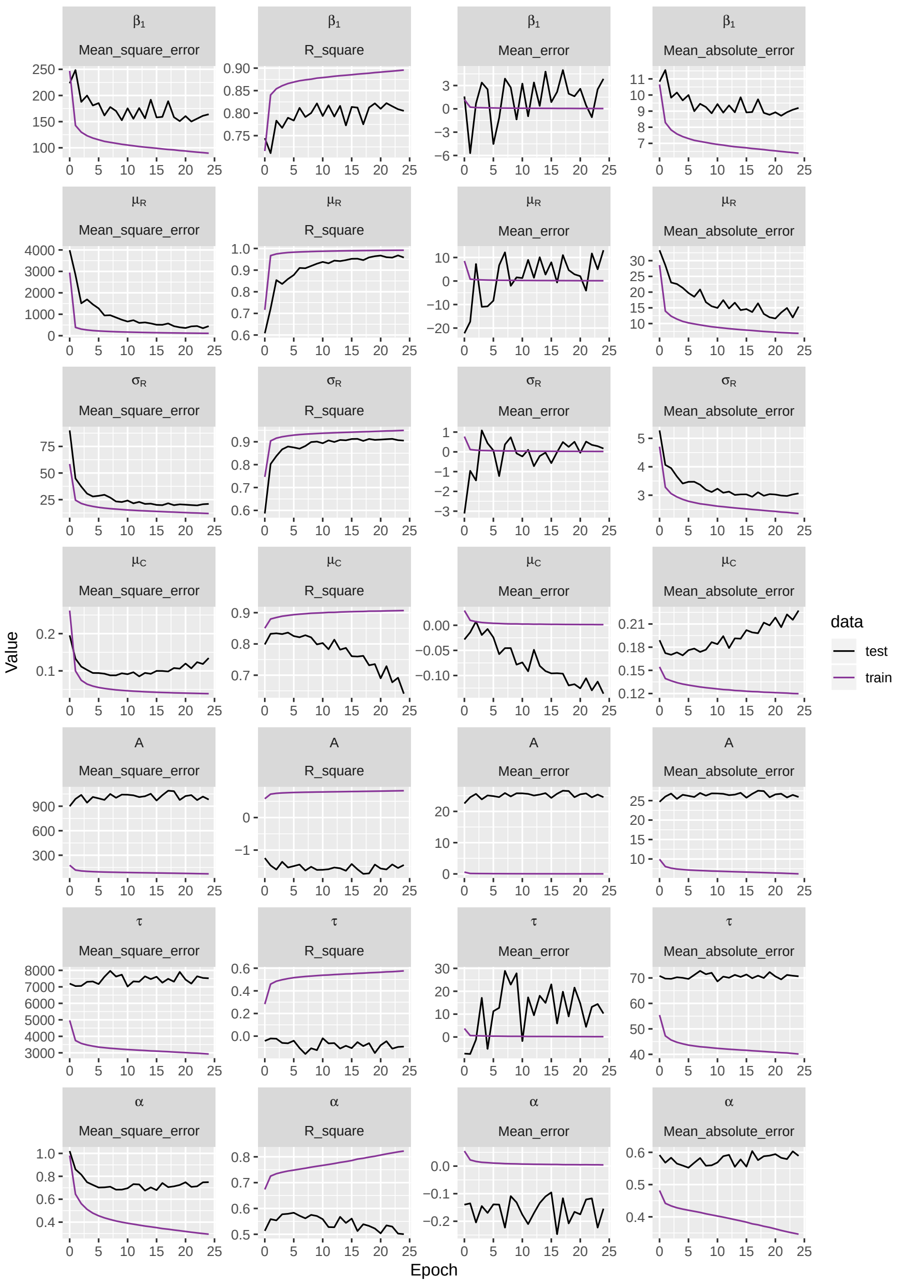


Figure 5.1: History of deep model training with noisy data for predicting parameters from diffusion model for conflict tasks of four measures: mean square error (used as a loss function, outer left column), R^2 (inner left column), mean error (inner right column), and mean absolute error (outer right column). The seven parameters from diffusion model for conflict tasks are reported in seven rows of the figure. The purple line represents training data, while the black line represents evaluation of the test data.

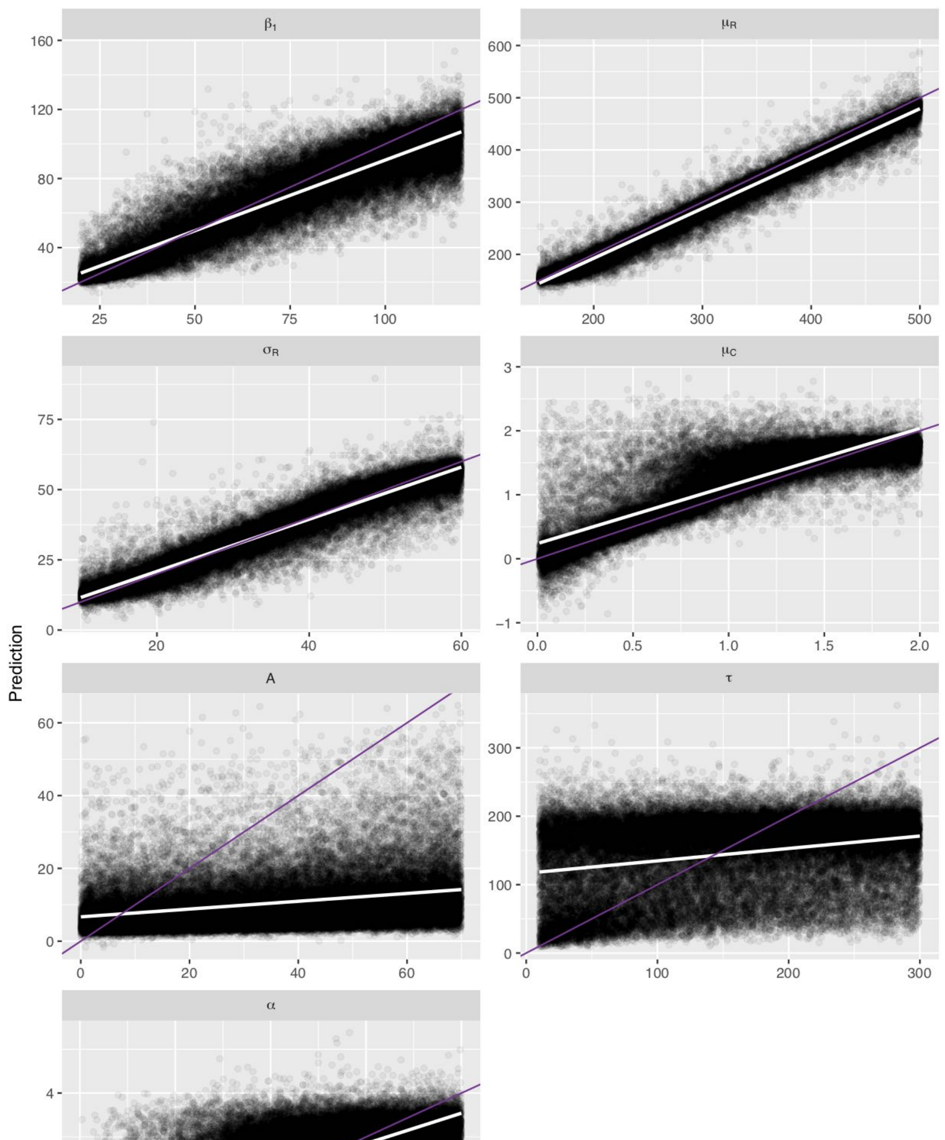


Figure 5.2: Actual and predicted parameter values from diffusion model for conflict tasks from deep learning models trained with noisy data and tested on 50,000 test distributions. Purple lines indicate identity lines, while white lines display best fit lines between actual and predicted values.

trials, making them noisy. The performance of the deep learning models in recovering parameters from these twenty distributions is reported in table 5.3 and the correlations between actual and recovered parameter values are displayed in figure 5.3. The results show that all parameters could be recovered well from noisy realistic distributions, even the "impulsivity" automatic activation parameters. This indicated that deep learning models trained on noisy reaction time distribution data are capable of recovering parameters from diffusion model for conflict tasks.

Table 5.3: Correlation coefficients (and p values) between actual and recovered parameter values from twenty test distributions, when the deep learning models trained with noisy data.

Parameter	Correlation coefficient	p value
β_1	0.935	$p = 1.6 \times 10^{-9}$
μ_R	0.942	$p = 5.7 \times 10^{-10}$
σ_R	0.989	$p = 2.2 \times 10^{-16}$
μ_C	0.870	$p = 6.2 \times 10^{-7}$
A	0.724	$p = 3.1 \times 10^{-4}$
τ	0.710	$p = 4.6 \times 10^{-4}$
α	0.710	$p = 4.6 \times 10^{-4}$

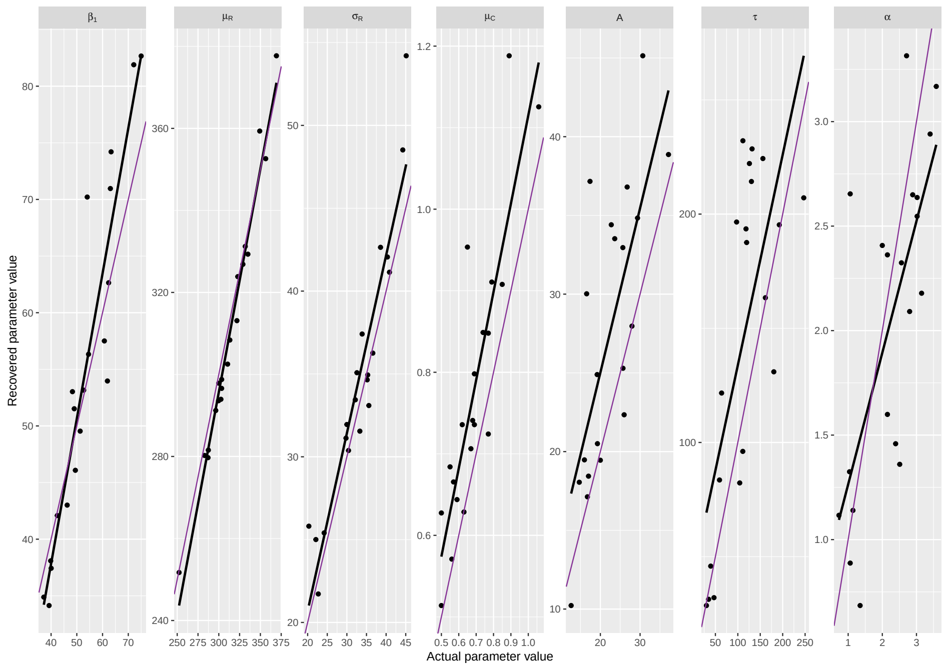


Figure 5.3: Correlation coefficients between actual (x axis) and recovered (y axis) parameter values from twenty test distributions, when the deep learning models trained with noisy data (reaction time distributions with 600 trials). Black solid lines display fitted regression lines, while solid purple lines display identity lines.

5.3 Transfer learning from deep learning models trained on clean data

5.3.1 Model architecture

The deep learning models trained in section 4 were used as the basis for transfer learning models. The last two layers (the output layer, and the last hidden layer of the network) of each of the deep learning models were removed. The weights of the remaining layers were frozen, so that the weights in these layers would not be adjusted during the model training process. Then, three additional hidden layers with trainable weights were added to the models, as well as final output layer. The three hidden layers contained 256, 512, and 512 neurons in this order. Finally, a linear output layer was added. Then the resulting deep learning models were trained with input data that was identical to the data used in section 5.2. The testing data was also identical to that in the previous section. The only difference between training the deep learning models in this section was that the number of epochs was limited to 20, to account for only three hidden layers that had trainable weights.

5.3.2 Predicting the diffusion model for conflict tasks parameters

The history of training of the transfer models is displayed in figure 5.4. The figure shows that all seven models trained well over the twenty epochs, however all models displayed overfitting to varying degree. This indicates that the models architectures were not optimal for recovery of parameters from unseen test data.

The evaluation of the transfer models in recovery of the parameters from the diffusion model for conflict tasks is reported in table table 5.4. The correlations between actual and recovered parameter values are shown in figure 5.5. The results show that all model parameters were recovered reasonably well by deep learning models pre-trained with clean data, except for automatic activation parameters. However, the performance of the transfer models was not as good as the performance of the deep learning models trained with noisy data from scratch. This was true for all parameters except the amplitude of automatic activation, which was recovered better by transfer models than models trained with noisy data. This could indicate that pre-training of the deep learning model for amplitude of automatic activation preserved some low-level features in data that could be picked up from noisy data.

Table 5.4: Correlation coefficients and p values between actual parameter values and recovered parameter values from transfer models trained with and tested on data with low number of trials.

Parameter	Correlation coefficient	p value
β_1	0.876	$p = 2.2 \times 10^{-16}$
μ_R	0.974	$p = 2.2 \times 10^{-16}$
σ_R	0.922	$p = 2.2 \times 10^{-16}$
μ_C	0.758	$p = 2.2 \times 10^{-16}$
A	0.317	$p = 2.2 \times 10^{-16}$
τ	0.202	$p = 2.2 \times 10^{-16}$
α	0.657	$p = 2.2 \times 10^{-16}$

The performance of the transfer models was also evaluated on the twenty test datasets, same as in section 5.2. The results of correlations between actual and recovered parameters from the twenty test distributions are reported in table 5.5 and shown in

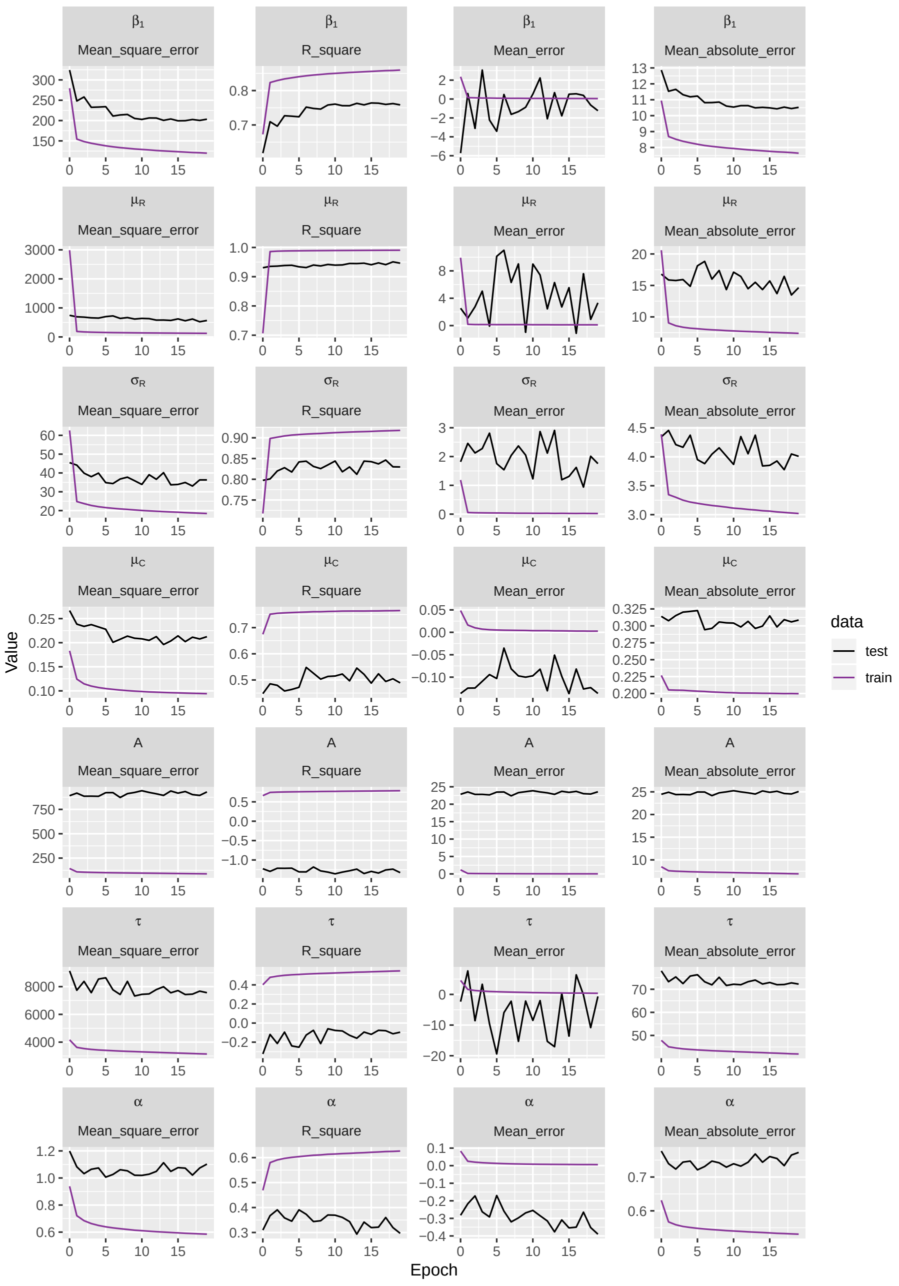


Figure 5.4: History of transfer model training with noisy data for predicting parameters from diffusion model for conflict tasks of four measures: mean square error (used as a loss function, outer left column), R^2 (inner left column), mean error (inner right column), and mean absolute error (outer right column). The seven parameters from diffusion model for conflict tasks are reported in seven rows of the figure. The purple line represents training data, while the black line represents evaluation of the test data.

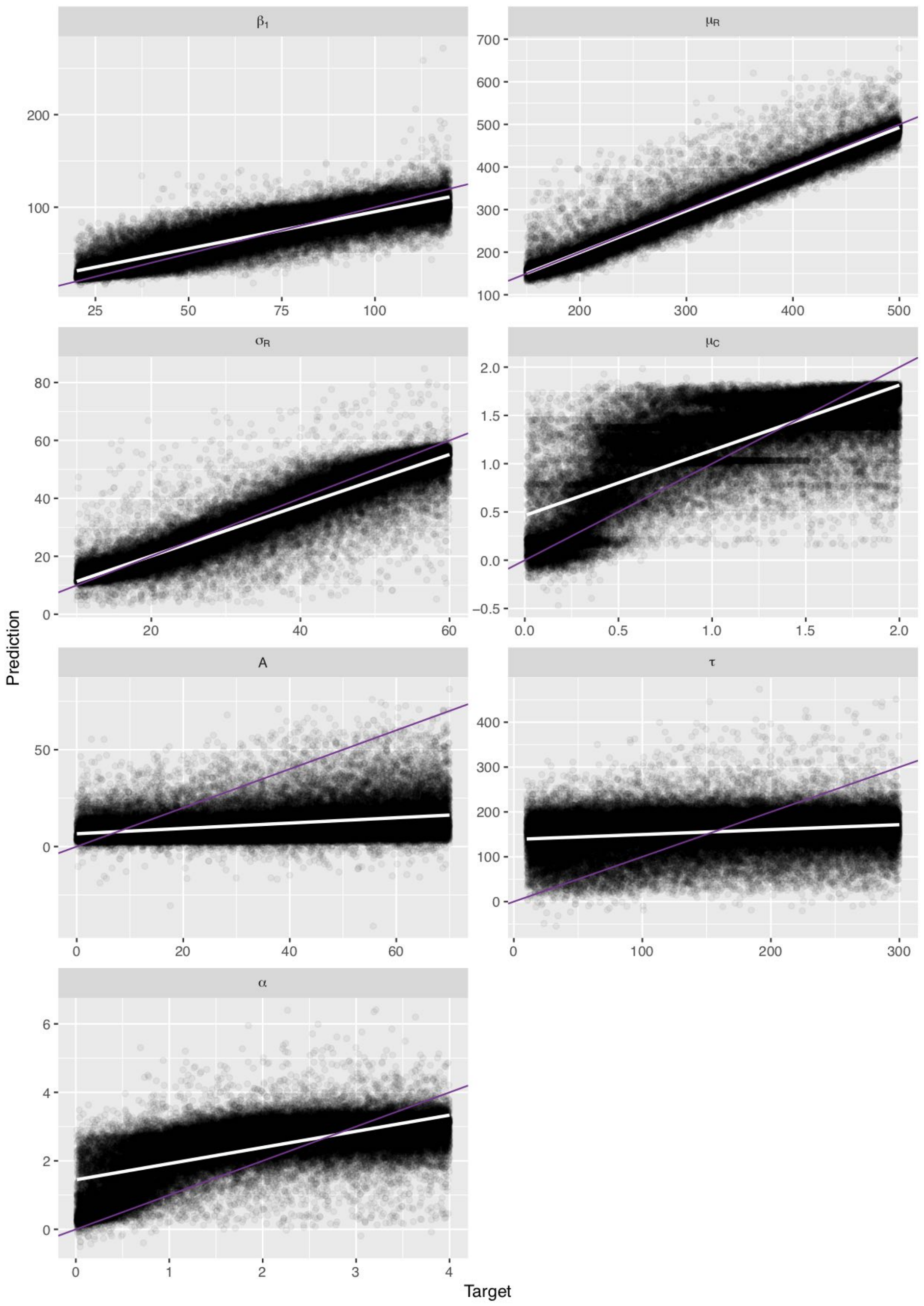


Figure 5.5: Actual and predicted parameter values from the diffusion model for conflict tasks from transfer learning models trained with noisy data and tested with 50,000 test distributions. Purple lines indicate identity lines, while white lines display best fit lines between actual and predicted values.

figure 5.6. The table shows that the correlations between actual and recovered parameters were high for all parameters from diffusion model for conflict tasks, except for drift rate for controlled process. There seems to be some differences in recovery of parameters between transfer models and the deep learning models trained with noisy data, however they are minor. The two automatic activation parameters and the shape of starting point distribution parameter appear to be recovered better by transfer models.

Table 5.5: Correlation coefficients and p values between actual parameter values and recovered parameter values from deep learning models pre-trained on data with high number of trials and tested on twenty test distributions with low number of trials.

Parameter	Correlation coefficient	p value
β_1	0.882	$p = 2.7 \times 10^{-7}$
μ_R	0.975	$p = 2.8 \times 10^{-13}$
σ_R	0.913	$p = 1.9 \times 10^{-8}$
μ_C	0.105	$p = 0.659$
A	0.809	$p = 1.6 \times 10^{-5}$
τ	0.744	$p = 1.7 \times 10^{-4}$
α	0.869	$p = 6.5 \times 10^{-7}$

5.4 Autoencoders for data denoising

5.4.1 Data

The data for autoencoders was the same training data as used in 4 and in 5.2. The reaction time distributions in both datasets were theoretically the same, as they came from the same sets of parameters of the diffusion model for conflict tasks. The only difference was the number of trials per distribution (15,000 for distributions with high number of trials and 600 for distributions with low number of trials). Because of the difference in trial numbers, there was some variation in reaction time distributions, which added variability to 92 input features. The 92 input features computed from the datasets with large number of trials were considered as features with low noise, whereas the input features from the dataset with low number of trials contained more noise, as individual trials had more of an effect on the overall reaction time distribution shape, and the resulting input feature measures. As in 5.2, input features from datasets with

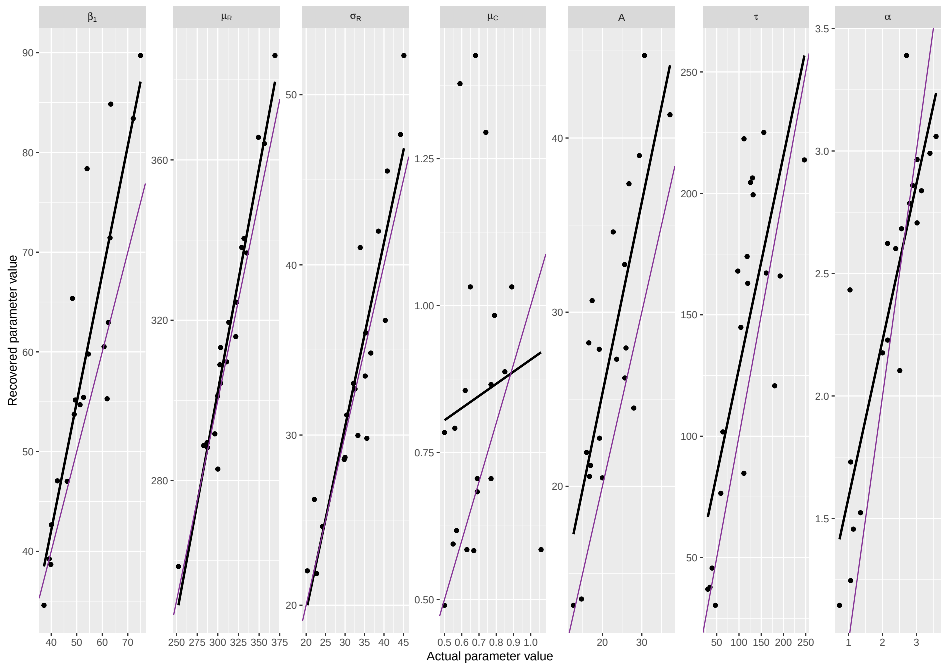
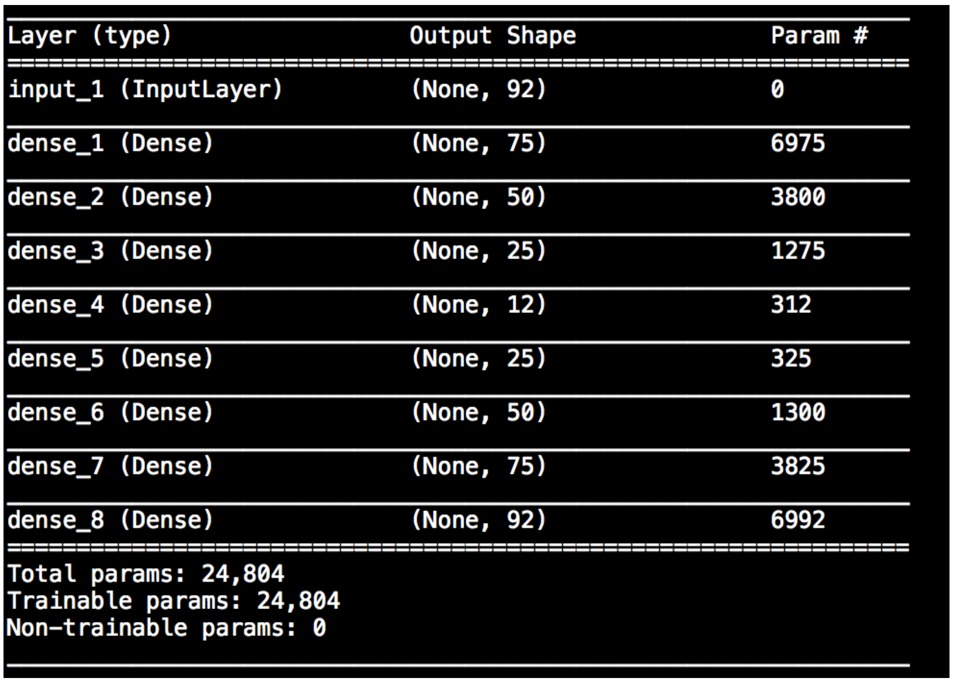


Figure 5.6: Correlation between predicted (y axis) and target (x axis) the diffusion model for conflict tasks parameters for noisy (600 points per distribution) input data, when the deep learning models were pre-trained with data with large number of trials. Black solid lines display fitted regression lines, while solid purple lines display identity lines.

low trial numbers that were NaNs were replaced with zeros to be compatible with the deep learning models.

5.4.2 Autoencoder architecture

The input into the autoencoder model were the features generated from the dataset of reaction time distributions with low number of trials. The output of the autoencoder model were the features generated from reaction time distributions with large number of trials. The autoencoder followed an encoder-decoder pattern, where input features were compressed to smaller and smaller dimensionality, and then decompressed back into the same dimensionality as inputs. As the number of input features in this dataset was 92, the autoencoder was created with four encoding layers and four decoding layers, going from 92, to 75, to 50, to 25, and finally to 12 features, and then all the way back from 12 to 25, to 50, to 75, and finally back to 92 features. The model architecture is summarized in figure 5.7. This architecture was selected from manual hyperparameter search, as it performed better (returned higher R^2 values) than architectures with more compression (encoding dimensions lower than 12), and also as compared with architectures with more encoding and decoding layers, as they did not bring any improvement in R^2 values compared to architecture with four encoding and decoding layers.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 92)                0
_________________________________________________________________
dense_1 (Dense)              (None, 75)                6975
_________________________________________________________________
dense_2 (Dense)              (None, 50)                3800
_________________________________________________________________
dense_3 (Dense)              (None, 25)                1275
_________________________________________________________________
dense_4 (Dense)              (None, 12)                312
_________________________________________________________________
dense_5 (Dense)              (None, 25)                325
_________________________________________________________________
dense_6 (Dense)              (None, 50)                1300
_________________________________________________________________
dense_7 (Dense)              (None, 75)                3825
_________________________________________________________________
dense_8 (Dense)              (None, 92)                6992
=================================================================
Total params: 24,804
Trainable params: 24,804
Non-trainable params: 0
_________________________________________________________________
```

Figure 5.7: Summary of autoencoder model architecture.

5.4.3 Predicting clean input data

To assess whether autoencoders help to denoise data, we first need to evaluate the relationship between data with large number of trials and data with small number of trials. To allow easier comparison with deep learning models, we are going to use R^2 value. When deep learning models are trained, the data used in training is scaled so that each feature has the same mean and standard deviation. This is especially important for autoencoders, because if the features are not scaled, then the models will prioritize features with larger numerical values over features with smaller numerical values. Therefore the comparisons are going to be made between scaled inputs.

R^2 calculations between input (data with low number of trials) and output (data with large number of trials) were implemented using SCI-KIT LEARN package in PYTHON. The R^2 between the noisy and clean scaled input features was 0.71.

The autoencoder model was trained for 100 epochs. The training history is reported in figure 5.8. The final R^2 value between clean input features and decoded input features was 0.42. This indicates that autoencoder could not improve the relationships between clean and noisy input data and therefore was not useful for recovering parameters from noisy data.

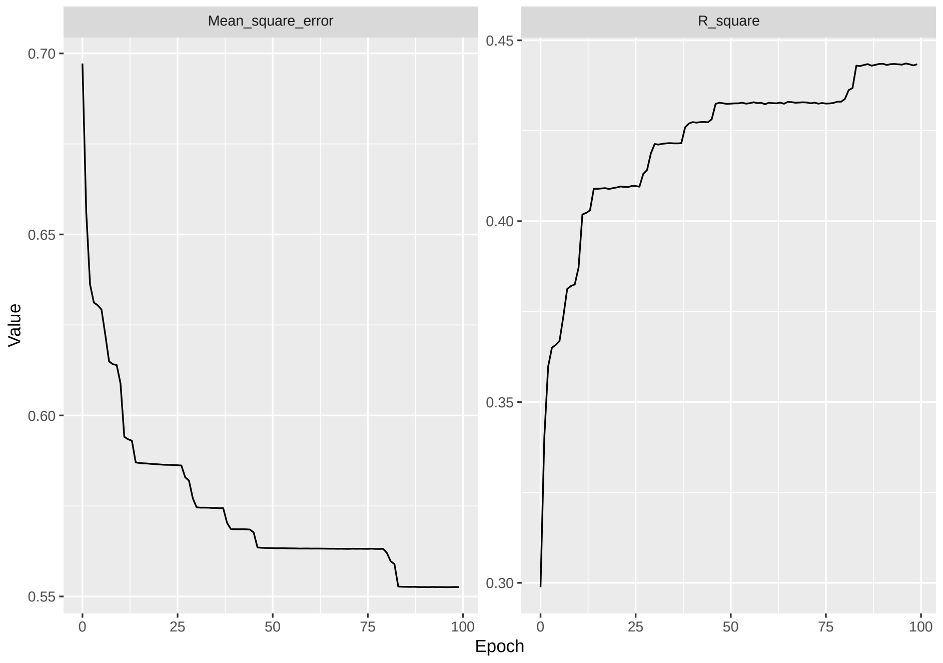


Figure 5.8: History of autoencoder model training.

5.5 Recommendations for use with experimental data

In order to see which method should be used when recovering diffusion model for conflict tasks parameters from reaction time distributions with low number of trials, we can compare correlations between recovered and actual parameter values from different methods. The four methods examined were differential evolution, deep learning models trained with clean data, deep learning models trained with noisy data, and transfer models. To allow comparison with differential evolution, which is a global optimization algorithm that resulted in the best performance for parameter recovery in section 3, correlations between actual and recovered parameter values from twenty test distributions were be inspected. The correlations are reported table 5.6. The highest correlation value for each parameter from diffusion model for conflict tasks is reported in bold. The table shows differential evolution does not recover any of the parameters better than any deep learning method. Out of the three deep learning methods, neither came as a clear winner, however the deep learning model trained with noisy data recovered three out of seven parameters the best.

Table 5.6: Comparison of different methods for parameter recovery from diffusion model for conflict tasks as evaluated on 20 test distributions. The best method for each parameter is displayed in bold.

Parameter	Differential evolution	Deep leaning - clean data	Deep leaning - noisy data	Deep leaning - transfer
β_1	0.61	0.875	**0.935**	0.882
μ_R	0.96	**0.976**	0.942	0.975
σ_R	0.77	0.903	**0.989**	0.913
μ_C	0.23	0.768	**0.870**	0.105
A	0.58	0.633	0.724	**0.809**
τ	0.71	**0.766**	0.710	0.744
α	0.30	0.829	0.710	**0.869**

To compare the different deep learning methods, the performance on the test data with 50,000 samples was examined. The correlations between actual and recovered parameter values for each of the parameter from diffusion model for conflict tasks are reported in table 5.7. The three methods compared were deep learning models trained with clean data, deep learning models trained with noisy data, and transfer models. The highest correlation between recovered and actual parameter values for each parameter is shown in bold. The table shows that the deep learning models trained with noisy data outperformed the other two methods. This result, combined with the results in table 5.6, suggest that the most appropriate method for parameter recovery from experimental data is deep learning models trained with noisy data.

The only outlier from the comparison of deep learning methods was the amplitude of automatic activation parameter, which was best recovered by the transfer learning model. This result was also true when investigating the correlations from the twenty test distributions. Though the differences between correlation coefficients were not huge, transfer learning could be considered as a more suitable method for recovering amplitude of

Table 5.7: Comparison of different deep learning methods for parameter recovery from diffusion model for conflict tasks as evaluated on 50,000 test distributions. Correlations in bold represent the highest correlation value between actual and recovered parameter values for each parameter.

Parameter	Deep leaning - clean data	Deep leaning - noisy data	Deep leaning - transfer
β_1	0.726	**0.909**	0.876
μ_R	0.922	**0.988**	0.974
σ_R	0.827	**0.953**	0.922
μ_C	0.847	**0.861**	0.758
A	0.088	0.286	**0.317**
τ	0.059	**0.280**	0.202
α	0.457	**0.750**	0.657

automatic activation from experimental data.

In order to recover parameters from diffusion model for conflict tasks using deep learning, the following steps should be taken:

- First step: prepare and clean experimental data. Exclude any trials or whole datasets that do not meet inclusion criteria.
- Step two: separate the data into any experimental conditions that were present. So if speed-accuracy was manipulated in a study, there should be two datasets per participant (one for speed trials and one for accuracy trials).
- Step three: separate the datasets into congruent, incongruent, and neutral trials. In this way, each dataset should contain three subsets.
- Step four: pass datasets through feature extractor, to obtain features used in deep learning models.
- Step five: Do feature preprocessing: load in data, convert to NUMPY arrays [8], replace any NaNs with zeros.
- Step six: scale the features using a pickled scaler.
- Step seven: load in the trained models.

[8]inputs to keras models need to be of NUMPY array type

- Step eight: predict diffusion model for conflict tasks parameters using loaded models.

If a condition (most likely neutral trials) is missing, the models will need to be retrained with fewer input features. There should not be much reduction in predictive power as all input features were heavily correlated, however the models will need to be retrained to cope with different input dimensions.

5.6 Conclusions

This chapter investigated three different approaches in improving recovery of parameters from diffusion model for conflict tasks using deep learning when input features come from reaction time distributions with low number of trials. Such datasets contain more noise, so when deep learning models are trained with clean datasets that contain reaction time distributions with high number of trials, the models do not perform well. The three different approaches to improve the performance of the deep learning were training deep learning models with noisy input data, using transfer learning to pre-train models with clean data before finalizing training with noisy data, and using autoencoders for input data denoising.

Out of the three methods, autoencoders did not perform at all. Denoising data of reaction time distributions with low number of trials with the autoencoder model returned data that was correlated with the data of reaction time distributions with high number of trials to a lesser degree than the correlation between the two datasets without any denoising. This result could be explained by limited number of inputs into the autoencoder model. In essence, the autoencoder model was presented with only one example of noise per set of diffusion model for conflict tasks parameters. If there were multiple examples of noise per the same set of parameters, the models might have been able to start discriminating between noise and signal in reaction time distributions. Another explanation could be that stacked autoencoder architectures, as described in (Huang et al., 2007), can only deal with mild disturbances in data, while the disturbances in reaction time distributions with low number of trials might have contained too much noise for the autoencoders to correct.

The other two methods, training deep learning models with noisy data, and using deep learning models pre-trained on clean data for transfer learning, both performed very well at recovering parameters, and outperformed the deep learning models trained with cleaned data. This suggests that if parameters from diffusion model for conflict tasks need to be recovered from reaction time distributions with low number of trials, deep learning models that have been tuned with such noisy inputs should be preferred over deep learning models that were trained with clean data only. Interestingly, all deep learning methods outperformed differential evolution in recovery of parameters. Coupled with huge amount of computational time that is needed to recover parameters using differential evolution, this suggests that deep learning methods are superior to global optimization algorithms.

An interesting result is that the deep learning models trained on clean data could not recover "impulsivity" parameters (amplitude of automatic activation and time to peak automatic activation) from diffusion model for conflict tasks from data with low number of trials, suggesting that the noise in reaction time distributions mask the signal that is necessary for recovery of those parameters. However, the recovery improved slightly when deep learning models were either trained with noisy data or pre-trained with clean data only, suggesting that some signal remains if models are allowed to learn from noisy data. Looking at the history figure, it is evident that the model overfitted significantly and just learned to predict noise. However, when twenty test distributions were evaluated, the "impulsivity" parameters were recovered very well, as indicated by high correlations between actual and recovered parameter values. This indicates that the twenty test distributions could be different from the set of 50,000 distributions used as a large test dataset. The main difference could be that the sets of parameter values chosen for the twenty test distributions make realistic reaction time distributions. This could be tested by inspecting whether the noise pattern is different in realistic reaction time distributions.

One of the biggest differences between the datasets with low and high number of trials comes in NaN values in the input features. The dataset of reaction time distributions with large number of trials did not contain a single NaN value in either of the 92 input features for all 150,000 train and 50,000 test samples. However the datasets of reaction time distributions with low number of trials contained a lot of NaN values in the input features. In this chapter, the NaN values were replaced with zeros, which could affect the model performance significantly. It would be interesting to see whether training the models without the input features that contain the most amount of NaNs would improve the performance of deep learning models trained with noisy data.

To conclude, the best approach to using deep learning models in order to recover parameters from diffusion model for conflict tasks seems to be training the models with noisy input data. Pretraining models with clean data also seems to work quite well with the exception of drift rate for controlled process.

As deep learning seems to provide good and very quick parameter estimates from diffusion model for conflict tasks, the model can now be easily applied to experimental studies employing response inhibition tasks. The next three chapters will investigate the stability of impulsive behaviour in individuals over time and between different response inhibition tasks and decision making tasks, while also manipulating the speed and accuracy of responding. Using such complicated study designs would previously limit researchers to only examining the behaviour of individuals by using behavioural measures alone, such as response times and accuracy rates, as application of complex models, such as diffusion model for conflict tasks, would be computationally unfeasible. Due to the work described in the last two chapters, deep learning models can be used to estimate model parameters from hundreds of individual reaction time distributions with very little effort, even if the models were not tuned for particular tasks (such as decision making task in the last chapter). Deep learning will therefore allow to investigate the stability of impulsive behaviour by allowing the examination of cognitive processes associated with individual parameters from the diffusion model for conflict tasks.

6 Response inhibition tasks with speed-accuracy trade-off

6.1 Introduction

If response inhibition tasks are to offer a valuable contribution to understanding of both the positive and the negative aspects of impulsive behaviour in healthy and clinical populations, they need to provide precise and valid measurements using reliable paradigms (Wöstmann et al., 2013). There are multiple response inhibition paradigms being used (e.g. motor response inhibition includes go/no-go and stop-signal tasks, while interference inhibition is tested by tasks like the Simon (Simon & Rudell, 1967), Eriksen flanker (Eriksen & Eriksen, 1974), and the Stroop (Stroop, 1935) tasks). Even though there is a general assumption that similar response inhibition tasks measure similar underlying response control mechanisms (Friedman & Miyake, 2004), and that all these tasks measure the ability to resolve response conflict due to interfering stimulus features that are irrelevant (Wöstmann et al., 2013), the correlations between different response task measures have been low or absent (Aichert et al., 2012). Moreover, test-retest reliability of these tasks needs to be reasonable to allow any inferences about endophenotypes or traits.

The speed-accuracy trade-off (SAT), is a principle that governs all human behaviour: decisions that are fast are error prone, and decisions that are accurate tend to be slow (Heitz, 2014). SAT is however rarely manipulated in response inhibition tasks. In experimental tasks, participants need to decide what level of speed and accuracy is the most suitable to them and to the task, and these levels may differ in different testing sessions and different tasks, and as well as vary between individuals. Although SAT has been studied extensively in perceptual decision making (J. Zhang & Rowe, 2014; B. U. Forstmann et al., 2010; Rae et al., 2014; Pote et al., 2016; Winkel et al., 2012), studies on the effect of SAT on inhibitory control are sparse.

Only a few studies so far have looked at how SAT affects task performance in response inhibition paradigms. One of them, by Wylie et al. (2009), looked at the effect of speed-accuracy strategy on Eriksen flanker task performance in individuals with Parkinson's disease and also in healthy controls. The participants in this study were instructed to either respond as accurately as possible or as fast as possible to the stimuli, and the researchers found that both the response times and the accuracy of responding differed in both individuals with Parkinson's disease and the healthy controls depending on the instruction set. Another study by Leotti and Wager (2010) investigated the strategies that participants use in the stop signal task and found that individuals varied in the stop signal performance, as some complied with regular stop signal paradigm instructions that emphasize speed, while the other participants did not comply with the instructions and preferred slower but more accurate style of responding. When the researchers manipulated speed and accuracy by varying rewards for either fast or accurate responses, the participants changed their behavior in accordance to the SAT instructions. A study by Van Wouwe et al. (2014) looked into the effects of Parkinson's disease on the ability to resolve conflicts in the Simon task under speed and accuracy pressures. The researchers found that both healthy controls and individuals with Parkinson's disease showed typical performance under SAT conditions: when instructions emphasized response speed, participants responded faster and made more errors than when accuracy was emphasized. They also found that patients with Parkinson's disease struggled to suppress incorrect response impulses during speed but not accuracy conditions, suggesting that specific patterns of impulsivity could be revealed by manipulating SAT during response inhibition tasks. Even though these studies provide evidence that individuals' inhibitory behavior does vary with task demands, there are no studies yet investigating whether this ability is stable in time and between different response inhibition tasks.

This chapter will therefore evaluate the performance in two response inhibition tasks: the Eriksen flanker task and the Stroop task, in two experimental sessions, separated by four weeks, while also manipulating SAT. The flanker task and the Stroop task were chosen as they both include interference inhibition, where participants need to suppress the activation that results from task-irrelevant stimuli that interfere with the task-relevant stimuli because of their similarity. The SAT will be manipulated with instructions, as participants will be asked to perform as fast as possible, as accurately as possible, or both fast and accurately in different blocks of the task. The participants will also be retested in four weeks time. This study design allows answering four questions. First, do individuals show speed-accuracy trade-offs in response inhibition tasks? The expected finding is that the speed-accuracy trade-offs will be observed in the two response inhibition tasks, as they were present in studies by Leotti and Wager (2010), Van Wouwe et al. (2014), and Wylie et al. (2009). The second experimental question, whether individuals' ability to change responses to different speed-accuracy demands is consistent between different response inhibition tasks, will be assessed by looking at correlations in performance measures between two response inhibition tasks. The existing evidence for this question is sparse, and inconsistent between a multitude of response inhibition tasks. If the Stroop and flanker tasks both reflect similar aspects of individuals' inhibition, the performance of the two tasks will correlate between the tasks, provided that the behavioural measures are sensitive enough to reveal the inhibition of performance. The two tasks might even correlate if they only reflect similarities in decision making aspects of the two tasks that are not specific to inhibition. However, no correlations between the tasks will be observed if the behavioural measures are not sensitive representations of the decision making and/or inhibitory processes, or if the tasks do not reflect the same aspects of inhibition. Third question is, does the change in responding induced by speed-accuracy manipulations remain stable over time within the same task? If the tasks measure stable traits of inhibition (and/or decision making), we could potentially see correlations between task performance in time, provided that the behavioural measures are sensitive enough to reveal these traits. If, however, response inhibition tasks do not measure any stable traits, or if there are any stable traits, but they are not revealed by behavioural measures, no correlations will be observed in time. The main reason to include speed and accuracy manipulations in the experimental studies of response inhibition is that speed and accuracy manipulations tend to have an effect on behavioural measures, by introducing more range in individuals responses: under speed pressure, faster but more erroneous responses are expected, whereas under accuracy pressures, the responses should become slower but more accurate. This increased range in behaviour might help to reveal correlations between response inhibition tasks and within response inhibition tasks between sessions, as relationships are easier to distinguish from a wider range of potential response space. The final question is, does the difference in responses between speed and accuracy conditions (SAT effect) remain stable in time or between the tasks? This question is specifically investigating whether individuals' change between fast and accurate responding is a trait that is maintained either within the same task over time, or between different tasks. The study by (Leotti & Wager, 2010) suggested that individuals choose strategies in responding under speed pressure, with some individuals complying, and others not complying to instructions. If those strategies are consistent over time, correlations within the same task but over time would be expected, provided that the behavioural measures are sensitive enough to reveal these strategies. If the strategies in speed an accuracy compliance are consistent over multiple tasks, correlations between the Stroop and the flanker tasks might also be expected, provided that the behavioural measures are sensitive enough to reveal these strategies.

6.2 Methods

6.2.1 Participants

Fifty-seven students from Cardiff University School of Psychology were recruited to participate in the study in exchange of course credit or payment, 6 male (mean age 20.5 years old) and 51 females (mean age 20.7 years old). Forty-eight participants returned for the second testing session (6 male and 42 female, mean ages 20.5 years and 20.6 years respectively).

Two participants were excluded from the analysis: one due to ill health during experiment that resulted in the premature termination of study (excluded from the first session), and another one due to failure to concentrate on the task and the inability to complete both tasks in a single session (excluded from both sessions) - the participant only completed 2755 trials in both sessions (compared to 6960 trials in the full study that most participants completed), and 13% of the trials that were completed had to be removed as described in the data cleaning section below.

6.2.2 Design

The design consisted of within-subjects instruction condition (speed, accuracy, or both speed and accuracy emphasis), and within-subjects trial congruency condition (congruent, incongruent, or neutral trials). All participants completed both tasks and both sessions, but they were not analysed as factors.

6.2.3 Apparatus

The experiment took place in a dimly lit room; it was run on a Macbook Air computer connected to a 36.5cm by 27.5cm colour display (60Hz, resolution 1280px by 1024px) and an external keyboard. The tasks were programmed in PsychoPy version 1.73.04 (Peirce, 2009). Participants were seated at a distance of about 60 cm in front of the external display screen. Responses were made on the keyboard by pressing the 'z' and 'm' keys for left and right flanker responses respectively. For the Stroop task, participants responded by pressing 'z', 'x', 'n', or 'm' keys to indicate 'red', 'blue', 'green', or 'yellow' responses respectively.

6.2.4 Tasks and Procedure

Two response inhibition tasks were completed in one session: the computer version of the Stroop task (Stroop, 1935) and the Eriksen flanker task (Eriksen & Eriksen, 1974). In the Stroop task, a word is shown on a black computer screen written in coloured ink. The participant's task is to respond to the colour of the ink of the word by pressing one of four keyboard keys, each corresponding to a colour. In the neutral congruency trials, the word meaning is unrelated to colours (e.g.: 'advice', 'cross', 'ship', or 'lot'); in congruent trial condition, the word spells the same colour to that of the ink (e.g.: 'green' written in green ink, 'blue' written in blue ink); while in the incongruent trial condition, the word meaning and the ink colour are mismatched (e.g.: 'green' written in yellow ink, 'blue' written in red ink).

In the Eriksen flanker task, the participants are shown an arrow in the middle of the screen, and their task is to respond to the direction of that arrow only by pressing a keyboard key, corresponding to either left or right direction. In the neutral congruency trials, the central arrow is flanked by two dashes below and two dashes above. In the congruent trials, the central arrow is flanked by four other arrows that point to the same direction as the central arrow; while in the incongruent trial condition, the flanker arrows point in the opposite direction as the central arrow.

Three different instructions were given to participants throughout the study in both tasks before the beginning of each block. In Speed instructions, participants were asked to respond as fast as possible while still being correct, in the Accuracy blocks, participants were told to respond as accurately as possible without losing too much speed, while in the Both Speed and Accuracy blocks [9], participants were told to be both fast and accurate. On top of that, participants were given different feedback in each instruction condition: in the speed condition, participants saw "Too slow" written on the screen if they responded to the trial after the cut off period for each task (600ms for the Stroop task and 500ms for the Eriksen flanker task); in the accuracy condition, "Incorrect" was displayed on the screen if participants made an error, while no feedback was given in the both speed and accuracy conditions. Participants were also given feedback if they responded too quickly irrespective of instruction condition (faster than 200ms for the Stroop task and

[9] called "neutral instructions" in figures and tables

150ms for the Eriksen flanker task). After each block, participants were shown their average RT and accuracy rate for the previous block.

Participants completed 12 blocks in each task, four were speed, four accuracy, and four neutral (both speed and accuracy). Each block consisted of 144 trials, 48 were congruent, 48 incongruent, and 48 neutral. The order of the blocks was random and the order of trials within each block was also randomized for each participant. At the end of each session, participants completed the paper version UPPS-P impulsive behaviour scale (Lynam et al., 2006), which consists of 59 questions and measures 5 subsets of impulsivity. The questionnaire data was not analysed in this thesis.

The experimental session was completed by participants twice; the two sessions were separated by at least 4 weeks. Participants took 1.5 hours to complete each session (3 hours in total), with the flanker task taking approximately 40 minutes to complete, the Stroop task – approximately 45 minutes, and the questionnaire – approximately 5 minutes.

6.2.5 Data preparation

Before any analysis, data was cleaned by removing responses that were longer than 1.5 seconds, and the anticipatory responses (<150ms). All participants had some trials removed - the values ranged from 0.8% to 3.1% of trials per participant, mean 1.5%. No other trials were removed. All analyses relating to reaction times only report reaction times of the correct responses, while measures relating to accuracy looked at both correct and incorrect responses.

6.3 Results

6.3.1 Does SAT affect task performance?

In order to see whether SAT affected the task performance, the reaction times, the accuracy of responses, the reaction time costs, and the accuracy costs in both the flanker and the Stroop tasks were inspected in both experimental sessions. The results relating to the reaction times are shown in figure 6.1, revealing that overall, participants responded faster in the flanker task than in the Stroop task, which was expected given that the Stroop task was slightly harder due to four response options. The figure also demonstrates that both the congruency of trials, and the SAT instructions of the blocks, had an effect on the reaction times of the correct responses. These effects were statistically significant for both tasks and both sessions (all F values and p values reported in table 6.1 outer left column).

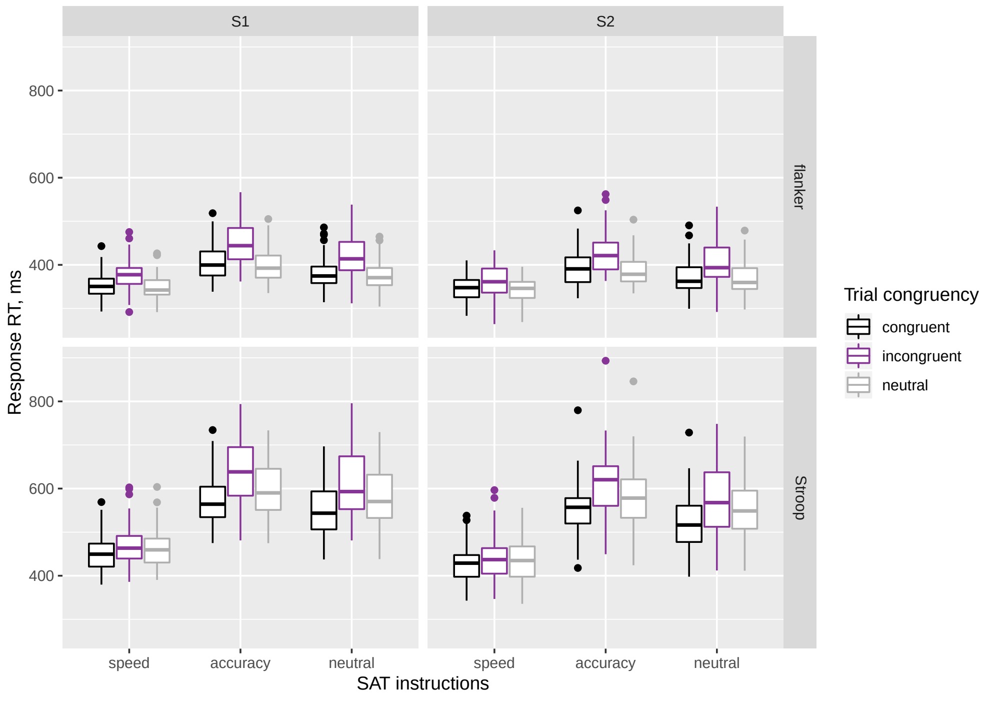


Figure 6.1: Effects of congruency (colour) and SAT instructions (x axis) on reaction times (y axis) in the flanker (upper row) and the Stroop (lower row) tasks in both experimental sessions (different columns). "Both fast and accurate" instructions are reported as "neutral". Dots indicate outliers.

Similarly, the accuracy of all of the responses is shown in figure 6.2. The figure shows that accuracy was overall lower in the Stroop task than in the flanker task, once again reflecting that the Stroop task was harder due to having four response options rather than the two response options in the flanker task. The SAT instructions affected the accuracy of responses in a predictable way, as the responses in speed instruction blocks were less accurate than in the accuracy instruction blocks. This pattern was more pronounced for the incongruent trials than the congruent trials. The interaction between SAT condition and congruency was statistically significant for both tasks and sessions (see table 6.1, inner left column).

Table 6.1: F values (p values) for the SAT x congruency interaction on reaction time and accuracy, and SAT effects on reaction time costs and accuracy costs. Significant results are shown in bold at the level of 0.006 (corrected for multiple comparisons). Stars next to p values indicate Greenhouse-Geisser correction for Mauchy's test of sphericity. The degrees of freedom were (4, 212) for session 1 and (4, 188) for session 2 for interaction effects, and (2, 106) for session 1 and (2, 94) for session 2 for the cost effects.

Task and session	Reaction time	Accuracy	Reaction time costs	Accuracy costs
Flanker Session 1	**34.5** (5.5×10^{-18})*	**25.0** (1.6×10^{-10})*	**45.5** (5.5×10^{-15})	**36.0** (1.1×10^{-10})*
Flanker Session 2	**24.2** (1.0×10^{-12})*	**11.2** (3.3×10^{-6})*	**28.2** (2.5×10^{-10})	**15.6** (6.1×10^{-6})
Stroop Session 1	**34.5** (6.9×10^{-20})*	**3.9** (0.005)	**53.0** (9.2×10^{-17})	**6.9** (0.001)
Stroop Session 2	**35.5** (4.6×10^{-18})*	**5.5** (3.5×10^{-16})*	**60.5** (1.7×10^{-17})	**9.7** (2.9×10^{-4})*

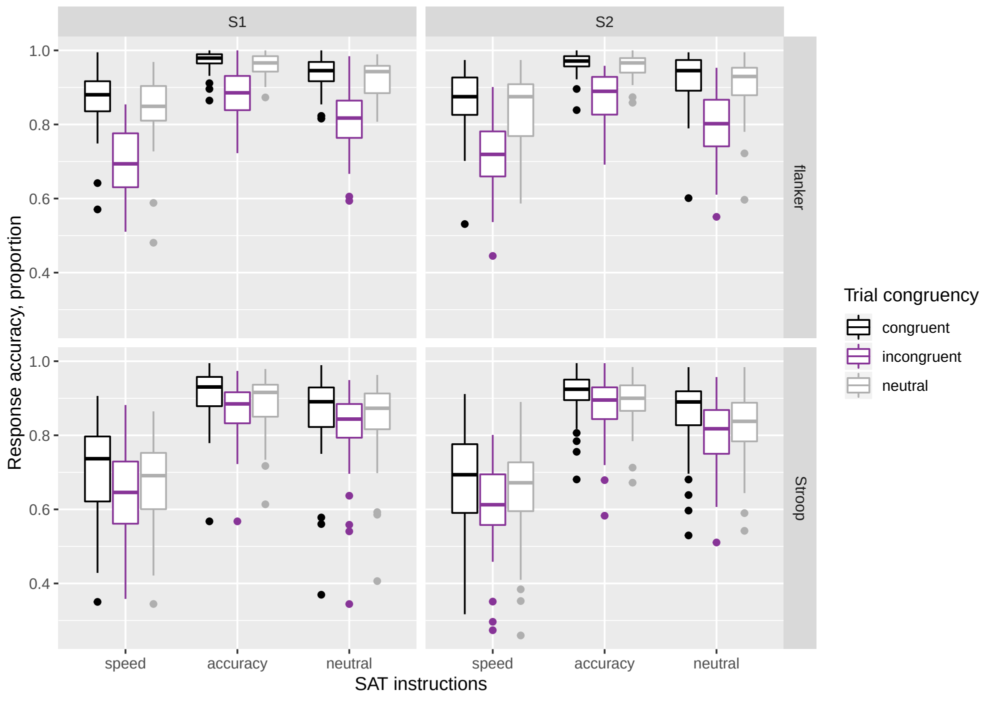


Figure 6.2: Effects of congruency (colour) and SAT instructions (x axis) on accuracy (y axis) in the flanker (upper row) and the Stroop (lower row) tasks in both experimental sessions (different columns). "Both fast and accurate" instructions are reported as "neutral". Dots indicate outliers.

Another performance measure of interest was the congruency costs: the difference in reaction times and accuracy between congruent and incongruent trials. Were congruency costs affected by the SAT manipulation? The results are shown in figure 6.3 for the reaction time costs and figure 6.4 for the accuracy costs, for both tasks and both sessions. The figures show that the congruency costs in reaction times were lower in the speed condition when compared to the accuracy condition, whereas the congruency cost in accuracy was higher in the speed condition when compared to the accuracy condition. This effect was statistically significant for all tasks and sessions (see table table 6.1 inner right column for reaction time costs and outer right column for accuracy costs).

6.3.2 Between-task correlations

As we have shown that SAT has an effect on performance in the flanker and the Stroop tasks, we can now investigate whether SAT manipulations also affect the correlations of task measures between the two different tasks performed in the same session. The correlations for reaction time costs are shown in figure 6.5 and for accuracy costs in figure 6.6. The figures show that for both measures, both sessions, and all three SAT conditions, correlations between the flanker and the Stroop task were very low. They are reported in table 6.2. The table shows that no between task correlations were significant for any session or SAT manipulation, suggesting that SAT manipulation does not help increase the correlations between the Stroop and the flanker tasks when compared to the standard instructions of “Be both fast and accurate”.

Table 6.2: Correlation coefficients(p values) between the reaction time costs and the accuracy costs in the flanker and the Stroop tasks, for all SAT manipulations and both sessions 1 and 2. Statistically significant correlations at the level of 0.008 (adjusted for multiple comparisons) are shown in bold.

SAT condition and session	Reaction time costs	Accuracy costs
Speed Session 1	0.15 (0.289)	-0.08 (0.556)
Speed Session 2	0.37 (0.011)	0.06 (0.693)
Accuracy Session 1	0.32 (0.017)	0.04 (0.784)
Accuracy Session 2	0.20 (0.172)	-0.00 (0.989)
Neutral Session 1	0.30 (0.026)	0.12 (0.400)
Neutral Session 2	0.25 (0.092)	-0.17 (0.242)

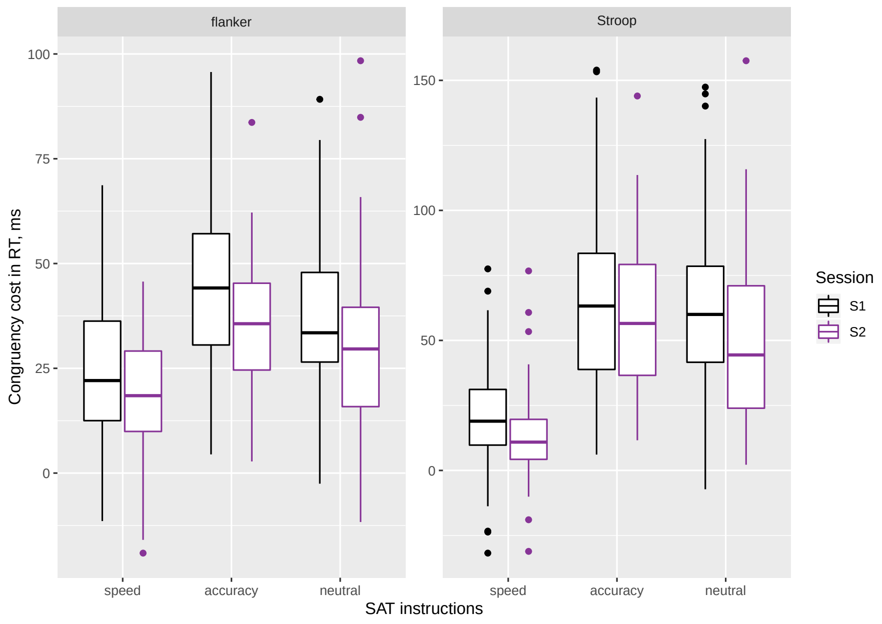


Figure 6.3: Effects of SAT instructions (x axis) on reaction time costs (y axis) in the flanker (left panel) and the Stroop (right panel) tasks in both experimental sessions (colour). “Both fast and accurate” instructions are reported as “neutral”. Dots indicate outliers.

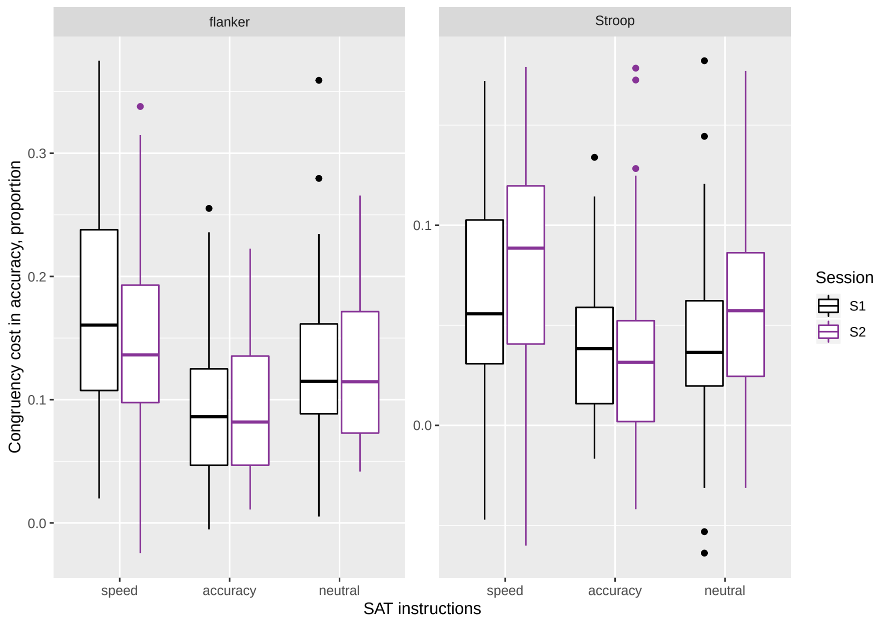


Figure 6.4: Effects of SAT instructions (x axis) on accuracy costs (y axis) in the flanker (left panel) and the Stroop (right panel) tasks in both experimental sessions (colour). "Both fast and accurate" instructions are reported as "neutral". Dots indicate outliers.

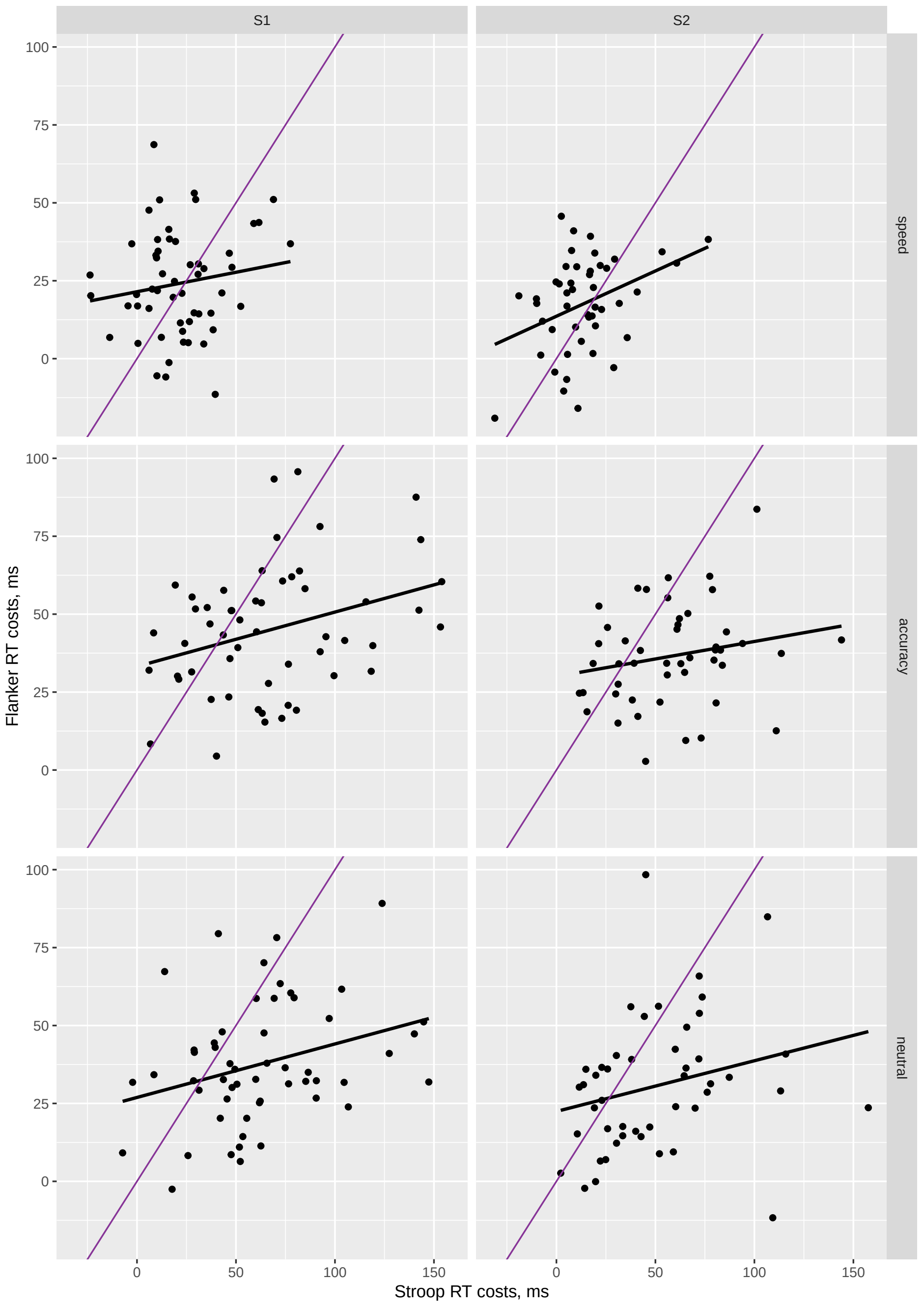


Figure 6.5: Correlations between the reaction time costs in the flanker and the Stroop tasks for session 1 (left column) and session 2 (right column) for different SAT instructions (rows). Black lines indicate the best fit lines for the correlations, while the purple lines display identity lines.

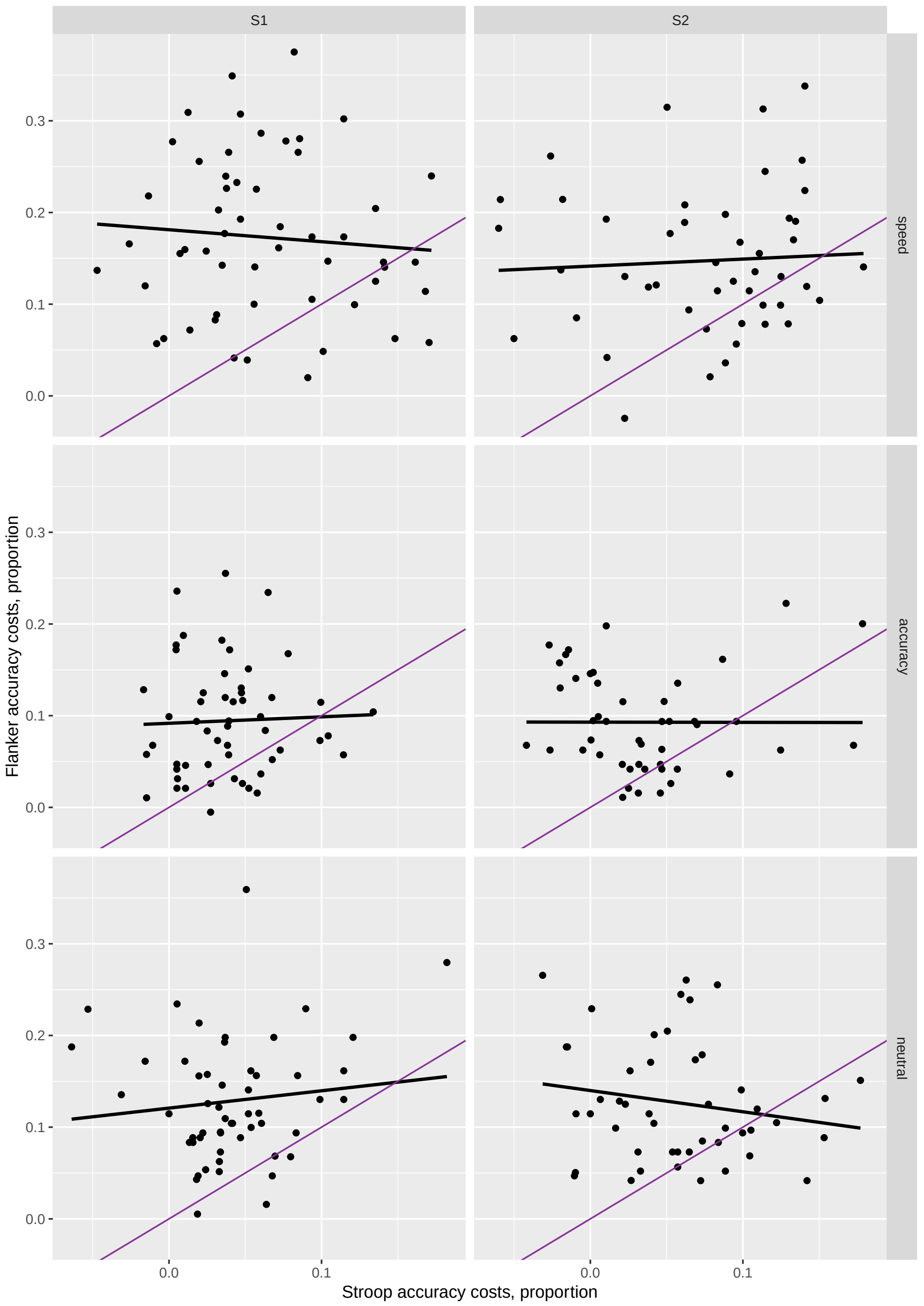


Figure 6.6: Correlations between the accuracy costs in the flanker and the Stroop tasks for session 1 (left column) and session 2 (right column) for different SAT instructions (rows). Black lines indicate the best fit lines for the correlations, while the purple lines display identity lines.

6.3.3 Between-session correlations

To see whether task performance was stable over time, the reaction time costs and the accuracy costs from session 1 were correlated with the same measures from session 2. The results are shown in figure 6.7 for the reaction time costs and figure 6.8 for the accuracy costs. The figures indicate that for both accuracy costs and the reaction time costs, there seems to be a stronger relationship between sessions in the same tasks than between different tasks in the same session. The correlation coefficients are reported in table 6.3. It appears that the reaction time costs correlate between sessions in the flanker and the Stroop tasks, however the accuracy costs correlate only in the flanker, but not the Stroop task. Adding the speed and accuracy manipulations does not seem to increase the correlations between sessions. This suggests that the reaction time costs are relatively stable over time, even without the SAT adjustments.

Table 6.3: Correlations (p values) between reaction time and accuracy costs between two testing sessions, for all SAT manipulations and both Stroop and flanker tasks. Correlations significant at the level of 0.008 (adjusted for multiple comparisons) are shown in bold.

SAT condition and task	Reaction time costs	Accuracy costs
Speed Flanker	**0.43** (0.003)	**0.55** (7.3×10^{-5})
Speed Stroop	**0.40** (0.005)	0.38 (0.009)
Accuracy Flanker	**0.47** (0.001)	**0.42** (0.003)
Accuracy Stroop	**0.47** (0.001)	0.19 (0.205)
Neutral Flanker	**0.59** (1.5×10^{-5})	**0.45** (0.001)
Neutral Stroop	**0.41** (0.005)	0.25 (0.096)

6.3.4 SAT costs between tasks and between sessions

The last question that was investigated in this chapter was whether the differences between congruency effect in reaction time and accuracybetween the speed and accuracy instructions (SAT cost; the difference between two difference scores) displayed stability across different testing sessions and between the two response inhibition tasks. The results for the SAT costs between the two tasks are shown in figure 6.9 and the results for the SAT costs between two experimental sessions - in figure 6.10. The figures show that there were was very little relationship between the SAT costs between either the flanker and the Stroop task or the two experimental sessions. The actual correlations are reported in table 6.4. The table shows that the only significant correlation between the SAT costs was in the reaction time costs in the Stroop task between the two experimental sessions.

Table 6.4: Correlations (p values) between the SAT costs in the reaction time costs and the accuracy costs between two testing sessions and two tasks. Correlations significant at the level of 0.0125 (adjusted for multiple comparisons) are shown in bold.

Condition	Reaction time costs	Accuracy costs
Between task session 1	0.07 (0.626)	-0.07 (0.620)
Between task session 2	0.17 (0.261)	0.32 (0.022)
Between session flanker task	-0.04 (0.768)	0.36 (0.013)
Between session Stroop task	**0.39** (0.006)	0.20 (0.179)

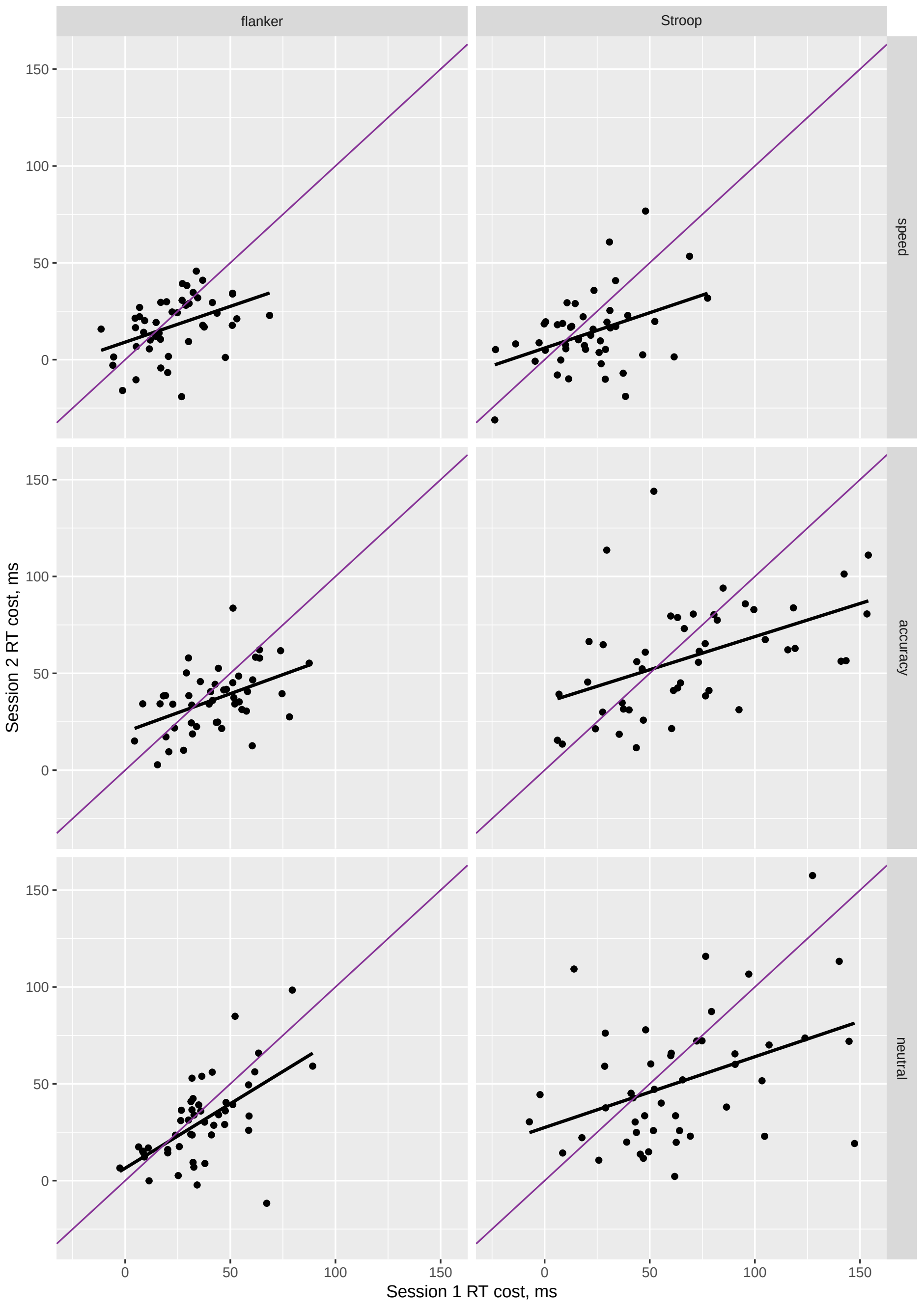


Figure 6.7: Correlations between the reaction time costs in session 1 and session 2 for the flanker (left column) and the Stroop (right column) tasks for different SAT instructions (rows). Black lines indicate the best fit lines for correlation, while the purple lines display identity lines.

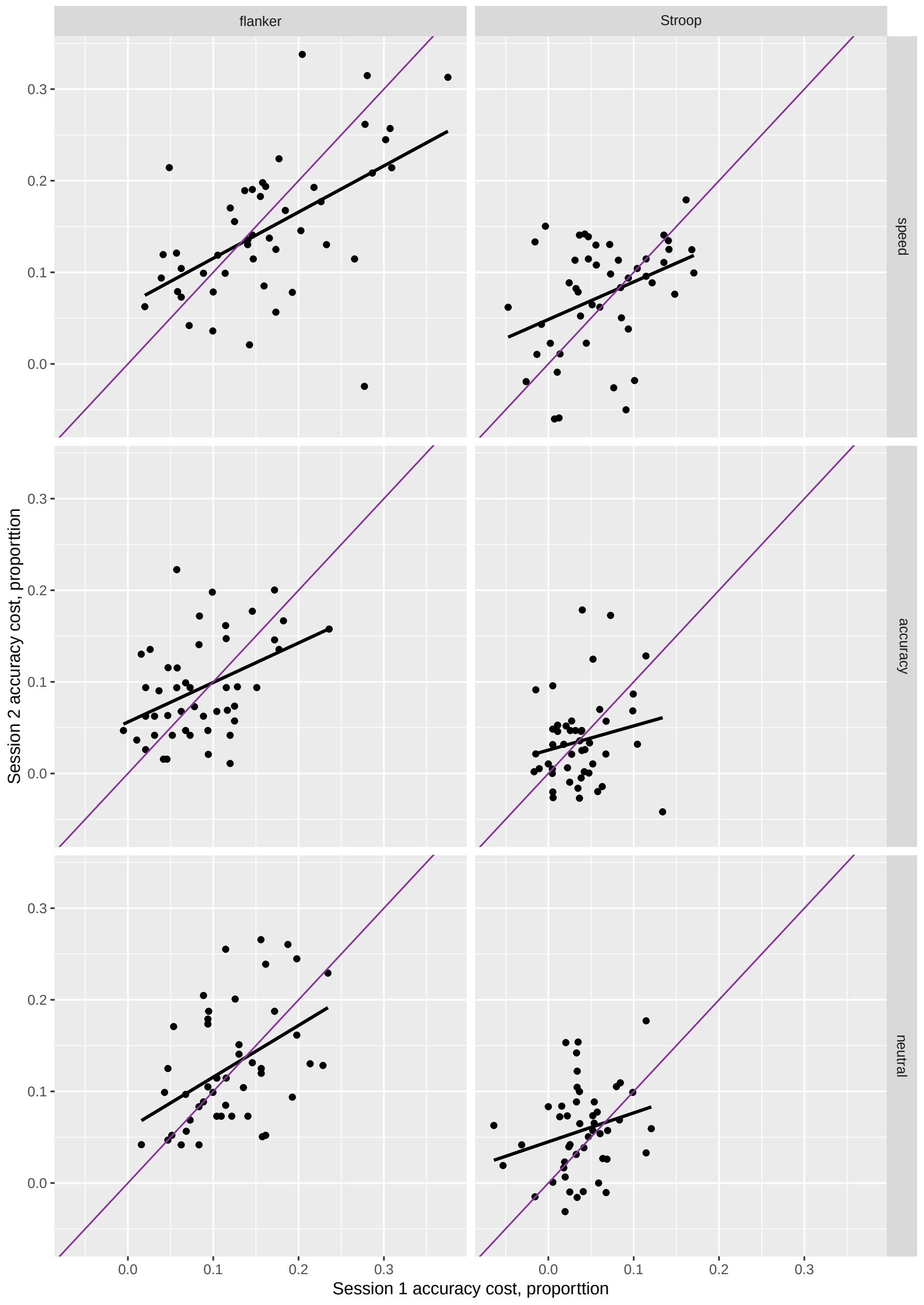


Figure 6.8: Correlations between the accuracy costs in session 1 and session 2 for the flanker (left column) and the Stroop (right column) tasks for different SAT instructions (rows). Black lines indicates the best fit lines for correlation, while the purple lines display identity lines.

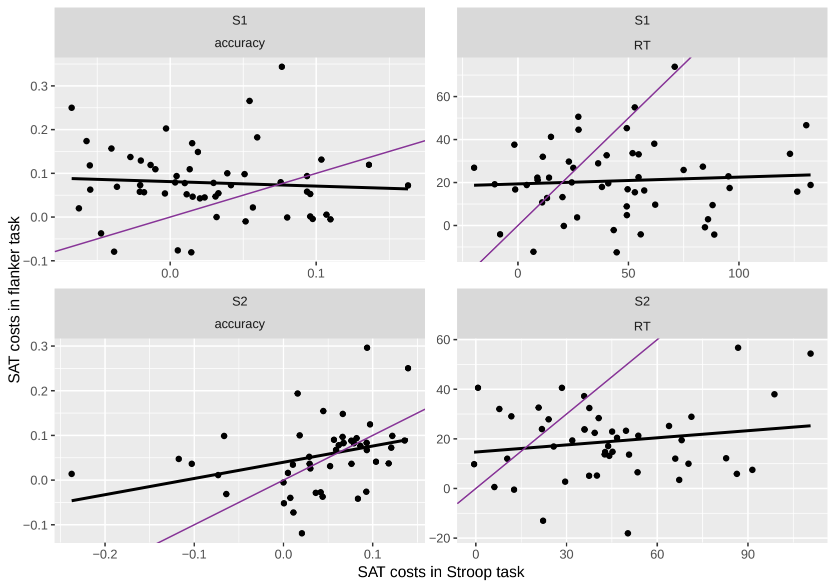


Figure 6.9: Correlations between SAT costs in reaction time costs (right column) and accuracy costs (left column) between two tasks for both session 1 (top row) and session 2 (bottom row). Black line indicates the best fit line for correlation, while the purple lines display identity lines.

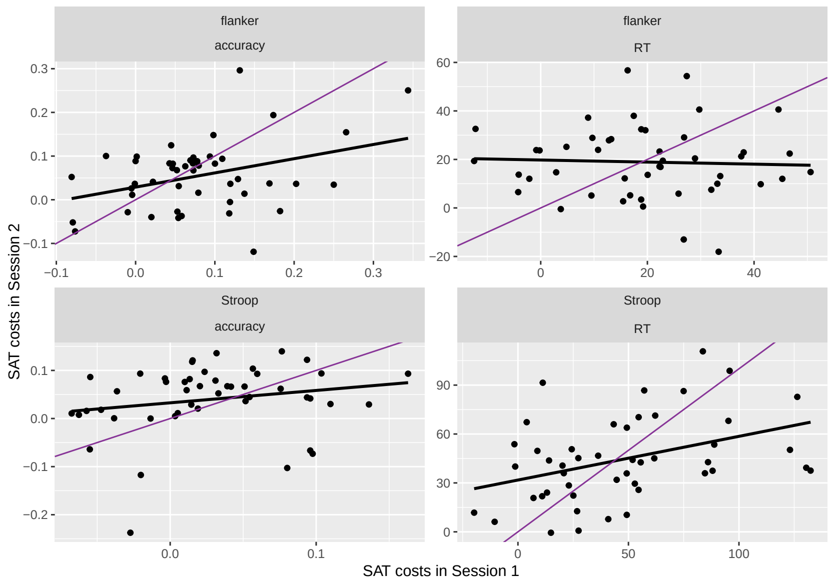


Figure 6.10: Correlations between SAT costs in reaction time costs (right column) and accuracy costs (left column) between two sessions for both flanker task (top row) and Stroop task (bottom row). Black line indicates the best fit line for correlation, while the purple lines display identity lines.

6.4 Discussion

This chapter investigated whether SAT has an effect on task performance in two different response inhibition tasks (the flanker task and the Stroop task), measured in two different points in time. The findings were consistent with the current sparse literature on the SAT manipulations in response inhibition tasks: individuals changed their behaviour when instructions emphasized either the speed or the accuracy of responding. The participants were the most accurate but also the slowest in the accuracy condition; while the opposite was true for the speed condition, as this resulted in the fastest response times but also the largest amount of errors. The congruency effects were also replicated: participants were slowest and made the most errors when responding to incongruent trials compared to congruent trials. Also, there was in interaction between the congruency of the trials and the SAT manipulation: the increase in errors between congruent and incongruent trials was greater in the speed condition compared to the accuracy condition, and the difference between the reaction times in congruent and incongruent trials was greater in the accuracy condition compared to the speed condition. Even though the overall pattern of performance was slightly different between the tasks (participants in the Stroop task were slightly slower and less accurate than in the flanker tasks), the results showed the same pattern in both tasks, and both experimental sessions.

The next question that this chapter tried to answer was whether the congruency effects were stable between the two different response inhibition tasks, by looking at the correlations between reaction time costs and accuracy costs from the Stroop and the flanker tasks. The results showed that neither the reaction time costs nor the accuracy costs correlated between the two tasks. The correlations did not improve when speed and accuracy manipulations were added to the instructions. This result was consistent for both experimental sessions. This suggests that the aspects of impulsivity captured by the two response inhibition tasks are not stable between the two tasks.

There could be several possible explanations for this finding. First, the Stroop task and the flanker task might be too dissimilar to expect similar performance from the participants. The Stroop task used in this study was quite different to the classical Stroop due to its adaptation as a computer task. This resulted in a four choice task, where participants were asked to make two decisions: which hand to use and then which finger to use for responding, which is different to just deciding on the colour to respond to. The two-decision process required to complete the Stroop task might have masked the stable aspects of impulsive behaviour if any exist.

Alternatively, the lack of correlations between the tasks could be explained by using difference scores (congruency costs as the difference between congruent and incongruent trials). This is explained well in Hedge, Powell, and Sumner (2018), where the researchers demonstrated that the difference scores do not consistently distinguish between individuals within a population, meaning that they would be difficult to use in correlational studies where differences between individuals is of interest, as in this chapter. Therefore the difference scores (reaction time costs and accuracy costs) might not be sensitive enough to reveal individual differences, and therefore correlations, between the two tasks.

The final question that this chapter intended to examine was whether behaviour in response inhibition tasks is stable in time. This was investigated by inspecting correlations between the reaction time costs in one session and another session within the same response inhibition task. The results showed that the performance in the flanker task was stable between the two experimental sessions, as measured by correlations in both the reaction time costs and the accuracy costs. The Stroop task only showed stability in time in reaction time costs, but not the accuracy costs. The speed and accuracy instructions did not increase the correlations when compared to regular instructions, for either the significant or insignificant effects, suggesting that the stability in time is unaffected by the SAT manipulations.

These findings in general are very interesting, because using difference measures did not hinder the reveal of the correlations between individuals in the different sessions. The lack of correlations between the accuracy costs while the correlations were there for the reaction time costs in the Stroop task is interesting, because the two measures, the reaction times and the error rates, are generally used interchangeably in the literature to show inhibitory control. This is consistent with Hedge, Powell, Bompas, et al. (2018), who showed very low correlations between the two measures, even though they supposedly measured the same thing. The same study showed that the correlations between the measures were more pronounced under speed instructions than under regular or accuracy instructions, however the SAT manipulations in the same sample did not improve the between-session correlations of either of the measures.

The next chapter will investigate whether the above issues can be addressed with the cognitive modelling framework. Cognitive modelling provides a theoretical framework for dissociating underlying cognitive mechanisms from the decision making tasks while also accounting for speed-accuracy trade-offs (B. Forstmann et al., 2016). Cognitive modeling is advantageous because it goes beyond description of data and seeks to provide an explanation of behaviour while being designed to be much simpler and abstract versions of human cognition (Lewandowsky & Farrell, 2010). The use of a decision making model specifically designed for response inhibition tasks, the diffusion model for conflict tasks (Ulrich et al., 2015), will allow to investigate not just whether the performance is stable between tasks and sessions, but also whether the aspects of behaviour that are related to impulsive decision making specifically, are stable between the tasks and the sessions.

7 Applying the diffusion model for conflict tasks to experimental data

7.1 Introduction

Section 6 has demonstrated that speed-accuracy trade-off (SAT) affects the performance in two response inhibition tasks, the Stroop task and the flanker task, and interacts with congruency to influence reaction time and accuracy measures. Even though there were clear effects of SAT on the reaction time and accuracy measures, it is difficult to make interpretations about impulsive behaviour from those measures alone, as outwardly they only reflect the speed of responding and the accuracy of responding. The two measures, even though supposedly reflecting the same outcomes, tend to not correlate well with each other (Hedge, Powell, Bompas, et al., 2018).

Both of these issues can be addressed with the cognitive modelling framework. Cognitive modelling provides a theoretical framework for dissociating underlying cognitive mechanisms from decision making tasks while also accounting for speed-accuracy trade-offs (B. Forstmann et al., 2016), therefore it is very suitable for the tasks in the previous chapter. For overview of the drift diffusion model by Ratcliff (1978) and the diffusion model for conflict tasks by Ulrich et al. (2015), please refer back to the introduction.

Application of a cognitive model to experimental data from section 6 allows investigation of multiple interesting questions, due to elaborate study design. First, application of the diffusion model by conflict tasks to the data would reveal how different speed-accuracy strategies affect model parameters. Boundary separation and caution is interesting for impulsivity and response inhibition research as in theory boundary separation is the only parameter that should be under direct influence of the participant. Pote et al. (2016) has managed to manipulate the level of caution and response threshold by subthalamic nucleus deep-brain stimulation in patients with Parkinson's disease, and induced impulsive action in patients when they were acting under speed pressure. This provides evidence that speed-accuracy trade-off and boundary separation is important in understanding impulsivity and caution. Even though historically only boundary separation has been assumed to be under direct control of participant (Voss et al., 2004), recently Rae et al. (2014) has challenged this assumption by demonstrating that changes in drift rate and non-decision time can also vary with distinct speed and accuracy pressures.

It is yet unknown whether drift rate and non-decision time would also vary under different speed and accuracy instructions in response inhibition tasks using the diffusion model for conflict tasks. It is also unknown whether different speed and accuracy pressures would have any influence on impulse function related parameters. If speed and accuracy pressures only have an effect on the general decision making aspects of the response inhibition tasks, then SAT manipulations should reveal different parameter values for different SAT conditions for the parameters shared between the drift diffusion model and the diffusion model for conflict tasks. The effects could be revealed for the automatic activation parameters, if the SAT manipulations also have an effect on the inhibitory aspects of the behaviour. Which specific parameters should vary with SAT manipulations is difficult to predict. The expectation is to at least observe an effect on the upper threshold parameter, as it is presumed to be under direct control of the individual.

Secondly, investigating model parameters from the data in section 6 would allow us to see whether it is impulsive behaviour, or just decision making overall, that results in correlations between different response inhibition tasks, if model parameters do indeed correlate between different response inhibition tasks. If impulsivity were to drive the correlations between the tasks, then we should see correlations between impulse parameters (amplitude of automatic activation and time to peak automatic activation) of the diffusion model for conflict tasks between the Stroop and the flanker tasks from section 6. Note that we did not find any correlations between the Stroop and the flanker tasks, in either the reaction time costs, or the accuracy costs. But this could have resulted from multiple reasons: either there being no relationship between the two tasks in either decision making or inhibitory aspects, or from the behavioural measures not being sensitive enough to reveal the relationships between the decision making and/or inhibitory processes. Looking at model parameters instead of behavioural measures therefore might offer more insight into the relationship between the Stroop and the flanker tasks. If SAT manipulations do affect model parameters, the relationship between the tasks might be easier to observe in the speed or the accuracy conditions as they might introduce more variance in the parameter values.

Finally, the design of section 6 allows us to inspect whether any of the parameters from the diffusion model for conflict tasks are stable in time. The behavioural results showed that the flanker task performance remained relatively stable over the four week period, whereas the Stroop task only showed stability when reaction time costs, but not accuracy costs, were considered. Which model parameters reflect this finding? We expect to see correlations between the two sessions in at least the upper threshold parameter, as it is firstly presumed to be affected by SAT manipulations, and would therefore offer more variance to reveal a relationship, and secondly, because the upper threshold parameter has an effect on reaction time of responding, and therefore should translate well between the findings in section 6 and this chapter. It is unknown whether the impusivity specific parameters should correlate within the tasks between the two different experimental sessions. If the impulsivity parameters are recovered well and if they reflect stable traits in task performance, then correlations between the parameter values in two experimental sessions might be observed.

7.2 Methods

7.2.1 Experimental data

The experimental data used in this chapter was identical to that from section 6, therefore please refer to the methods section of section 6 for details on participants, procedure, tasks, apparatus, design, and data preparation.

7.2.2 Model fitting

Experimental data was separated by participant, session, task, and SAT condition. This resulted in 612 sets of data that were fitted: $((55 + 47) \times 3 \times 2)$, where 55 and 47 is the number of participants in session 1 and session 2 respectively, 3 is the number of SAT manipulations, and 2 is the number of tasks. In total, seven parameters were fitted (upper threshold, non-decision time, standard deviation of non-decision time, drift rate for controlled process, amplitude of automatic activation, time to peak automatic activation, and shape of starting point distribution), while shape of automatic activation was fixed to 2 and diffusion constant was fixed to 4.

Deep learning models were used to estimate parameters from experimental data. The models that we used were the same as in section 5: the models trained with reaction time distributions with low number of trials. For this, the input features were extracted from each dataset, as described in section 4.2, then scaled according to the scaler fitted to the training dataset from section 5, and then applied individually for each diffusion model for conflict tasks parameter.

7.3 Results

7.3.1 What effect does speed-accuracy trade-off have on model parameters?

To ensure that deep learning recovered the parameters from the experimental data well, the model parameter sets were used to produce RT distributions of 1000 trials from each of congruent, incongruent, and neutral conditions. The accuracy, and the 25th, 50th, and 75th quantiles of RT were calculated for both correct and incorrect responses. Same measures were also calculated for the actual RT distributions obtained from participant responses. They are plotted in figure 7.3 to figure 7.6 for Stroop Session 1, Stroop Session 2, Flanker Session 1, and Flanker Session 2 respectively. The figures show that deep learning recovered the parameters very well as the predicted measures appear very closely matched to actual measures, with exception to very few participants. Note that individual dots in the figures represent an RT distribution for a SAT condition, so there will be three distributions (with speed, accuracy, and both speed and accuracy conditions) represented in each panel of the figure.

The recovered parameter values from the diffusion model for conflict tasks are shown in figure 7.1 for the flanker task and figure 7.2 for the Stroop task. As can be seen in the two figures, SAT did have an effect on parameter values. In the speed condition, the upper threshold is reduced, compared to accuracy condition, which is consistent with SAT account in drift diffusion model framework. However it looks like SAT does affect other parameters, not just the upper threshold, particularly the non-decision time parameters. This means that individuals seem to be able to change their response execution time when pressed for speed as opposed to accuracy, with non-decision time going down in speed condition, but variability of non-decision time going up. Shape of starting point distribution is also affected, but it is difficult to make interpretations about this parameter.

As differences between parameters themselves are not interesting, only main effects of SAT are going to be inspected. The results of the ANOVAs (with Greenhouse-Geisser corrections when Mauchy's test of sphericity was violated) are reported in table 7.1. The table shows that SAT instructions affected upper threshold, non-decision time, variability

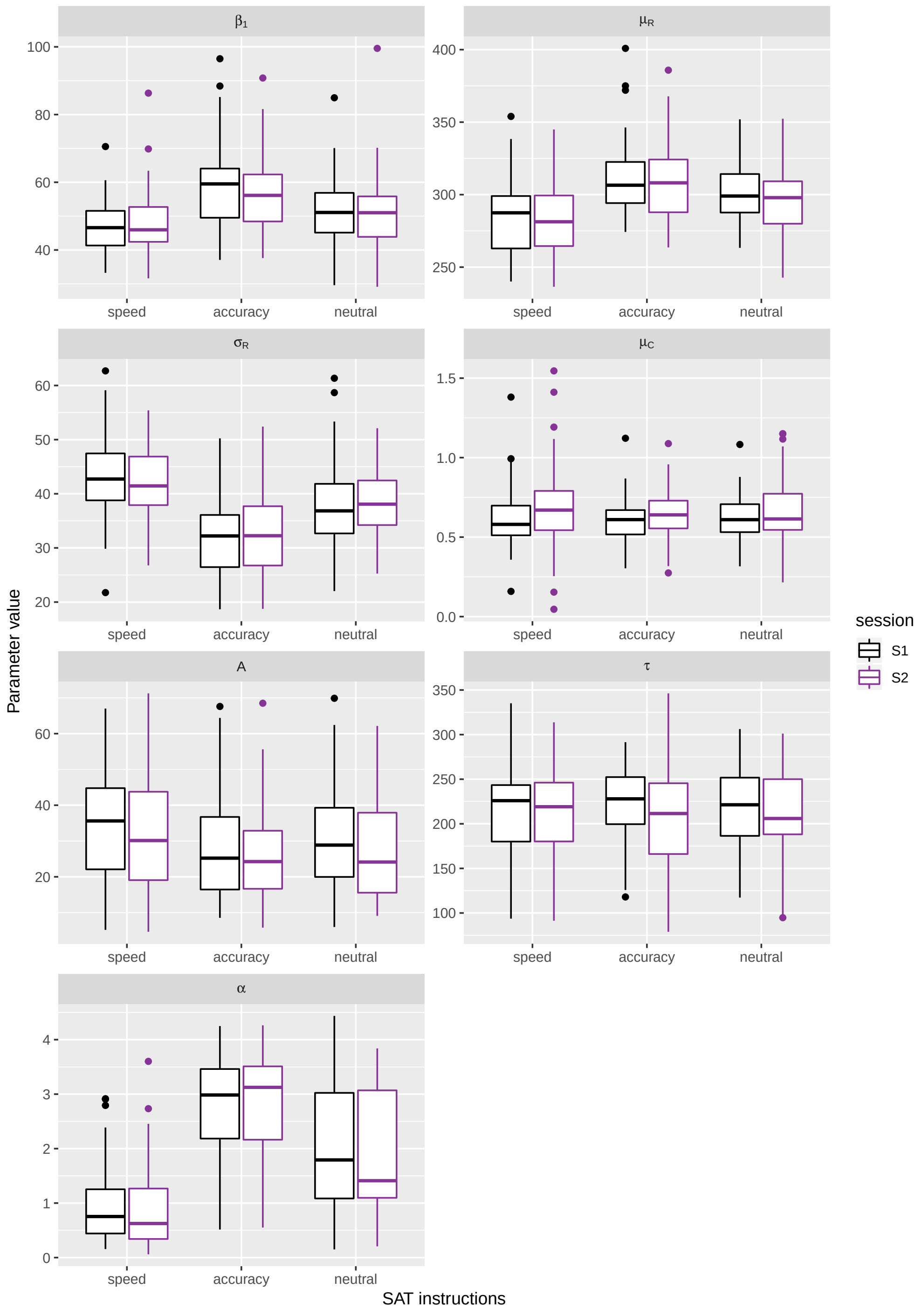


Figure 7.1: Effect of SAT on the diffusion model for conflict tasks parameters in flanker task. Results from session 1 are shown in black while results from session two are shown in purple. Dots indicate outliers.

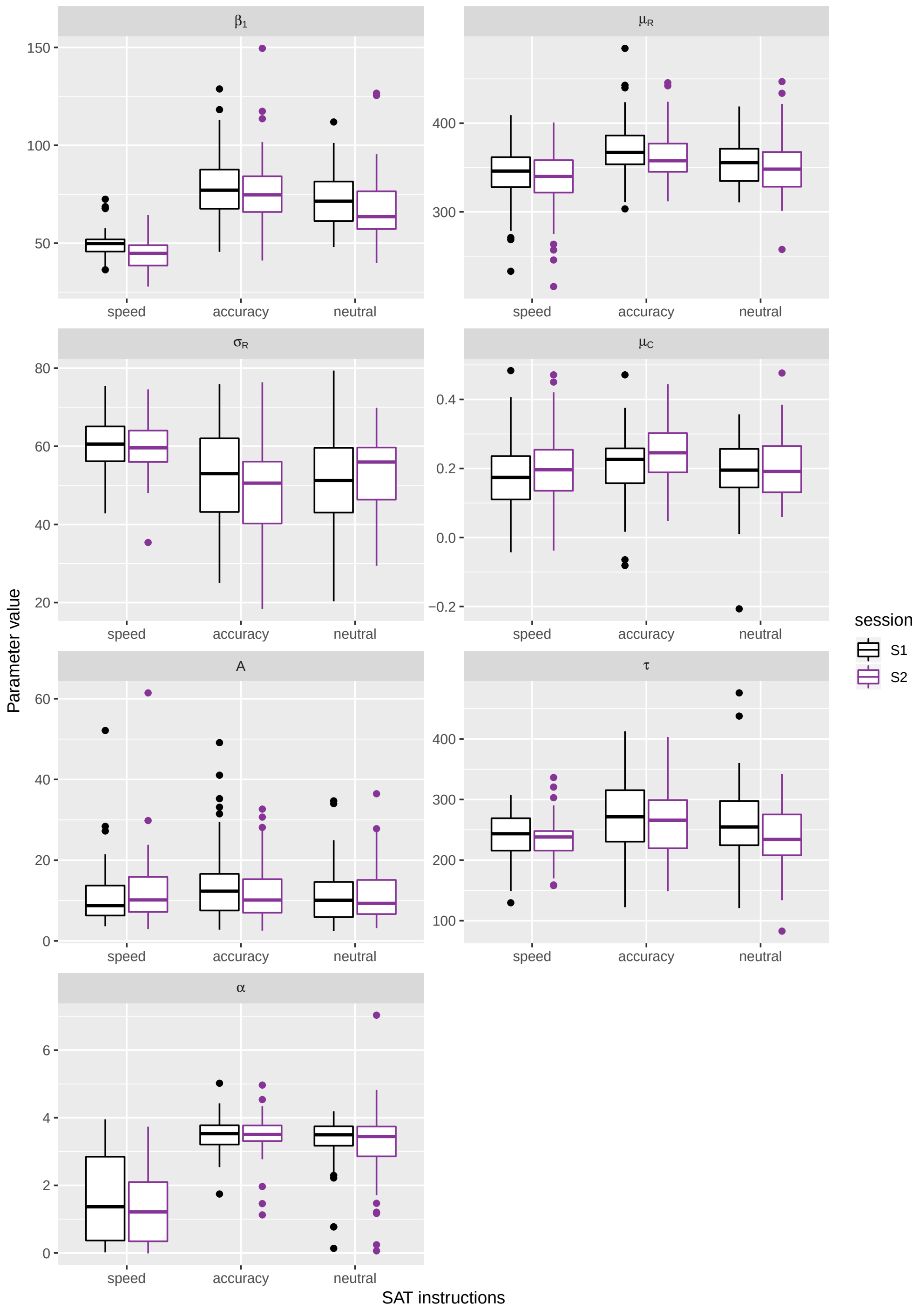


Figure 7.2: Effect of SAT on the diffusion model for conflict tasks parameters in Stroop task. Results from session 1 are shown in black while results from session two are shown in purple. Dots indicate outliers.

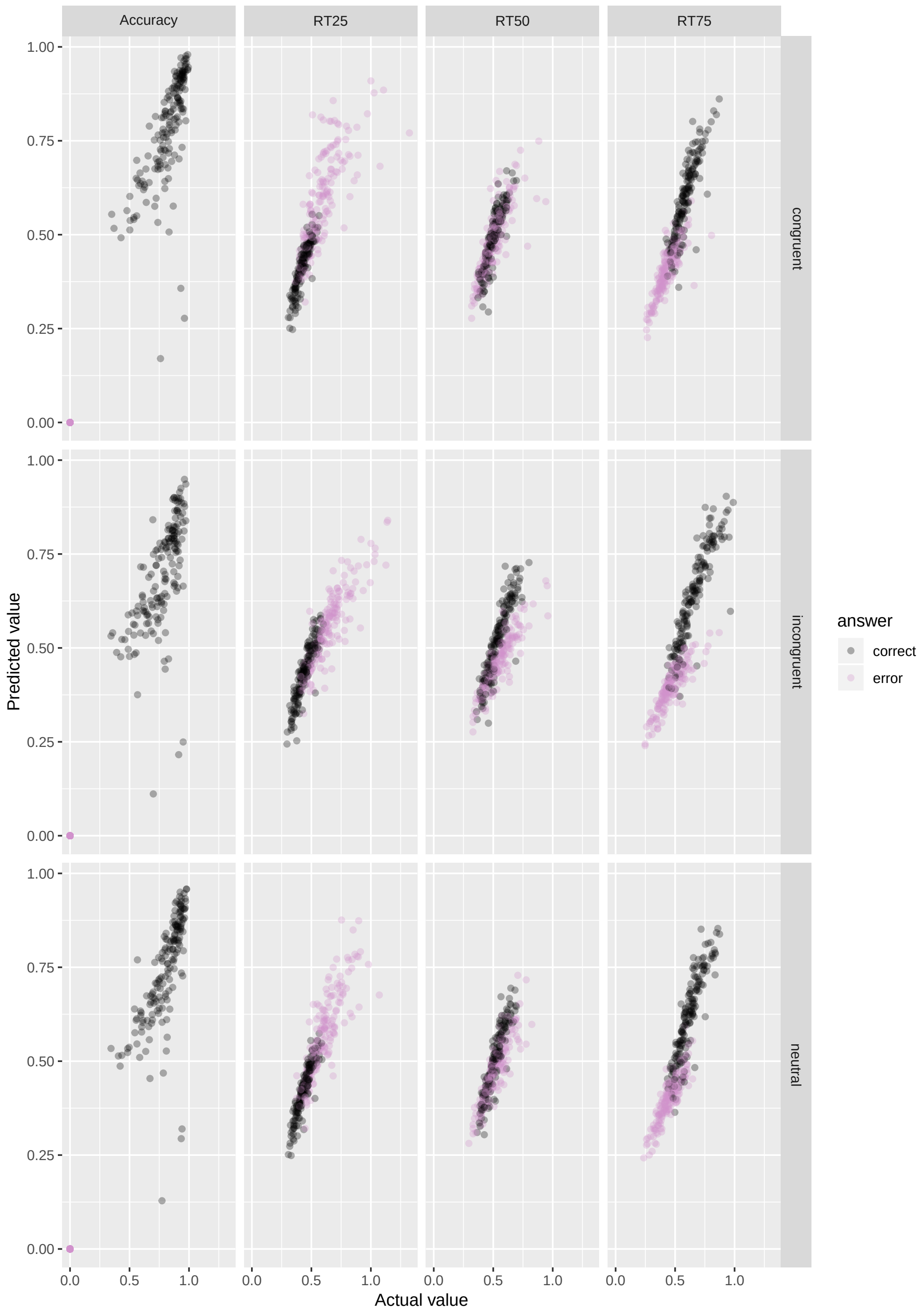


Figure 7.3: Predicted and actual accuracy, 25th RT quantile, 50th RT quantile, and 75th RT quantile (columns) of RT distributions for Stroop task Session 1, for both correct (black dots) and incorrect (pink dots) responses. The rows separate the trials by congruency.

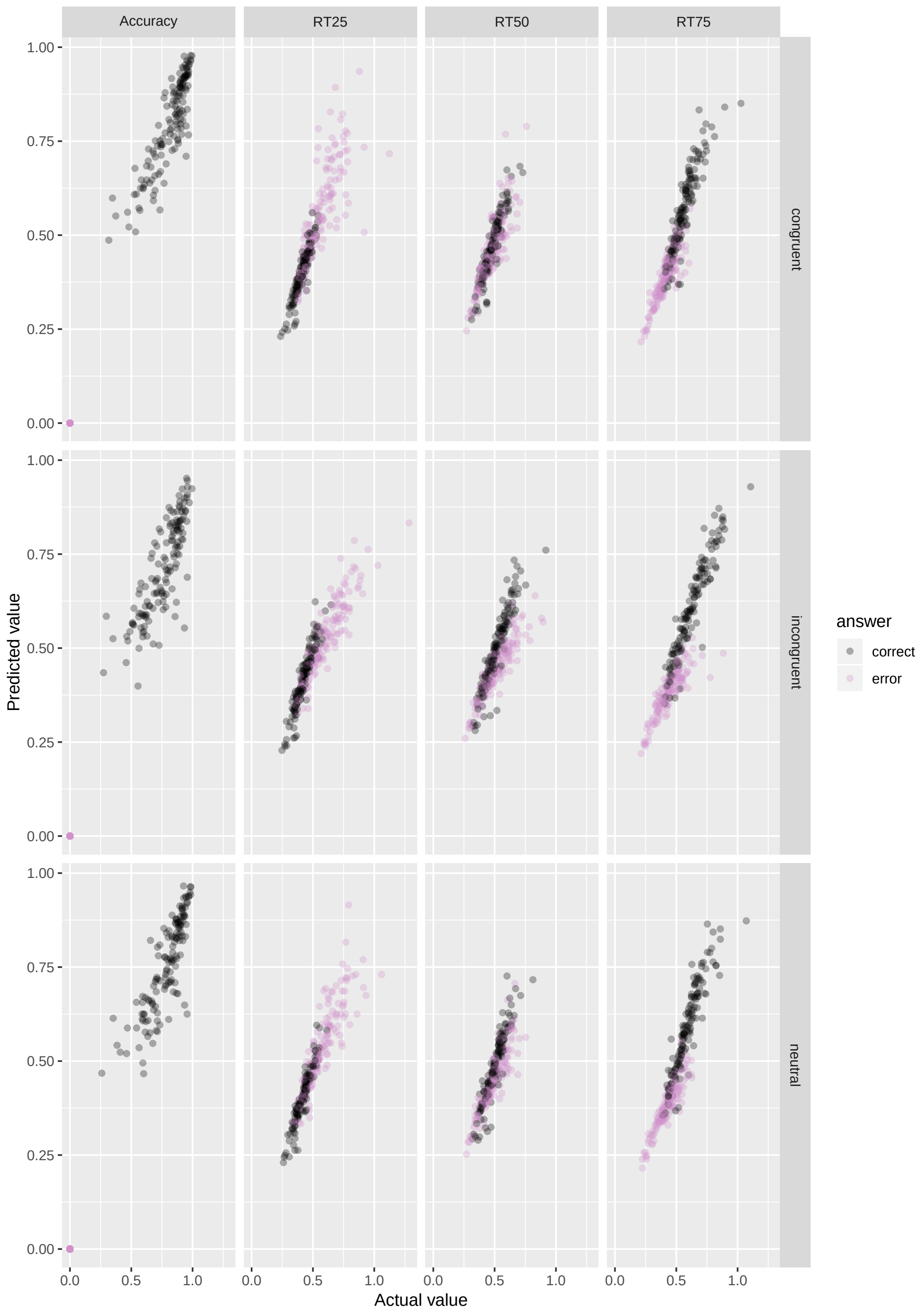


Figure 7.4: Predicted and actual accuracy, 25th RT quantile, 50th RT quantile, and 75th RT quantile (columns) of RT distributions for Stroop task Session 2, for both correct (black dots) and incorrect (pink dots) responses. The rows separate the trials by congruency.

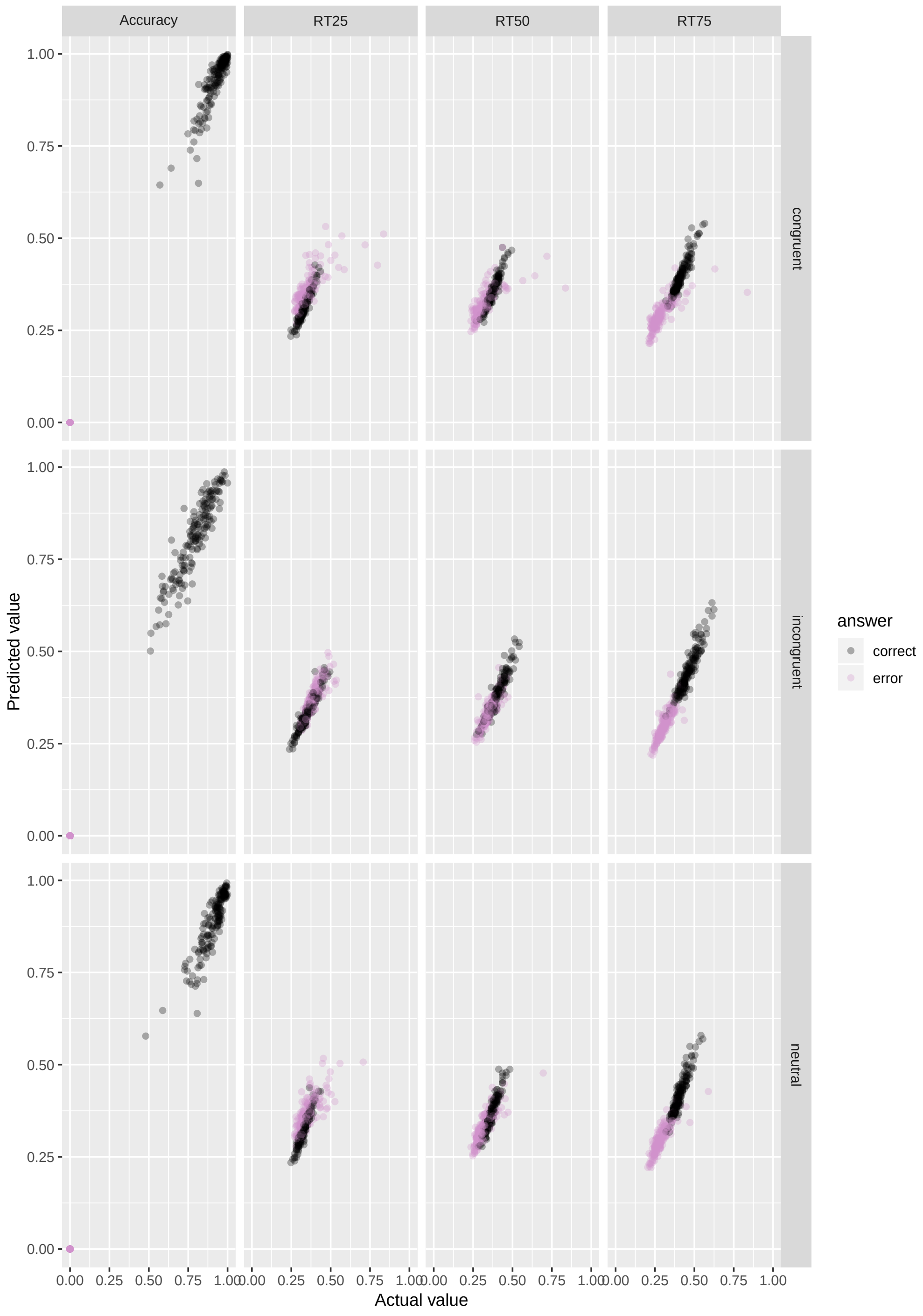


Figure 7.5: Predicted and actual accuracy, 25th RT quantile, 50th RT quantile, and 75th RT quantile (columns) of RT distributions for flanker task Session 1, for both correct (black dots) and incorrect (pink dots) responses. The rows separate the trials by congruency. The accuracy is measured in proportion, while the RT quantiles in seconds.

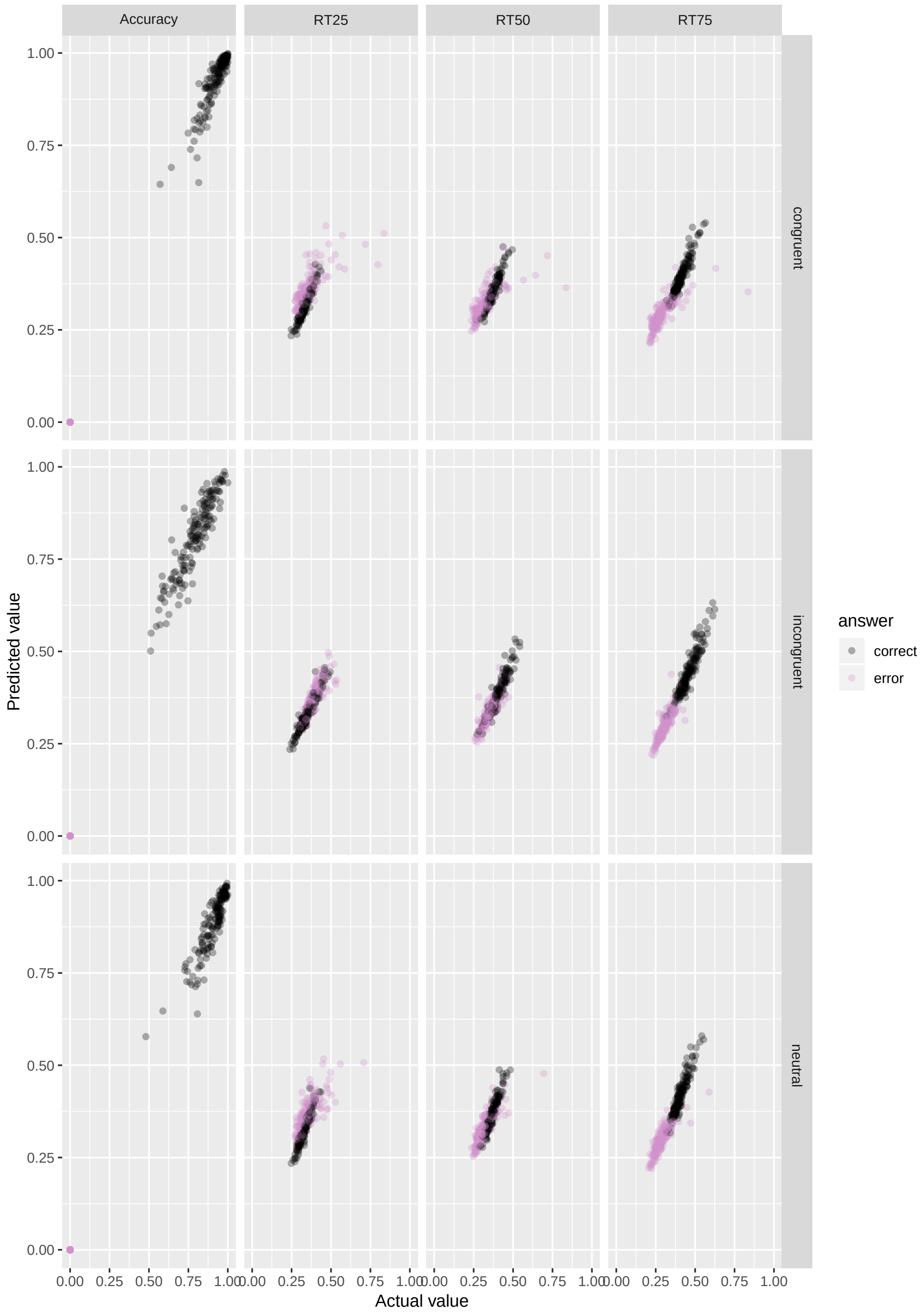


Figure 7.6: Predicted and actual accuracy, 25th RT quantile, 50th RT quantile, and 75th RT quantile (columns) of RT distributions for flanker task Session 2, for both correct (black dots) and incorrect (pink dots) responses. The rows separate the trials by congruency. The accuracy is measured in proportion, while the RT quantiles in seconds.

of non-decision time, and shape of starting point distribution. The results were the same for both Stroop and flanker tasks, and for both experimental sessions. There was no effect of SAT instructions on impulsivity parameters (amplitude of automatic activation and time to peak automatic activation), or the drift rate for controlled process. The only impulsivity-related parameter affected by SAT manipulations appears to be the upper threshold.

Table 7.1: F values (p values) indicating the statistical difference between model parameter values from different SAT conditions for flanker and Stroop tasks for both session 1 and session 2. Star after p value indicates Greenhouse-Geisser correction when Mauchy's test for sphericity was violated. The significant results are shown in bold (at corrected for multiple comparisons level of 0.002. Degrees of freedom for session 1 were (2,108), and for session 2 they were (2, 92).

Parameter	Flanker Session 1	Flanker Session 2	Stroop Session 1	Stroop Session 2
β_1	**40.2** (6.7×10^{-12})*	**10.3** (2.1×10^{-4})*	**122.3** (1.5×10^{-21})*	**98.1** (5.6×10^{-21})*
μ_R	**40.8** (6.4×10^{-12})*	**36.9** (2.4×10^{-11})*	**20.7** (1.0×10^{-6})*	**12.0** (7.3×10^{-5})*
σ_R	**57.8** (8.4×10^{-18})	**35.6** (3.5×10^{-12})	**15.0** (1.8×10^{-6})	**23.3** (6.4×10^{-9})
μ_C	0.2 (0.791)*	1.0 (0.379)*	2.8 (0.075)*	4.8 (0.012)*
A	5.8 (0.004)	2.0 (0.152)*	3.2 (0.045)	0.2 (0.756)*
τ	0.6 (0.920)	0.4 (0.645)	5.9 (0.004)	5.4 (0.006)
α	**77.8** (1.2×10^{-21})	**87.7** (4.8×10^{-22})*	**82.7** (2.7×10^{-18}))*	**78.7** (1.2×10^{-20}))

7.3.2 Between task correlations of model parameters

The correlations between model parameters from the flanker task and the Stroop task are shown in figure 7.7 for session 1 and figure 7.8 for session 2, for all SAT instruction conditions. The correlation coefficients (with p values) are reported in table 7.2.

When the neutral instruction condition is considered by itself, only the upper threshold parameter showed a correlation between tasks in session 1, but not session 2. In session 2, only the non-decision time parameter correlated between the two tasks. When

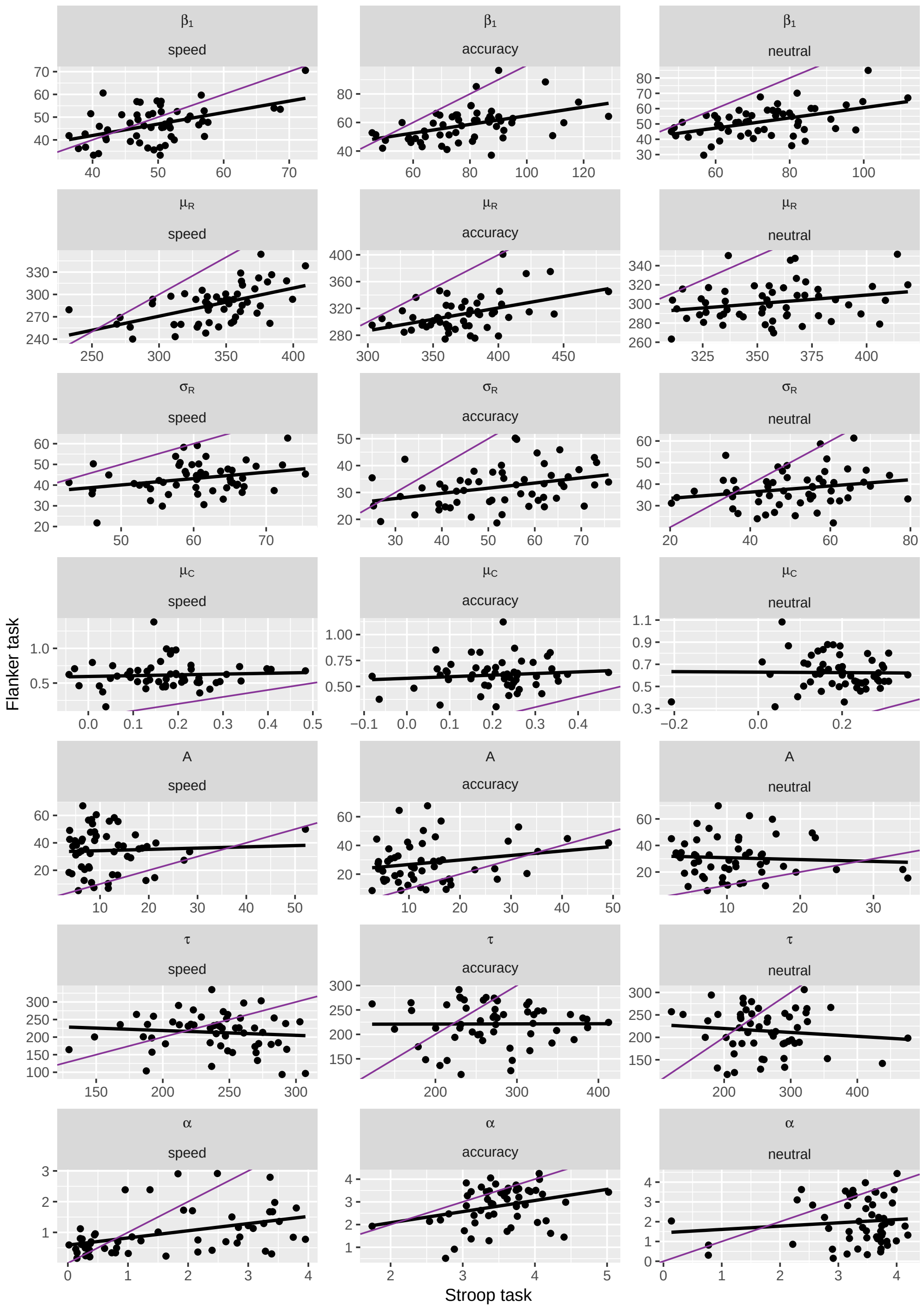


Figure 7.7: Correlations between model parameters from Stroop task (x axis) and model parameters from flanker task (y axis) from session 1 of the first study, for speed (left column), accuracy (middle column), and neutral (right column) instruction conditions. Black line shows actual correlation between the two tasks, whereas purple line indicates the identity line.

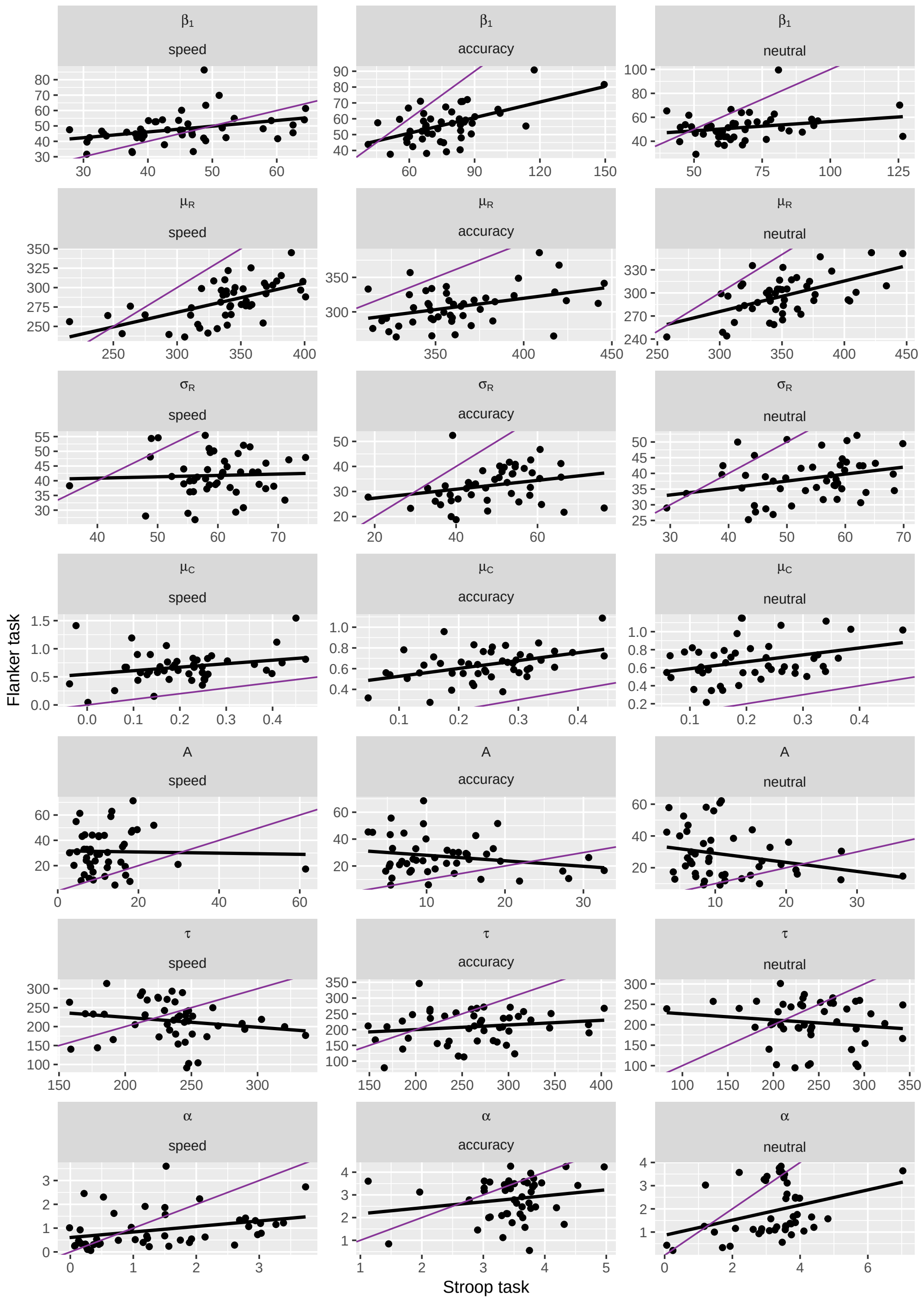


Figure 7.8: Correlations between model parameters from Stroop task (x axis) and model parameters from flanker task (y axis) from session 2 of the first study, for speed (left column), accuracy (middle column), and neutral (right column) instruction conditions. Black line shows actual correlation between the two tasks, whereas purple line indicates the identity line.

the speed and accuracy conditions were inspected, it appeared that the non-decision time also correlated between the two tasks in speed condition in both sessions, but only in session 1 was a correlation revealed in the accuracy condition. In session 2, in the accuracy condition, drift rate for controlled process also correlated between tasks.

The impulsivity parameters (amplitude of automatic activation and time to peak automatic activation) did not correlate between tasks, neither in neutral SAT instructions, nor in the speed or the accuracy instructions.

It must be noted that correlations in session two are harder to spot as the number of participants who came back for session two was lower. Also, very strict criteria for significance were applied to adjust for multiple comparisons, making correlations even harder to spot.

However, overall, it appears that very few parameters from the diffusion model for conflict task correlate between the Stroop and the flanker tasks. Moreover, adding SAT manipulations to the experiments does not seem to help with correlations between tasks for model parameters.

7.3.3 Between session correlations of model parameters

The correlations between model parameters from session 1 and session 2 of the tasks are shown in figure 7.9 for the flanker task and figure 7.10 for the Stroop task for all SAT instruction conditions. The correlation coefficients (with p values) are reported in table 7.2.

The figures and table show that in the neutral condition, flanker task displayed moderate significant correlations between session 1 and session 2, even after harsh adjustments for multiple comparisons, for all parameters but drift rate for controlled process (which resulted from the harsh cutoff for significance set at 0.0024, while the actual value for drift rate was 0.003). This was true for all drift diffusion parameters, as well as automatic activation parameters. Therefore adding SAT manipulations is not necessary for correlation between sessions for the flanker task, if anything, most correlations seem to be lower and non-significant in the speed condition.

The results are similar for the Stroop task: four out of seven diffusion model for conflict

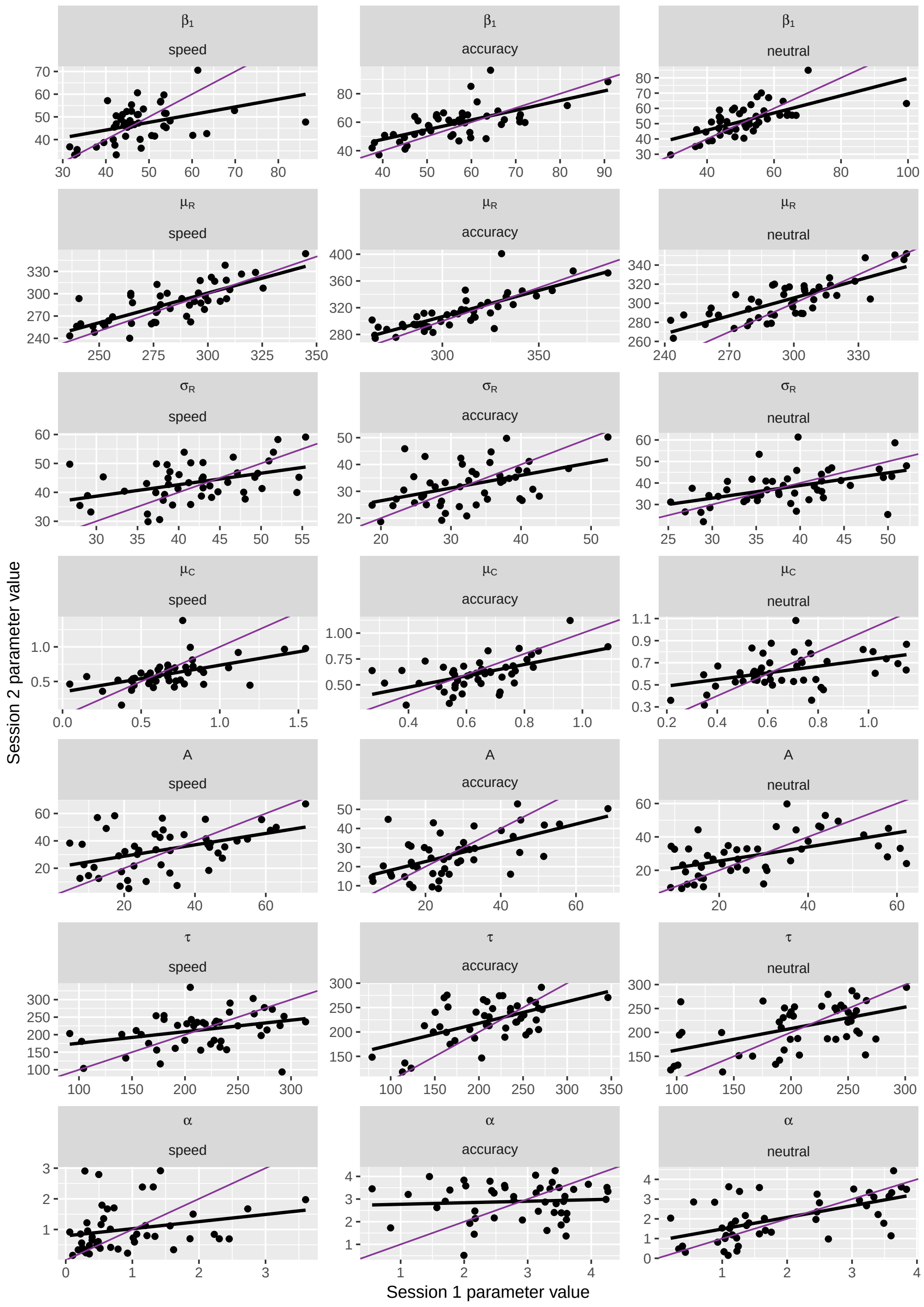


Figure 7.9: Correlations between model parameters from Session 1 (x axis) and Session 2 (y axis) from flanker task, for speed (left column), accuracy (middle column), and neutral (right column) instruction conditions. Black line shows actual correlation between the two tasks, whereas purple line indicates the identity line.

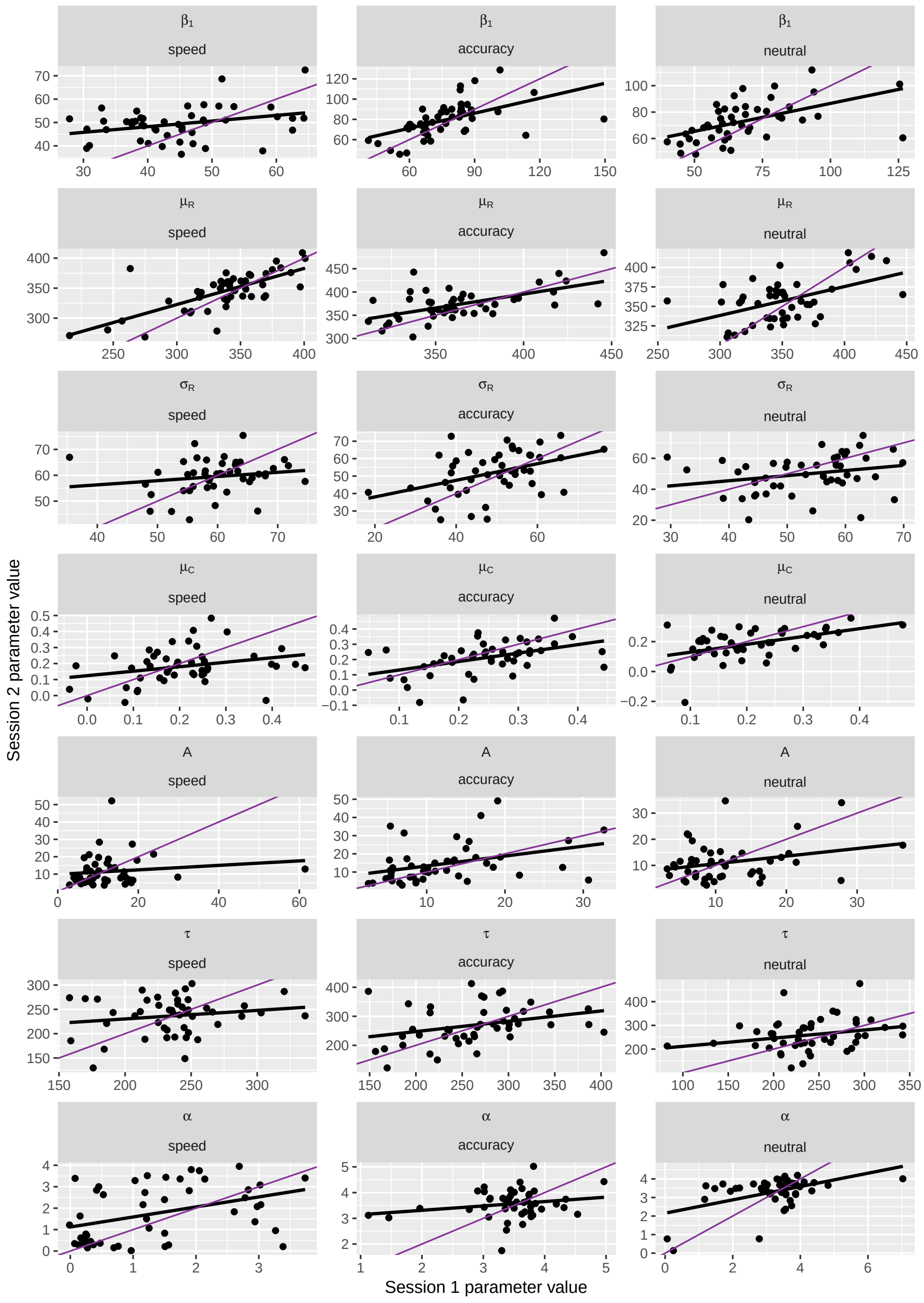


Figure 7.10: Correlations between model parameters from Session 1 (x axis) and Session 2 (y axis) from Stroop task, for speed (left column), accuracy (middle column), and neutral (right column) instruction conditions. Black line shows actual correlation between the two tasks, whereas purple line indicates the identity line.

Table 7.2: Correlation coefficients (p values) between parameter values from flanker and Stroop tasks for both session 1 (top) and session 2 (bottom)

	Session 1		
SAT instructions	Speed	Accuracy	Neutral
β_1	**0.47** (2.8×10^{-4})	**0.45** (5.1×10^{-4})	**0.49** (1.6×10^{-4})
μ_R	**0.53** (2.9×10^{-5})	**0.45** (5.7×10^{-4})	0.25 (0.070)
σ_R	0.29 (0.031)	0.36 (0.012)	0.22 (0.101)
μ_C	0.07 (0.632)	0.11 (0.420)	-0.01 (0.918)
A	0.05 (0.727)	0.22 (0.109)	-0.07 (0.635)
τ	-0.10 (0.457)	0.00 (0.974)	-0.12 (0.399)
α	**0.42** (0.001)	0.30 (0.024)	0.12 (0.393)
	Session 2		
	Speed	Accuracy	Neutral
β_1	0.36 (0.012)	**0.56** (4.2×10^{-5})	0.25 (0.092)
μ_R	**0.58** (1.9×10^{-5})	0.39 (0.006)	**0.56** (4.8×10^{-5})
σ_R	0.05 (0.756)	0.27 (0.067)	0.31 (0.037)
μ_C	0.26 (0.080)	**0.44** (0.002)	0.33 (0.022)
A	-0.03 (0.864)	-0.21 (0.152)	-0.26 (0.077)
τ	-0.19 (0.201)	0.17 (0.261)	-0.14 (0.343)
α	0.33 (0.026)	0.20 (0.188)	0.34 (0.019)

task parameters show significant correlations in the neutral instruction condition, only three in the accuracy condition, and only one in the speed condition, indicating that adding SAT manipulation to experimental procedures does not help with improving correlations between model parameters. The variability of non-decision time was the only drift diffusion model parameter that did not correlate between sessions in the Stroop task. Unlike with the flanker task, the impulsivity parameters did not correlate between sessions in the Stroop task.

7.3.4 SAT costs in model parameters between tasks and between sessions

The results reported in section 6 revealed that there did not seem to be a relationship between the difference in reaction time or accuracy costs between the speed and accuracy conditions between the two experimental sessions, while only the SAT difference in reaction time costs in the Stroop task showed significant correlation. Do parameters from the diffusion model for conflict tasks show better stability in SAT costs either between tasks or between sessions? As five of the seven model parameters were

Table 7.3: Correlation coefficients (p values) between parameter values from session 1 and session 2 for both flanker (top) and Stroop (bottom) tasks

	Flanker task		
SAT instructions	Speed	Accuracy	Neutral
β_1	0.41 (0.004)	**0.64** (1.1×10^{-6})	**0.63** (1.7×10^{-6})
μ_R	**0.77** (3.0×10^{-10})	**0.81** (7.2×10^{-12})	**0.79** (5.3×10^{-11})
σ_R	0.42 (0.003)	**0.45** (0.002)	**0.49** (4.4×10^{-4})
μ_C	**0.54** (0.002)	**0.56** (4.3×10^{-5})	0.43 (0.003)
A	0.44 (0.024)	**0.59** (1.4×10^{-5})	**0.52** (1.7×10^{-4})
τ	0.34 (0.020)	**0.56** (4.1×10^{-5})	**0.50** (3.4×10^{-4})
α	0.25 (0.095)	0.07 (0.619)	**0.58** (1.7×10^{-5})
	Stroop task		
	Speed	Accuracy	Neutral
β_1	0.32 (0.030)	**0.53** (1.3×10^{-4})	**0.52** (1.5×10^{-4})
μ_R	**0.72** (1.4×10^{-8})	**0.58** (1.9×10^{-5})	**0.50** (4.0×10^{-4})
σ_R	0.17 (0.247)	0.42 (0.003)	0.25 (0.090)
μ_C	0.29 (0.046)	**0.46** (0.001)	**0.50** (3.7×10^{-4})
A	0.14 (0.359)	0.37 (0.010)	0.28 (0.056)
τ	0.17 (0.244)	0.32 (0.030)	0.26 (0.078)
α	0.39 (0.006)	0.21 (0.160)	**0.50** (3.7×10^{-4})

affected by SAT manipulations, do the same parameters show stability in the change between the speed and accuracy conditions? The results are shown in figure 7.11 for SAT costs between the two tasks and in figure 7.12 for the results between two experimental sessions. The figures show that there were was very little relationship between the SAT costs between either the flanker and the Stroop task or the two experimental sessions. The actual correlations are reported in table 7.4. The table shows that the only two parameters that showed consistent effects in the change between SAT conditions were upper threshold and non-decision time. The non-decision time was significantly correlated only between the two sessions, as between task correlations failed to reach significance due to strict adjustments for multiple comparisons. Neither variability of non-decision time nor the shape of starting point distribution showed consistent effects in SAT differences, even though the parameters themselves were affected by SAT manipulations.

Table 7.4: Correlation coefficients (p values) between differences in parameter values between speed and accuracy conditions between tasks in session 1 (outer left) and session 2 (inner left) and between sessions in the flanker task (inner right) and the Stroop (outer right) tasks. Significant correlations at the level of 0.0035 are displayed in bold (adjusted for multiple comparisons).

Parameter	Between task S1	Between task S2	Between session flanker	Between session Stroop
β_1	**0.42** (0.001)	**0.50** (3.7×10^{-4})	**0.51** (2.6×10^{-4})	**0.54** (9.8×10^{-5})
μ_R	0.37 (0.006))	0.41 (0.004)	**0.55** (6.2×10^{-5})	**0.60** (7.3×10^{-6})
σ_R	0.19 (0.162)	-0.16 (0.277)	0.24 (0.112)	0.41 (0.004)
μ_C	-0.03 (0.837)	0.32 (0.028)	0.27 (0.064)	0.16 (0.284)
A	-0.04 (0.769)	-0.24 (0.109)	0.30 (0.041)	-0.01 (0.965)
τ	0.08 (0.548)	-0.10 (0.497)	0.40 (0.005)	0.16 (0.283)
α	0.25 (0.060)	0.24 (0.104)	0.08 (0.610)	0.39 (0.006)

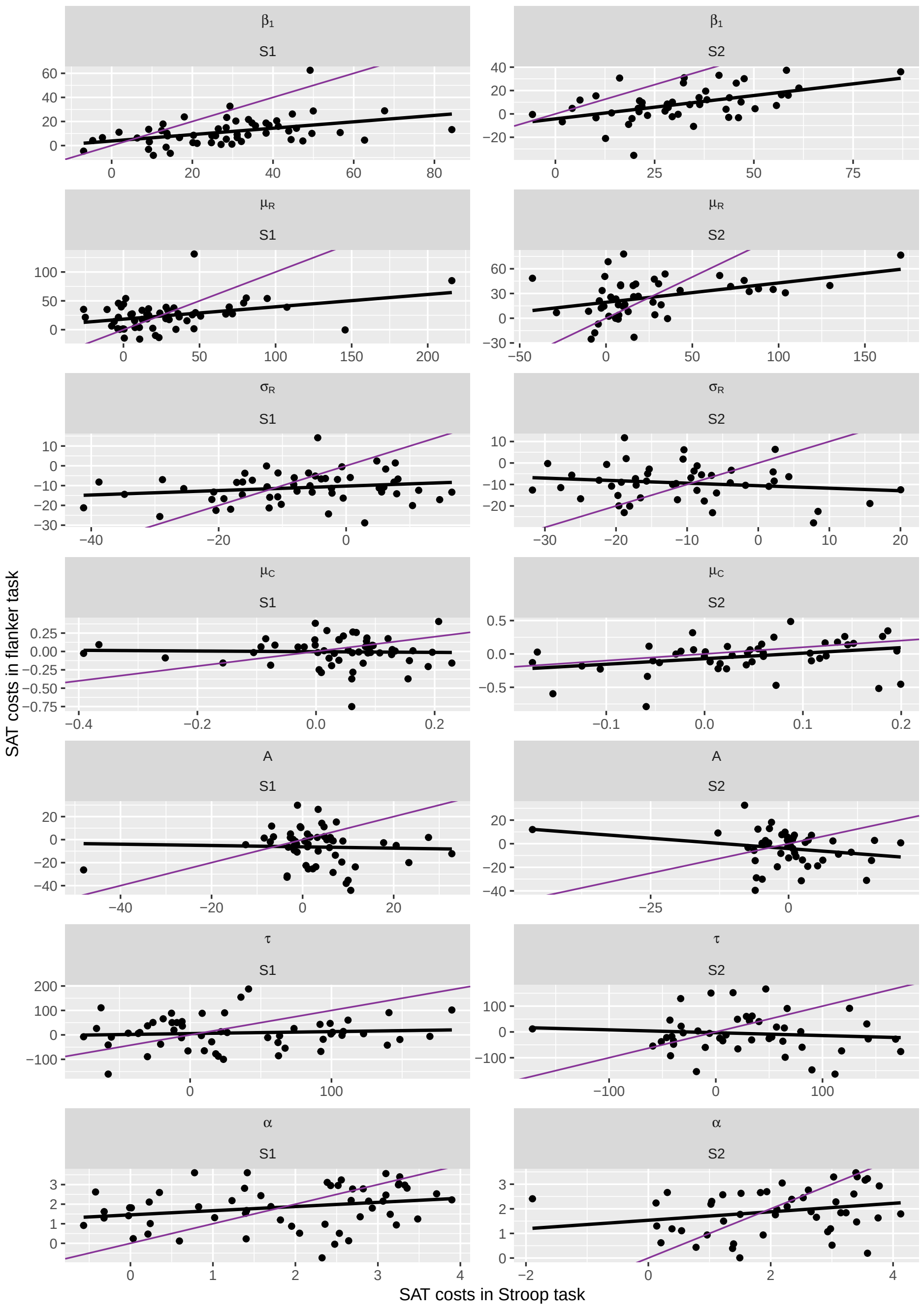


Figure 7.11: Correlations between SAT costs in model parameters (different rows) between two tasks for both session 1 (left column) and session 2 (right column). Black line indicates the best fit line for correlation, while the purple lines display identity lines.

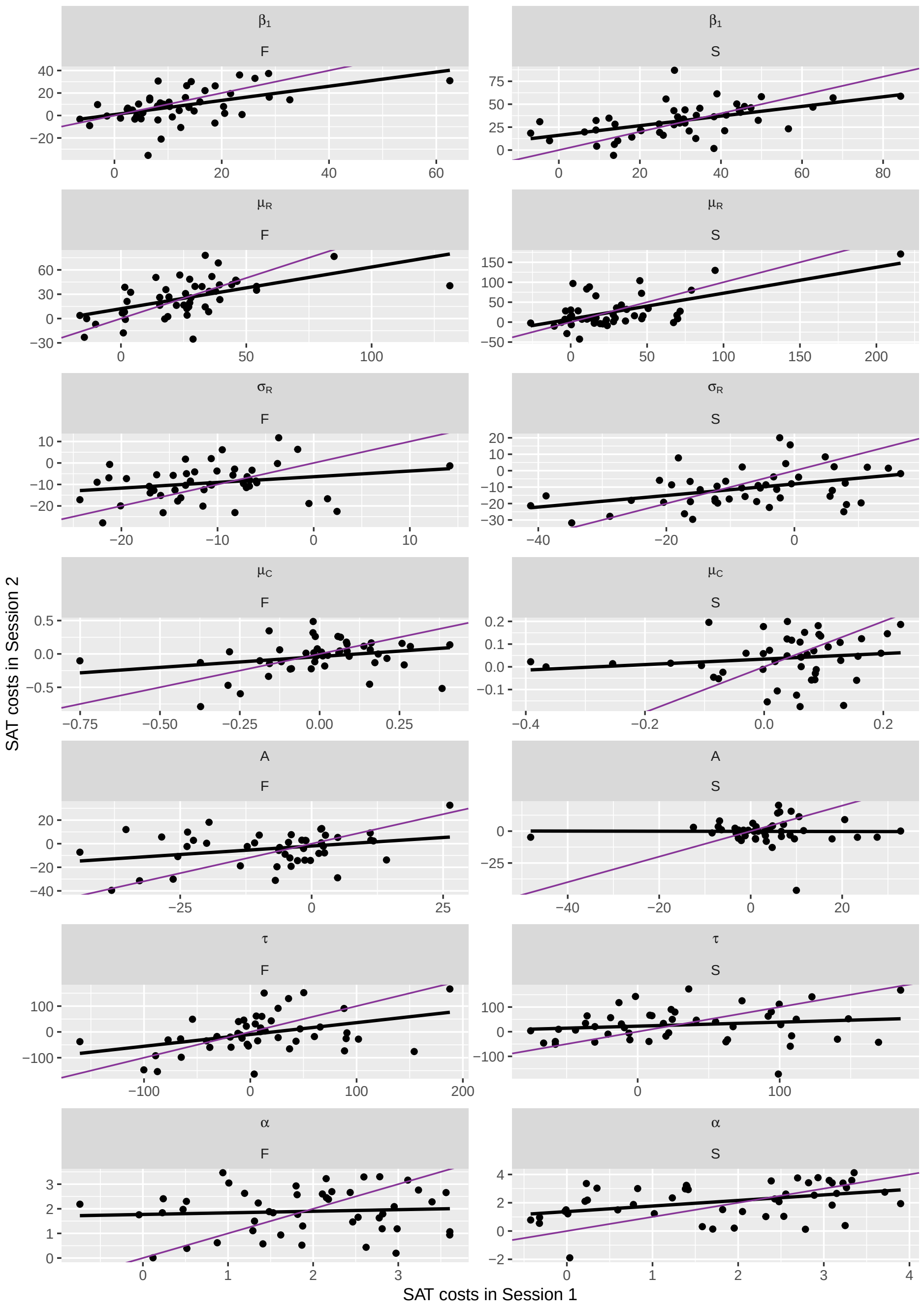


Figure 7.12: Correlations between SAT costs in model parameters (different rows) between two sessions for both flanker task (left column) and Stroop task (right column). Black line indicates the best fit line for correlation, while the purple lines display identity lines.

7.4 Discussion

In this chapter, the diffusion model for conflict tasks was applied to behavioural data collected from participants who performed Stroop and flanker task twice over two sessions separated by four weeks. The main purpose of investigating parameters from conflict diffusion model was to examine whether they provide any additional insights into questions from section 6 that were not present in behavioural data, namely, whether model parameters show more stability between response inhibition tasks and between experimental sessions than behavioural measures; on top of examining how speed-accuraccy trade-offs affect model parameters in response inhibition tasks.

The results showed that out of seven parameters from the diffusion model for conflict tasks that were recovered from experimental data, four were affected by SAT manipulation: upper threshold, non-decision time, variability of non-decision time, and shape of starting point distribution. This suggests that similarly to the drift diffusion model, the SAT affects more than just boundary separation (upper threshold), and has significant effects on other parameters in the diffusion model for conflict tasks as well. The idea that SAT only affects boundary separation has been challenged by many findings, for example, J. Zhang and Rowe (2014); Rae et al. (2014); Dambacher and Hübner (2015), who all found effects of SAT on non-decision time, drift rate, or both, on top of the expected variations in boundary separation. The results of the current study however did not find any differences in drift rate (for controlled process), which could be a mark of two aspects: the current study used response inhibition, rather than perceptual decision making tasks; as well as using the diffusion model for conflict tasks, rather than the drift diffusion model, which provides additional parameters to account for the decision making process. This result was replicated in two different tasks, as well as two different experimental sessions, making it unlikely to be a result of statistical noise.

The finding that SAT affects non-decision time in response inhibition tasks using the diffusion model for conflict tasks is consistent with the literature from sequential sampling models that use perceptual decision making. This study however expands the findings by revealing differences in all non-decision process related parameters: variability of non-decision time and the shape of starting point distribution. The impulsivity parameters of the diffusion model for conflict tasks, the amplitude of automatic activation, and the time to peak automatic activation, were not affected by SAT manipulation in this study, suggesting that speed and accuracy pressures do not alter processes of decision making that are automatic to the stimuli.

Applying the diffusion model for conflict tasks to two response inhibition tasks allowed investigation into whether model parameters show stability between the two tasks, as behavioural measures in section 6 showed no relationship between the two tasks. Replacing behavioural measures with model parameters when investigating the relationship between the Stroop and the flanker tasks did not deliver great results: there were some correlations between model parameters, namely upper threshold (boundary separation) and non-decision time, as well as a few other spurious looking correlations that did not replicate between the two experimental sessions. SAT manipulation only brought very minimal advantage over using neutral instructions that emphasize both speed and accuracy. There was no relationship between the impulsivity parameters in the two tasks.

In section 6, all of the behavioural measures from the flanker task, namely reaction time costs and accuracy costs, correlated between the two experimental sessions. Therefore it was not unexpected that all parameters from the diffusion model for conflict tasks correlated between sessions in the flanker task as well, including the impulsivity parameters. The manipulation of SAT, just like in section 6, did not seem to add much benefit to between-session correlations, as the relationships between model parameters did not improve with either speed of accuracy emphasis. The results from the Stroop task were different to those of the flanker task, in that only a few model parameters, like boundary separation (upper threshold), non-decision time, and drift rate (for controlled process), correlated between sessions. However, it must be noted that none of the correlations between model parameters in the Stroop task, under neutral instructions that emphasise both speed and accuracy, were negative or zero. This suggests that model parameters are stable within the same response inhibition task, but they might be more difficult to reveal in some tasks compared to others.

The limitations regarding the use of the four-choice Stroop task that were discussed in section 6 apply to an even greater extent here, as the drift diffusion model and subsequently the diffusion model for conflict tasks, are both two-alternative-choice models. As the Stroop task in the current study had four alternative choices, it is unknown how well the model is capable of explaining the performance in the task as if it were a two-choice task. However to fit the model, the responses were coded as binary in a correct vs incorrect choice manner. This overall should not make the application of the model itself to the task invalid, as Krajbich and Rangel (2011) found that the brain uses similar computational processes to make binary and trinary choices by applying extension of the drift diffusion model, but it might affect the recovery of the parameters to some extent anyway. This could explain the worse correlations between parameters between sessions in the Stroop task compared to the standard two-choice flanker task, and also a lack of correlations between the two response inhibition tasks.

Application of the diffusion model for conflict tasks to experimental data in this chapter was very easy, which was the result of using trained deep learning models from section 5 to recover parameters from the diffusion model for conflict tasks, using very little compute resources. The whole process took no more than a few minutes, to extract parameters from 612 individual sets of data. If traditional global optimization algorithms were used instead, the process would have taken over one and a half years with significantly more compute resources. The application of deep learning models allowed not having to put any constraints on the data itself, like presuming that some parameters should be stable between sessions as they come from the same individuals, instead, the data was allowed to speak for itself and reveal the similarities if there were any.

Even though deep learning models struggled to recover the impulsivity parameters from the diffusion model for conflict tasks in section 5 well, the impulsivity parameters seemed to remain stable in time in the flanker task. This suggests that the deep learning models were able to fit more than noise in the data, as it is very unlikely that noise patterns in individuals would remain stable in time. Nonetheless, the results regarding impulsivity parameters should be interpreted with caution until the deep learning models are better able to recover those parameters.

The correlations between parameters from the diffusion model for conflict tasks from different tasks might reflect similarities in decision making in these tasks that may or may not be related to response inhibition at all. For example, boundary separation is supposedly important in impulsive decision making (lower threshold - more impulsive decisions), whereas other parameters, like non-decision time, are general to any decision making, so correlations in this parameter would reflect similarities in decision making that are not related to impulsive behaviour. This will be investigated in section 8, by comparing performance in two tasks, the flanker task that has response inhibition, and a perceptual decision making task of motion coherence that does not have a response inhibition element.

8 Response inhibition and perceptual decision making with speed-accuracy trade-off

8.1 Introduction

In order for response inhibition tasks to be useful as an endophenotype of impulsive behaviour or impulsive disorders, they need to be stable over time and reflect some aspects of impulsive decision making. Section 6 and section 7 have shown that the performance in the flanker task is relatively stable over time, however, they have also demonstrated how the performance of the task is affected by manipulations in speed and accuracy pressures. The effects of SAT on simple decision making tasks without impulsivity elements are well established, in that responses are more accurate but slower under accuracy pressures and conversely faster but more error prone under speed pressures (see Heitz (2014) for a review). The exact same pattern of behaviour is also found in the sparse literature on the effects of SAT on response inhibition tasks (Leotti & Wager, 2010; Wylie et al., 2009; Van Wouwe et al., 2014). The studies by (Wylie et al., 2009) and (Van Wouwe et al., 2014) also demonstrated how speed pressure affects Parkinson's patients and healthy controls differently in the interference effects of response inhibition tasks, suggesting that SAT strategies have a larger effect in response inhibition tasks in individuals who are more prone to impulsive decision making. However all response inhibition tasks have elements that overlap with general decision making tasks without the response inhibition aspect. This poses a question: does speed-accuracy trade-off affect just the general decision making process, or does it also affect impulsive decision making specifically?

Investigating this question in the cognitive modelling framework is beneficial, as it allows decomposing the performance in decision making and response inhibition tasks into individual cognitive elements, some of which are universal across decision making (e.g. non-decision time and drift rate), while others are specific to response inhibition (e.g. amplitude of automatic activation and time to peak automatic activation). As the diffusion model for conflict tasks by Ulrich et al. (2015), applied in section 7, is an adapted version of the drift diffusion model, it makes comparison between model parameters from response inhibition and simple decision making tasks very straightforward.

In section 6 we have demonstrated how behavioural measures, namely reaction time costs and accuracy costs, did not correlate between two response inhibition tasks - the flanker task (Eriksen & Eriksen, 1974) and the computer version of the Stroop task (Stroop, 1935). Inspection of parameters from the diffusion model for conflict tasks in section 7 revealed very few correlations between model parameters. While adding speed and accuracy pressure to participants did not affect the correlations in section 6, there were some effects in section 7 where correlations between some parameters were either only present under speed or accuracy, but not both speed and accuracy pressures, or were slightly larger with SAT manipulations. However it is unknown how much the complexity of the Stroop task affected these results. The task was a four-choice rather than a two-choice task, with a lot more complex response mappings than the flanker task. Rather than trying to find which, if any, response inhibition tasks display similarities in either behavioural or cognitive construct measures between the tasks, we can first try to establish whether there are similarities in task performance irrespective of whether the tasks contain response inhibition elements or not, and whether SAT manipulations help to reveal them.

This is going to be investigated in the current chapter. The research presented here had participants perform two tasks, one response inhibition (Eriksen flanker, (Eriksen & Eriksen, 1974)), and one perceptual decision making task (random dot kinematogram, RDK) (Britten, Shadlen, Newsome, & Movshon, 1992) without the response inhibition element. Both tasks contained same speed-accuracy instruction manipulations as in section 6. The two tasks were chosen as they were relatively similar, but one without response inhibition element, and one with. Both flanker and RDK tasks have similar stimulus response bindings (stimulus either goes left or right, responses are made with left or right finger respectively), they are both two-choice tasks, and RDK allows for varying difficulty to mimic congruency without the inhibition element. Firstly, we expect to see the SAT manipulations have an effect on both the behavioural measures of the two tasks, as well as some model parameters (upper threshold, non-decision time, variability of non-decision time, and the shape of the starting point distribution). As section 7 did not find that SAT affected the drift rate for controlled process in the response inhibition tasks, we should also not observe this in the flanker task in this chapter. However, we should see coherence levels affect the drift rate for controlled process in the RDK task. However it is unclear whether SAT manipulations will further affect the drift rate for controlled process parameter. Secondly, we will be able to investigate similarities between the flanker task and the RDK task in the behavioural measures and the model parameters. If both tasks rely on similar decision making processes, then we would expect to see a correlation between the accuracy costs and the reaction time costs between the two tasks, provided that the behavioural measures are sensitive enough to reveal the decision making processes. The same relationship can be investigated by inspecting model parameters, as they reveal decision making process in more detail than the behavioural measures. The speed and accuracy manipulations, as in section 6 and section 7, might help to reveal these relationships as they should provide more range in both the behavioural measures, as well as the model parameters.

8.2 Methods

8.2.1 Participants

Students from Cardiff University School of Psychology undergraduate courses were recruited to participate in this study in exchange of course credit. Eighty one participants (six males) completed the study. Six participants that participated in Study 1 also participated in this study. All participants had normal or corrected-to-normal vision, and reported no motion sickness when viewing moving visual stimuli as recruitment criteria (due to random dot kinematogram task). Participant characteristics are reported in table 8.1.

Table 8.1: Demographics of participants in the study

	Participants	Mean Age (Range)	Handedness
Male	6	19.3 (18-20)	6(100%) right handed
Female	75	19.3 (18-26)	64(85%) right handed
Total	81	19.3 (18-26)	70(86%) right handed

8.2.2 Design

The design consisted of within-subjects instruction condition (speed, accuracy, or both speed and accuracy emphasis), and within-subjects trial congruency condition (congruent, incongruent, or neutral trials in the flanker task, and hard or easy trials in the RDK task). All participants completed both tasks, but they were not analysed as factors.

8.2.3 Apparatus

The experiment took place in a dimly lit room; it was run on a Mac Mini computer connected to a 36.5cm by 27.5cm colour display (60Hz, resolution 1280px by 1024px) and an external keyboard. The tasks were programmed in PsychoPy version 1.73.04 (Peirce, 2009). Participants were seated at a distance of about 60 cm in front of the computer screen. Responses were made on keyboard by pressing the 'z' and 'm' keys for left and right responses respectively.

8.2.4 Tasks and procedures

Two tasks were used in this study. The first task was a response inhibition task, already used in first study. This was the Eriksen flanker task (Eriksen & Eriksen, 1974), where participants saw a white arrow in the middle of the black screen. The central arrow was flanked with either two other arrows or dashes above and two other arrows or dashes bellow the central arrow. In total five objects were presented in each trial. In congruent trials, the direction of flanker arrows matched the direction of the central arrow. In the incongruent trials, the direction of the flanker arrows was opposite to that of the central arrow. In neutral trials, the central arrow was flanked by dashes. In half of the trials, the central arrow was pointing right, while in the other half is was pointing left. There were 144 trials per block, 48 congruent, 48 incongruent, and 48 neutral.

The second task used was a random dot kinematogram (RDK) task (Britten et al., 1992). RDK is a two-choice perceptual decision-making task, which requires participants to decide whether a cloud of dots is moving to the left or the right of the screen. In this task, white dots were presented in a central oval in the middle of a black screen. There were 50 dots in the central oval, with proportion of the dots displaying a random motion, while the rest moved coherently to the right or to the left of the screen. There were two types of motion coherence - 0.15 (hard condition) and 0.30 (easy condition), as in the hard condition fewer dots displayed a coherent motion pattern, while in the easier condition more dots moved with the same pattern. The signal dots were moving left in half the trials and right in other half of the trials. There were 120 trials per block, 60 hard and 60 easy. The details of the task are also provided in Hedge, Powell, Bompas, et al. (2018).

There were three types of blocks in both tasks, same as in section 6. The first type of block asked participants to respond as fast as possible, the second type of block instructed participants to respond as accurately as possible, while the last type of block asked participants to be both fast and accurate. Different feedback was given in these blocks. In speed blocks, participants received feedback if they responded too slow (feedback said "Too slow" on the screen) (slower than 600ms for the RDK task and 500ms for the flanker task), in accuracy blocks participants saw feedback saying "Incorrect" if they gave the wrong answer. In neutral blocks, no feedback was given. If participants responded too quickly (faster than 200ms for the RDK task and 150ms for the flanker task), they saw feedback saying "Too fast" irrespective of the type of block. The presentation of blocks within a task was random.

Participants completed four questionnaires after completing the tasks. The first was UPPS-P questionnaire (Lynam et al., 2006), the second questionnaire was NEO-FFI (Costa & MacCrae, 1992). Then two questionnaires on compliance were administered. First, the Gudjonsson Compliance Scale (Gudjonsson, 1989), is intended to measure trait compliance. This questionnaire consists of 20 statements, to which participants respond by choosing "True" or "False". Finally, a Situational Compliance Scale (Gudjonsson, Sigurdsson, Einarsson, & Einarsson, 2008) was used to assess participants' compliance as a state measure. No analysis was performed on the questionnaire data in this thesis.

Participants took 1.5 hours to complete the session, with flanker task taking approximately 40 minutes to complete, RDK task – approximately 40 minutes, and the questionnaires – approximately 10 minutes.

8.2.5 Data preparation

Before any analysis, data was cleaned by removing responses that were longer than 1.5 seconds, and anticipatory responses (<150ms). One participant was removed from the analysis of the flanker task as they had 19% responses removed from such cleaning. All of the remaining participants had some trials removed - the values ranged from 0.6% to 14.7% of trials per participant, mean 2.8%. No other trials were removed. All analyses relating to reaction times only report reaction times of the correct responses, while measures relating to accuracy looked at both correct and incorrect responses.

8.2.6 Model fitting

Experimental data was separated by participant, task, SAT condition, and coherence condition for the RDK task. This resulted in 726 sets of data that were fitted: $((81 \times 2 + 80) \times 3)$, where 81 was number of participants in RDK, two coherence conditions, and 80 was participants in flanker, 3 was the number of SAT manipulations. In total, seven parameters were fitted (upper threshold, non-decision time, standard deviation of non-decision time, drift rate for controlled process, amplitude of automatic activation, time to peak automatic activation, and shape of starting point distribution) for the flanker task, and five parameters (upper threshold, non-decision time, standard deviation of non-decision time, drift rate for controlled process, and shape of starting point distribution) for the RDK task, while the shape of automatic activation was fixed to 2 and the diffusion constant was fixed to 4.

Deep learning models were used for parameter fitting. The models that we used were the same as in section 5, the models trained with reaction time distributions with low number of trials. For this, the input features were extracted from each dataset, then scaled according to the scaler from training dataset containing reaction time distributions with low number of trials, and then fitted individually for each diffusion model for conflict tasks parameter.

The models trained in section 5 could only be used with the flanker task as is, as they were trained with input features that contain congruent, incongruent, and neutral trials. As the RDK task only has neutral trials, the deep learning models were retrained to only use those trials as input features. The training of the models and the architecture of the models was identical to section 5.2. The only difference was that only the neutral input features were used to train the deep learning models. The resulting deep learning models trained with neutral input features were then applied to recover parameters from the RDK task, from low coherence and high coherence datasets individually.

8.3 Results

8.3.1 Does SAT affect performance in flanker and RDK tasks?

In order to see whether SAT affected the task performance, the reaction times, the accuracy of responses, the reaction time costs, and the accuracy costs in both flanker and RDK tasks were inspected. The results relating to the reaction times are shown in figure 8.1 and to accuracy of responding - in figure 8.2. The figures show that the participants were generally faster and more accurate in the flanker task than in the RDK task, confirming that the RDK task was slightly more difficult. The effects of SAT are also clearly visible in the figures: participants in both tasks were generally faster under speed instructions than under accuracy instructions, but more accurate under accuracy instructions than under speed instructions. The effects of trial congruency (and coherence in the RDK task) are also visible: the participants were faster and more accurate in congruent trials compared to incongruent trials.

The interaction between congruency/coherence and SAT manipulation was also tested, and the results are shown in table 8.2. The results indicate that in both flanker and RDK tasks and for both measures, interaction of SAT and congruency/coherence had significant effects on performance.

Table 8.2: F values (p values) for the SAT x congruency interaction on reaction time and accuracy, and SAT effects on reaction time costs and accuracy costs. Significant results are shown in bold at the level of 0.006 (corrected for multiple comparisons). Stars next to p values indicate Greenhouse-Geisser correction for Mauchy's test of sphericity. The degrees of freedom were (2, 160) for session 1 and (4, 316) for session 2 for interaction effects, and (2, 160) for session 1 and (2, 158) for session 2 for the cost effects.

Task and session	Reaction time	Accuracy	Reaction time costs	Accuracy costs
Flanker	**29.7** (1.8×10^{-14})*	**5.5** (0.003)*	**30.5** (2.9×10^{-10})*	**7.7** (0.002)*
RDK	**50.8** (8.4×10^{-18})	**13.2** (5.1×10^{-6})*	**50.8** (8.4×10^{-18})	**13.2** (5.1×10^{-6})

The congruency costs were also examined: the difference in reaction times and accuracy between congruent and incongruent trials in the flanker task, and the difference between low coherence and high coherence trials in the RDK task. The results are shown in

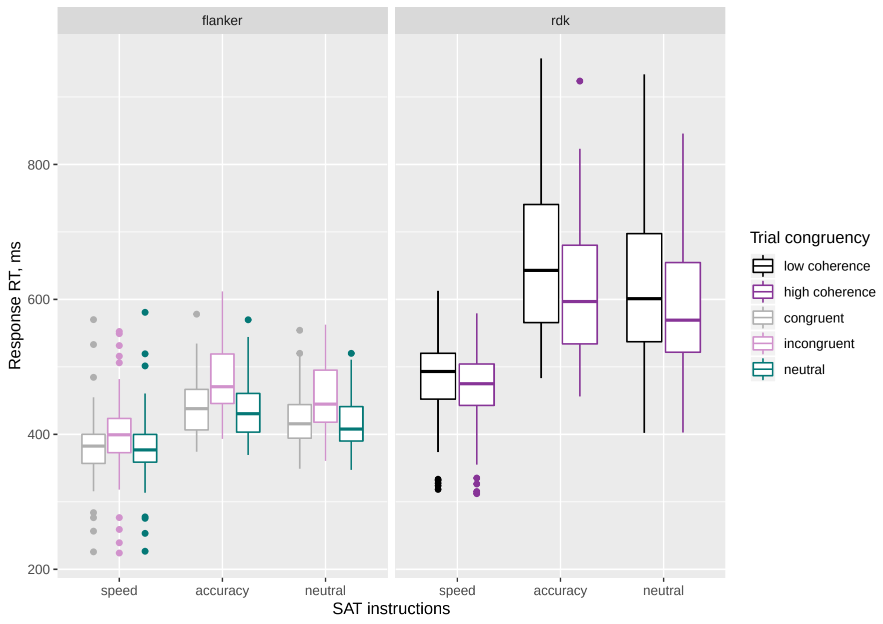


Figure 8.1: Effects of congruency/coherence (colour) and SAT instructions (x axis) on reaction times in both flanker (left column) and RDK (right column) tasks. “Both fast and accurate” instructions are reported as “neutral”. Dots indicate outliers.

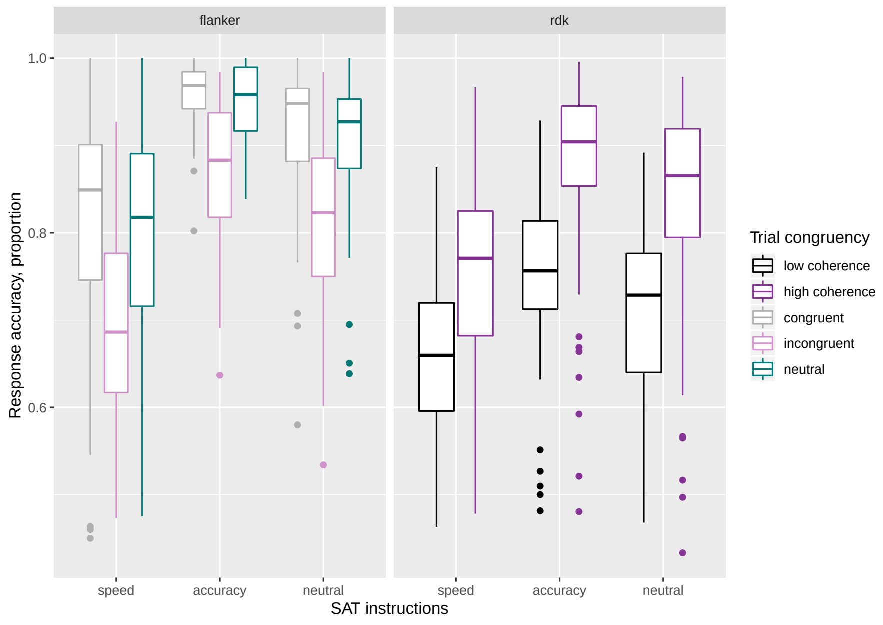


Figure 8.2: Effects of congruency/coherence (colour) and SAT instructions (x axis) on accuracy of responding in both flanker (left column) and RDK (right column) tasks. “Both fast and accurate” instructions are reported as “neutral”. Dots indicate outliers.

figure 8.3 for reaction time costs and figure 8.4 for accuracy costs, for both tasks. The figures show that congruency costs in reaction time were lower in speed conditions when compared to accuracy condition, whereas the congruency cost in accuracy was higher in speed conditions when compared to accuracy conditions, but only in the flanker task. The accuracy costs in the RDK task were lower in the speed condition than in the accuracy condition. These effects were statistically significant for all tasks (see table table 8.2 inner right column for reaction time costs and outer right column for accuracy costs).

8.3.2 Does task performance correlate between response inhibition and perceptual task?

As we have shown that manipulating SAT with instructions has an effect on performance in the flanker and the RDK tasks, we can now investigate whether SAT manipulations also affect the correlations between the measures from the two tasks. The correlations for reaction time costs are shown in figure 8.5 and for accuracy costs in figure 8.6. The figures show that during both speed and accuracy and accuracy only instructions, correlations in both reaction time costs and accuracy costs were very low. They are reported in table 8.3. The table shows that reaction time costs did not correlate between the two tasks in any of the SAT condition, however the accuracy costs had a small but significant correlation between the two tasks but only in under speed pressure. This suggests that SAT manipulations help to reveal similarities between performance in two tasks.

Table 8.3: Correlations (p values) between reaction time and accuracy costs between the RDK and the flanker tasks. Correlations significant at the level of 0.008 (adjusted for multiple comparisons) are shown in bold.

SAT condition	Reaction time costs	Accuracy costs
Speed	0.20 (0.069)	**0.33** (0.002)
Accuracy	0.10 (0.390)	0.11 (0.348)
Neutral	0.06 (0.564)	0.20 (0.078)

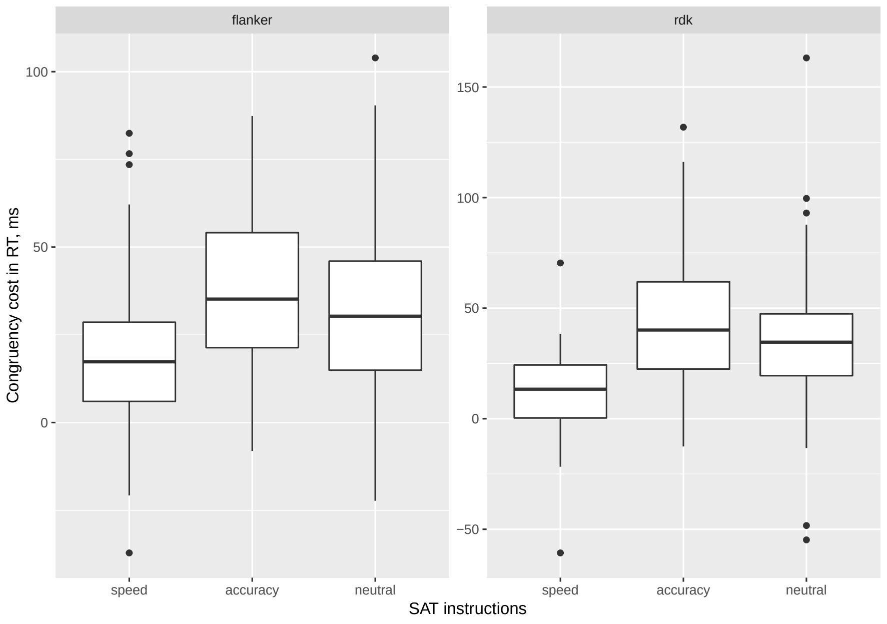


Figure 8.3: Effects of SAT instructions on reaction time costs in both flanker (left column) and RDK (right column) tasks. “Both fast and accurate” instructions are reported as “neutral”. Dots indicate outliers.

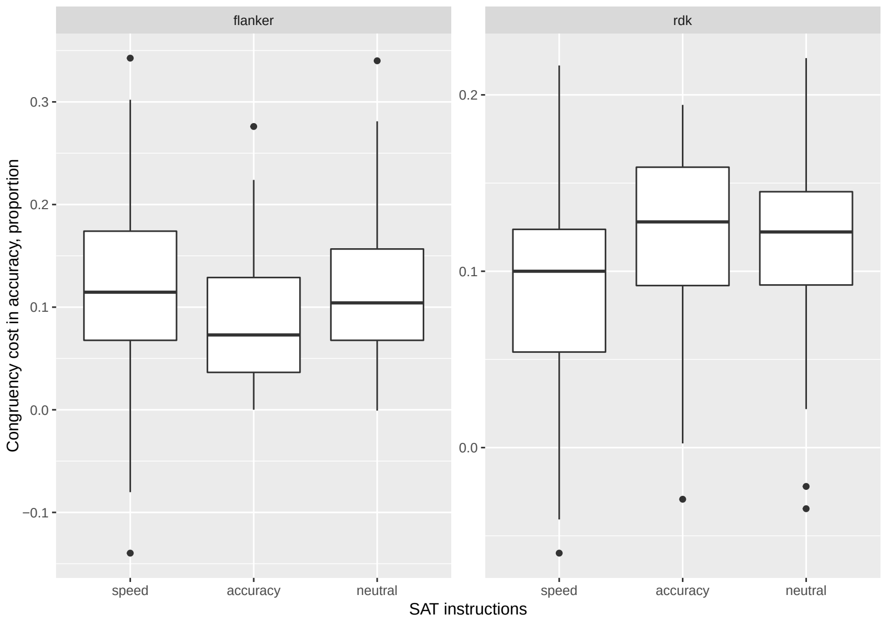


Figure 8.4: Effects of SAT instructions on accuracy costs in both flanker (left column) and RDK (right column) tasks. “Both fast and accurate” instructions are reported as “neutral”. Dots indicate outliers.

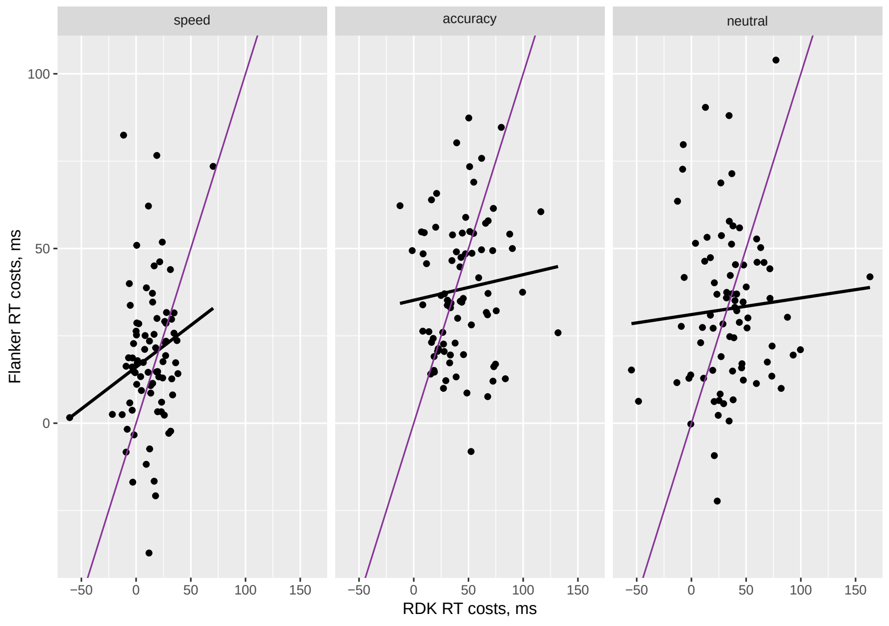


Figure 8.5: Correlations between reaction time costs in flanker and RDK tasks for different SAT instructions (columns). Black line indicates the best fit line for correlation, while the purple lines display identity lines.

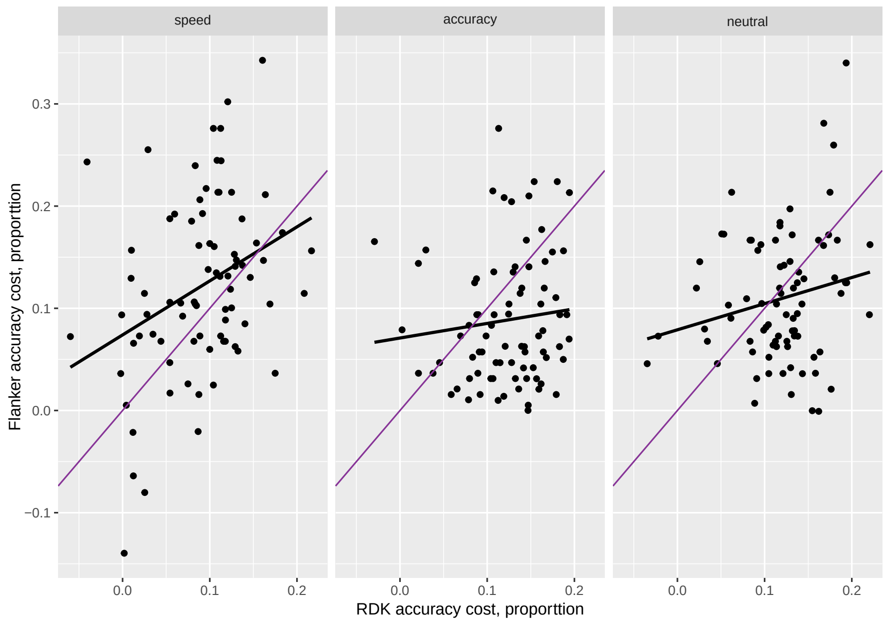


Figure 8.6: Correlations between accuracy costs in flanker and RDK tasks for different SAT instructions (columns). Black line indicates the best fit line for correlation, while the purple lines display identity lines.

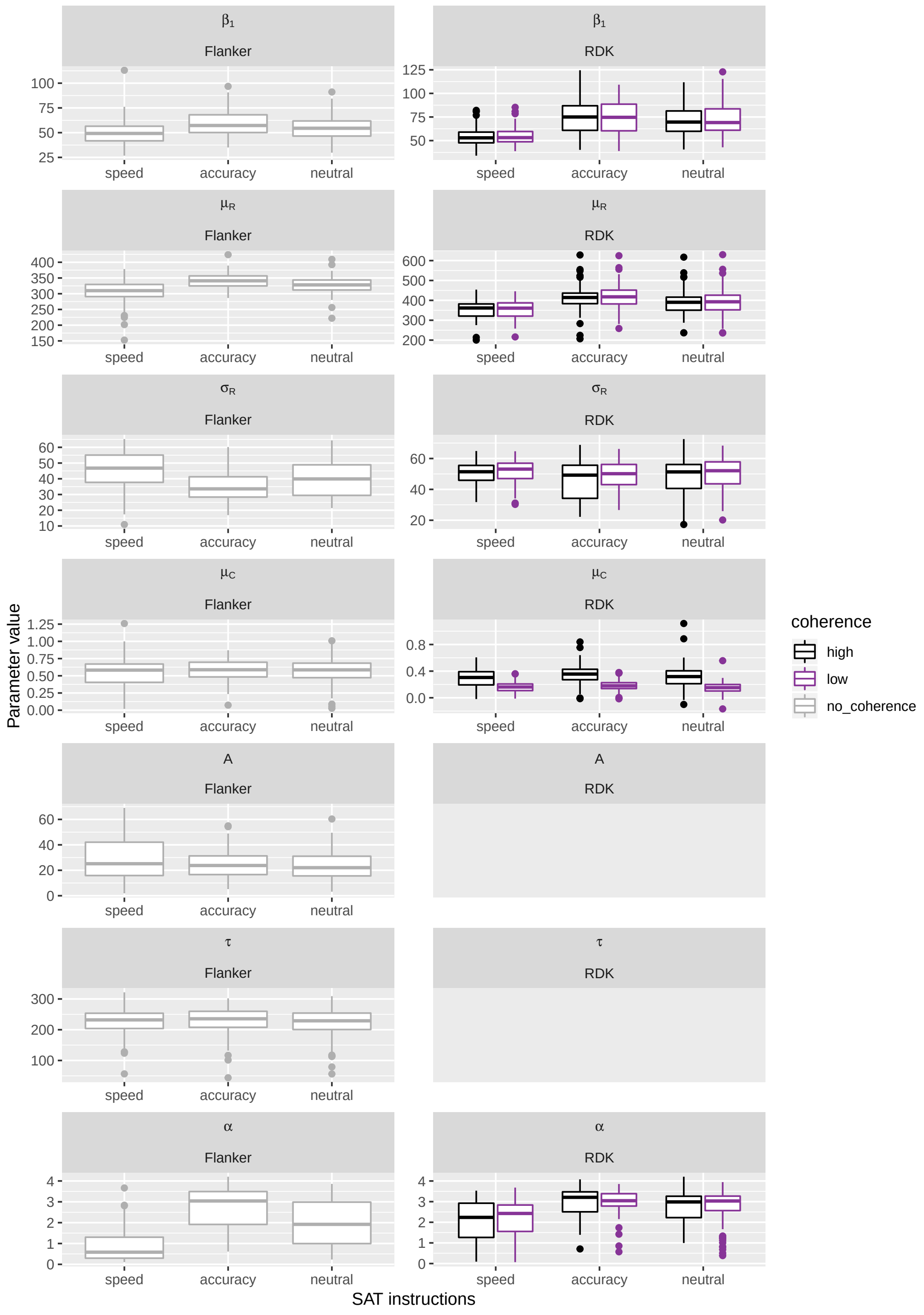


Figure 8.7: Diffusion model for conflict tasks parameters in flanker (left column) and RDK (right column) tasks. Parameters from different coherence levels in the RDK task are represented as different colours. A and *tau* parameters are empty in the RDK task as they only apply to response inhibition tasks. “Both fast and accurate” instructions are reported as “neutral”. Dots indicate outliers.

8.3.3 SAT costs between tasks

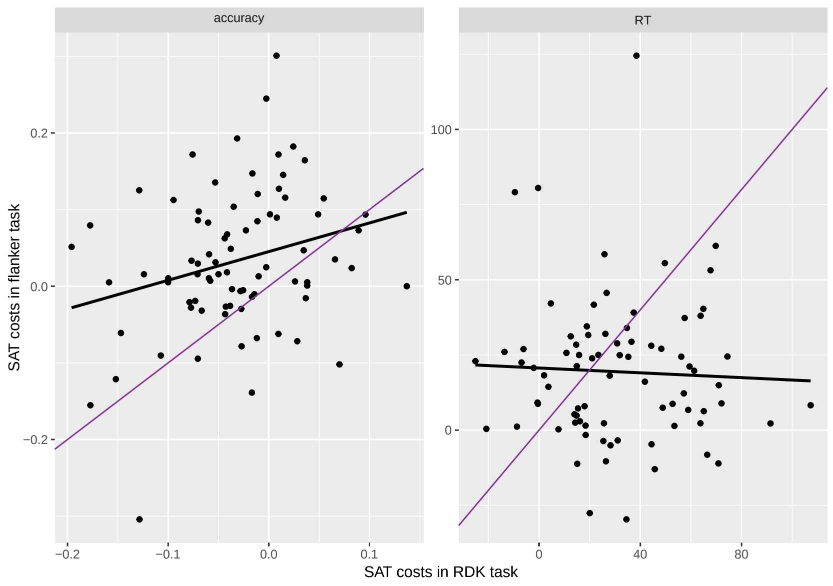


Figure 8.8: Correlations between SAT costs in reaction time costs (right column) and accuracy costs (left column) between two tasks. Black line indicates the best fit line for correlation, while the purple lines display identity lines.

The last question that was investigated in this chapter was whether the differences between reaction time and accuracy costs between the speed and accuracy instructions displayed stability between the two response inhibition tasks. The results for SAT costs between the two tasks are shown in figure 8.8 . The figures show that there were was very little relationship between the SAT costs between the flanker and the RDK task. The actual correlations were r(78) = -0.04, p = 0.694 for reaction time costs and r(78) = 0.26, p = 0.019 for accuracy costs, with the correlation between accuracy costs being significant. This is inconsistent with results from section 6, which showed a correlation in reaction time costs. It must be noted however that the correlations were low and could have been driven by outliers, both here and in section 6.

8.3.4 How do model parameters relate in an impulsivity and non-impulsivity task?

To ensure that deep learning recovered the parameters from the experimental data well, the model parameter sets were used to produce RT distributions of 1000 trials from each of congruent, incongruent, and neutral conditions for the flanker task, and neutral trials but with two coherence levels for the RDK task. The accuracy, and the 25th, 50th, and 75th quantiles of RT were calculated for both correct and incorrect responses. Same measures were also calculated for the actual RT distributions obtained from participant responses. They are plotted in figure 8.9 and figure 8.10 for flanker task and RDK task respectively. The figures show that deep learning recovered the parameters very well as the predicted measures appear very closely matched to actual measures, with exception to very few participants. The predictions for RDK task are particularly of interest, given that the deep learning models were not tuned at all to predict parameters from the drift diffusion task, but simply retrained to predict parameters from neutral trials only. Even without extra tuning, the deep learning models seem to be doing a good job of predicting the model parameters. Note that individual dots in the figures represent an RT distribution for a SAT condition, so there will be three distributions (with speed, accuracy, and both speed and accuracy conditions) represented in each panel of the figures.

To examine whether model parameters are affected by SAT manipulation in the same way in two types of task, response inhibition and perceptual decision making, the diffusion model for conflict tasks parameters were predicted for the Flanker and the RDK task using deep learning models as described in the methods. As the RDK task does not have a response inhibition element, only five out of seven parameters were estimated, excluding the automatic activation ones. The predicted parameters are presented in figure 8.7.

Table 8.4: F values (p values) indicating the statistical difference between model parameter values from different SAT conditions for flanker and RDK tasks. Star after p value indicates Greenhouse-Geisser correction when Mauchy's test for sphericity was violated. The significant results are shown in bold (at corrected for multiple comparisons level of 0.004. Degrees of freedom were (2, 160) for RDK task and (2,158) for flanker task.

Parameter	Flanker task	RDK task
β_1	**21.2** (3.5×10^{-8})*	**86.3** (3.9×10^{-26})
μ_R	**58.6** (3.3×10^{-15})*	**63.5** (5.1×10^{-21})
σ_R	**46.4** (9.7×10^{-16})*	**7.5** (7.5×10^{-4})
μ_C	1.7 (0.192)*	**6.1** (0.003)
A	5.6 (0.007)*	-
τ	0.4 (0.667)	-
α	**129.5** (5.0×10^{-34})	**50.2** (1.2×10^{-17})

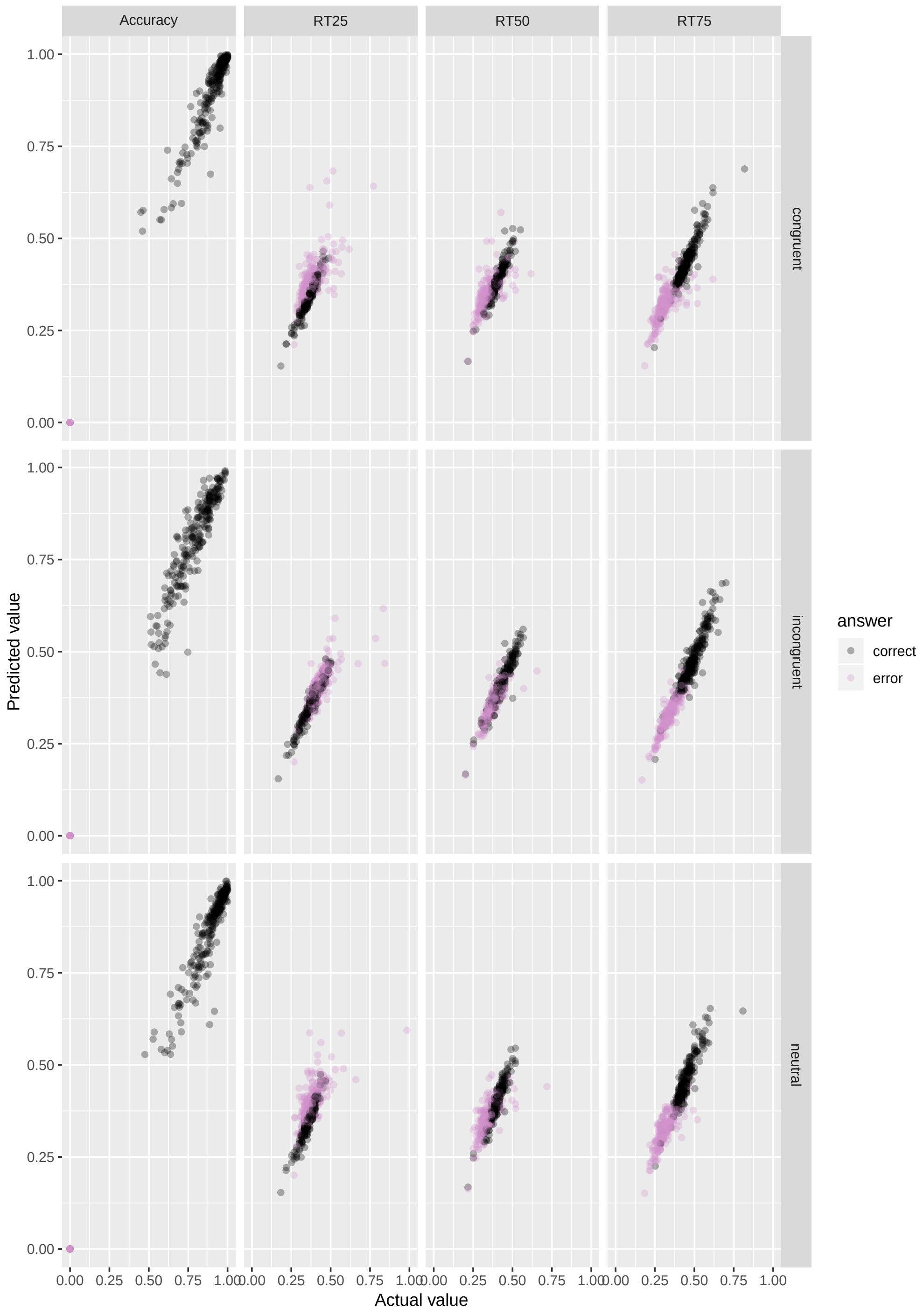


Figure 8.9: Predicted and actual accuracy, 25th RT quantile, 50th RT quantile, and 75th RT quantile (columns) of RT distributions for flanker task, for both correct (black dots) and incorrect (pink dots) responses. The rows separate the trials by congruency. The accuracy is measured in proportion, while the RT quantiles in seconds.

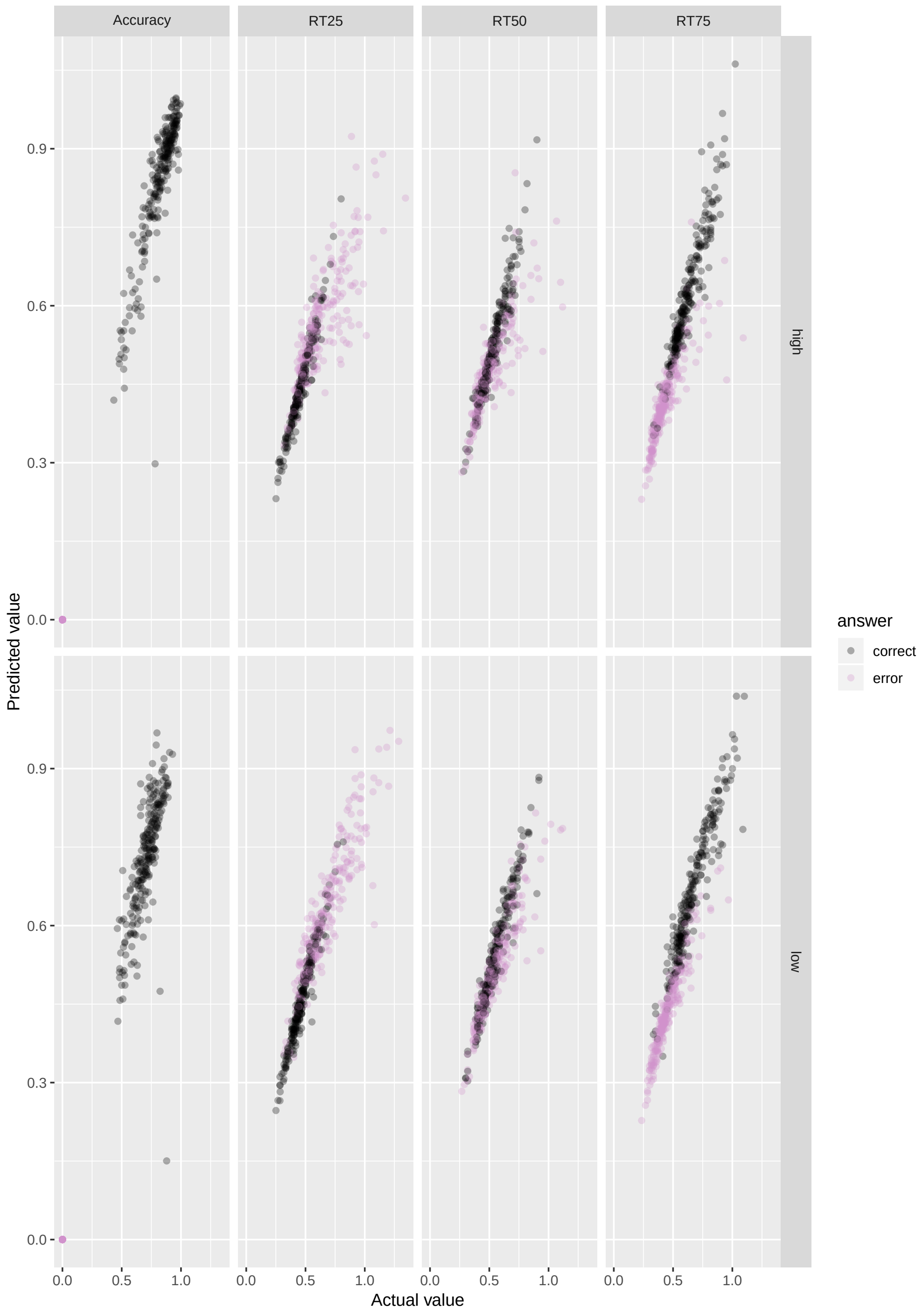


Figure 8.10: Predicted and actual accuracy, 25th RT quantile, 50th RT quantile, and 75th RT quantile (columns) of RT distributions for RDK task, for both correct (black dots) and incorrect (pink dots) responses. The rows separate the trials by RDK dot coherence. The accuracy is measured in proportion, while the RT quantiles in seconds.

As the differences between model parameter values are not important, the data from each parameter was evaluated separately. First, we want to know whether SAT has an effect on model parameters in the two tasks. The results are shown in table 8.4. The table shows a similar pattern of results as the previous chapter, in that SAT had a significant effect on upper threshold, non-decision time, variability of non-decision time, shape of starting point distribution, and mildly the drift rate for controlled process in the RDK task, but not flanker task. This is all consistent with the results in the previous chapter.

On top of main effects of SAT, we can also investigate which parameters are affected by coherence in the RDK task. We find that variability of non-decision time and drift rate for controlled process is affected by coherence (F(1,80)= 8.6, p = 0.004 for variability of non-decision time, and F(1,80) = 258.5, p = 8.9×10^{-27} for drift rate for controlled process). Only drift rate for controlled process would be statistically significant when adjusted for multiple comparisons. All other parameters were not affected by coherence (all $F < 3.7$, $p > 0.05$). There was also an interaction between SAT and coherence on the drift rate for controlled process (F(2,160) = 3.4, p=0.036), but it was not significant when adjusted for multiple comparisons (due to five different comparisons for each model parameter).

To see whether parameters in RDK correlate with parameters in the flanker task, the relationship between the two tasks was visualized, once for low coherence trials, and then second time for high coherence trials. The results are shown in figure 8.11 for low coherence trials and figure 8.12 for high coherence trials.

The correlation coefficients are reported in table 8.5 for both low coherence and high coherence correlations. The table demonstrates that all parameters correlate across the two tasks to some degree, however, the shape of the starting point distribution could be spurious as it did not replicate between the two coherence conditions. The upper threshold only correlated between the tasks in the speed and accuracy conditions, not the neutral condition, indicating that more extreme responding is needed to reveal the correlations. The finding remained the same for both coherence conditions. The non-decision time parameter correlated only in the neutral and speed condition, suggesting that faster responses reveal the correlations. The correlations between non-decision time seem stronger in the speed than in the neutral condition, suggesting that SAT is beneficial in revealing the relationship between the tasks. Variability of non-decision time seems to be correlated between the two tasks, but SAT manipulation does not seem to bring

Table 8.5: Correlation coefficients (p values) between parameter values from flanker task and RDK task for both low coherence (top) and and high coherence (bottom) trials. Significant correlations at level of 0.003 adjusted for multiple comparisons are shown in bold.

	Flanker task - RDK low coherence		
SAT instructions	Speed	Accuracy	Neutral
β_1	**0.33** (0.003)	**0.53** (5.3×10^{-7})	0.20 (0.084)
μ_R	**0.58** (2.3×10^{-8})	0.25 (0.027)	**0.36** (9.7×10^{-4})
σ_R	0.29 (0.008))	**0.48** (9.7×10^{-6})	**0.40** (2.4×10^{-4})
μ_C	**0.41** (1.5×10^{-4})	**0.38** (5.6×10^{-4})	**0.39** (4.0×10^{-4})
α	0.20 (0.082)	0.11 (0.349)	0.26 (0.018)
	Flanker task - RDK high coherence		
	Speed	Accuracy	Neutral
β_1	**0.37** (6.9×10^{-4}))	**0.56** (8.2×10^{-8})	0.22 (0.048)
μ_R	**0.60** (4.5×10^{-9})	0.26 (0.019)	**0.39** (3.4×10^{-4})
σ_R	**0.38** (6.5×10^{-4})	**0.40** (2.3×10^{-4})	**0.41** (2.1×10^{-4})
μ_C	0.31 (0.005)	**0.35** (0.002)	0.23 (0.044)
α	0.19 (0.096)	**0.34** (0.002))	0.16 (0.151)

any improvement. The same is true for drift rate for the controlled process, but only in low coherence condition, but no effect of SAT, suggesting that it is the coherence level rather than SAT revealing the relationship.

8.3.5 SAT costs in model parameters between tasks and between sessions

Results reported in section 7 revealed that change in the upper threshold and the non-decision time parameters between the speed and accuracy instructions correlated between two response inhibition tasks. Do the same parameters show relationship in stability between a response inhibition and a perceptual decision making task? The results are shown in figure 8.13. The figure demonstrates that there were was very little relationship between the SAT costs between the two tasks. The actual correlations are reported in table 8.6. The table shows that only the non-decision time parameter demonstarted consistent change between the SAT conditions, while all the other parameters, including the upper threshold, did not. Even the correlations in non-decision time were much lower compared to results in section 7.

Table 8.6: Correlation coefficients (p values) between differences in parameter values between speed and accuracy conditions between the two tasks. Data from high coherence trials in RDK tasks is displayed in the left column, and data from low coherence trials in the right column. Significant correlations at the level of 0.005 are displayed in bold (adjusted for multiple comparisons).

Parameter	High coherence trials	Low coherence trials
β_1	0.26 (0.018)	0.20 (0.083)
μ_R	**0.32** (0.004)	**0.33** (0.003))
σ_R	0.04 (0.702)	0.11 (0.329)
μ_C	0.09 (0.420)	0.22 (0.051)
α	0.05 (0.660)	0.13 (0.252)

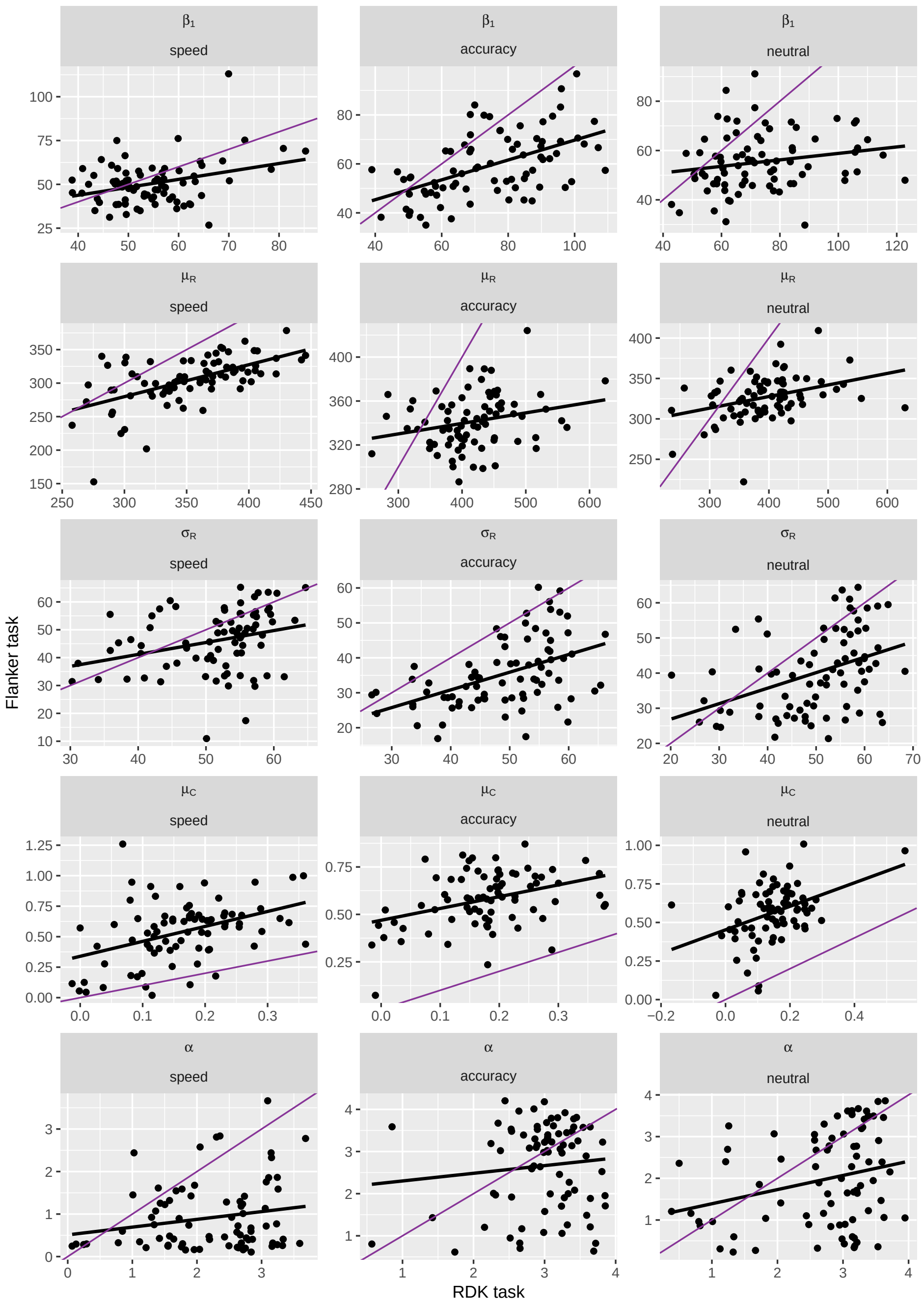


Figure 8.11: Correlations between parameter values in flanker task (y axis) and RDK low coherence trials (x axis) for different SAT instructions (columns). Black line indicates the best fit line for correlation, while the purple lines display identity lines.

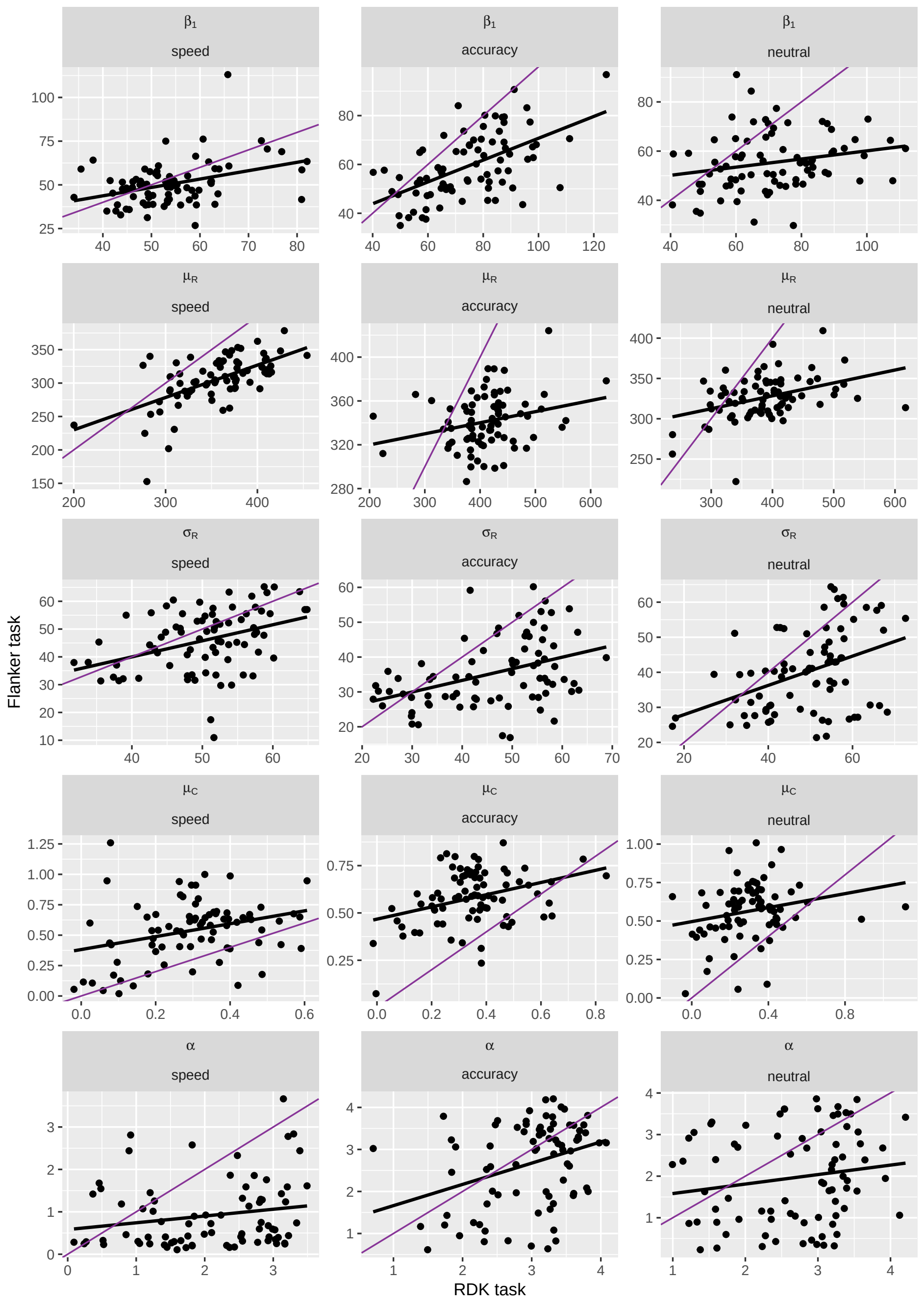


Figure 8.12: Correlations between parameter values in flanker task (y axis) and RDK high coherence trials (x axis) for different SAT instructions (columns). Black line indicates the best fit line for correlation, while the purple lines display identity lines.

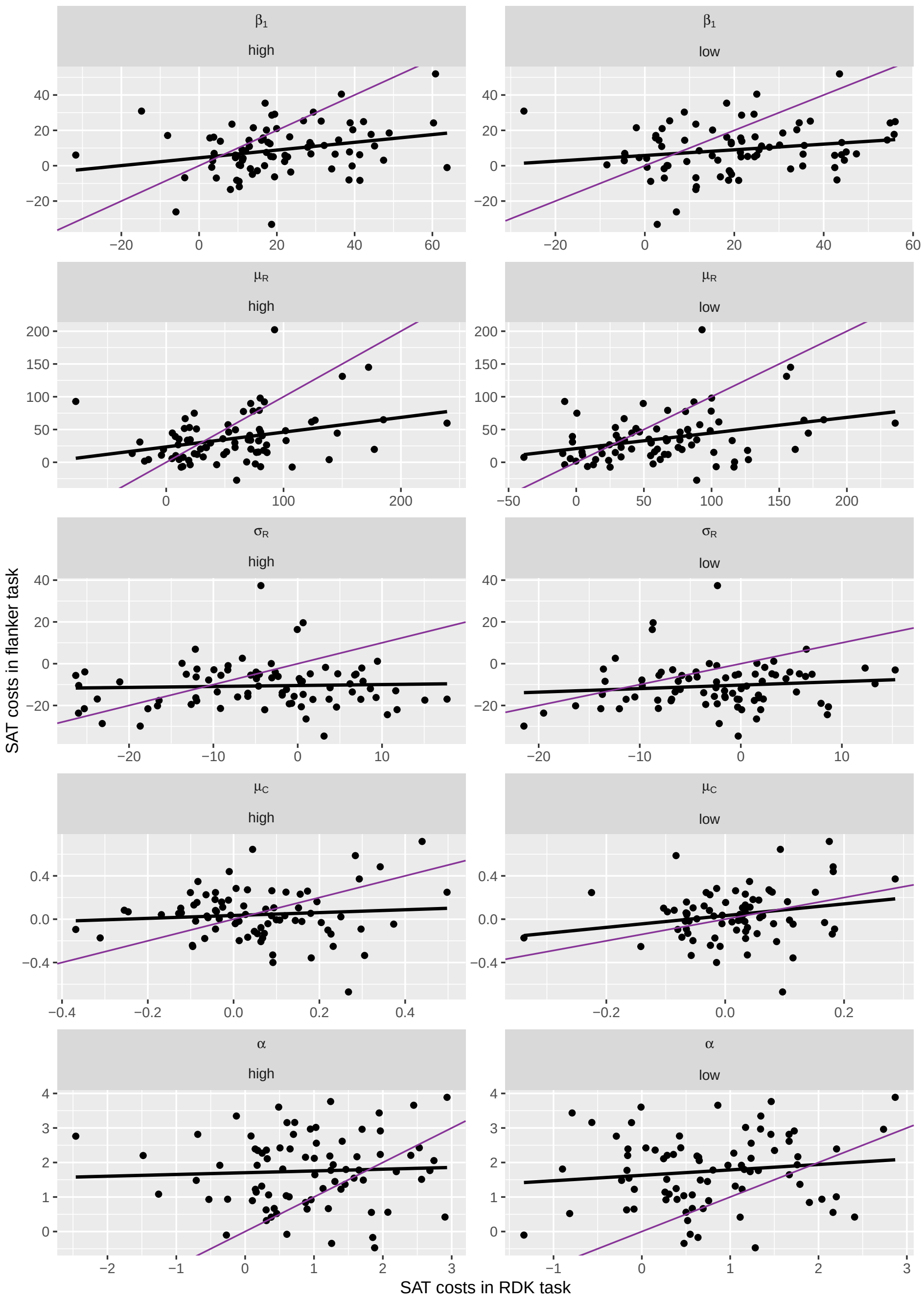


Figure 8.13: Correlations between SAT costs in model parameters (different rows) between two tasks for high coherence trials (left column) and low coherence trials (right column). Black line indicates the best fit line for correlation, while the purple lines display identity lines.

8.4 Discussion

The study presented in this chapter has replicated the results of section 6 by showing that speed-accuracy trade-off affects performance of the flanker task, as the participants changed their behaviour when instructions emphasize either speed or accuracy of responding. The participants were the most accurate but also the slowest in the accuracy condition; while under speed pressure, responses were faster but more error prone. These effects were also present in the task without response inhibition, the motion coherence task.

Given that section 6 found no correlations between the performance in two response inhibition tasks, even with SAT manipulations, we did not predict to find any relationship between performance in a response inhibition and a perceptual decision making task, but surprisingly, accuracy costs correlated between the two tasks, but only under accuracy pressure. The reaction time costs showed no relationship between the two tasks. As no correlations were found between the two response inhibition tasks in section 6, the evidence that manipulating speed and accuracy might help reveal relationship between different decision making tasks seems limited.

The choice of using the diffusion model for conflict tasks to examine model parameters in section 7 allowed application of the model in this chapter as well, even on a task without response inhibition element. As the diffusion model for conflict tasks utilizes the same parameters as the drift diffusion model, it can be applied to decision making tasks without inhibitory processes, by just omitting the impulsivity parameters. This allowed investigations into how SAT affects the model parameters from the two tasks, and whether the model parameters, rather than behavioural measures, relate in the two tasks. The results from section 7 were replicated again, by finding that the same model parameters were affected by the SAT manipulation: upper threshold (boundary separation), non-decision time, variability of non-decision time, and shape of starting point distribution. One notable exception was that in the RDK task, drift rate for controlled process was affected by SAT manipulations, agreeing with J. Zhang and Rowe (2014); Rae et al. (2014); Dambacher and Hübner (2015). However in the flanker task the drift rate was not affected by SAT manipulations, suggesting that maybe the effects of SAT could be different in response inhibition and perceptual decision making, as the finding was replicated across three studies that used the flanker task.

Between task correlations of model parameters, just like in section 7, were sparse, but visible across more parameters: on top of boundary separation (upper threshold) and non-decision time in section 7, variability of non-decision time and drift rate (for controlled process) seemed to correlate between the two task. Correlations between some of the model parameters were improved by the SAT manipulations, suggesting that more extreme behaviour makes similarities between task performance more visible in model parameters. This also suggests that looking into model parameters instead of behavioural measures for establishing relationships between response inhibition tasks could be beneficial, as they might reveal that the relationships come from similarities in processes common to all decision making, and not just impulsive decision making in itself.

9 Discussion

The focus of this thesis was investigation of whether deep learning could be a viable alternative to global optimization algorithms in the recovery of parameters from diffusion model for conflict tasks by Ulrich et al. (2015), which is an extension of the drift diffusion model (Ratcliff, 1978). The application of the diffusion model for conflict tasks to experimental studies is not straightforward because the model is non-differentiable; for this reason global optimization algorithms need to be utilized instead, without any guarantees that they would find a global minimum of the objective function. Even though global optimization algorithms are capable of recovering some parameters from the diffusion model very well (White et al., 2018), they are a huge drawback of the excessive computational time. As each dataset (reaction time distribution) is treated in isolation by the global optimization algorithms, parameter recovery from an individual dataset takes hours, sometimes days. When researchers have hundreds of datasets to apply the diffusion model for conflict tasks to, but no access to high performance computing clusters, the application of the model becomes unfeasible (Ambrosi et al., 2019). For this reason, alternative methods are needed that can recover the parameters as well as global optimization algorithms, but without the computational overhead. Deep learning methods could offer this alternative.

The main reason why global optimization algorithms are so computationally expensive is that they need to compute reaction time distributions at each step of the optimization. Even though the process of computing reaction time distributions is expensive, the global optimization algorithms do not store or reuse this information in any way, and the process has to be repeated for each dataset individually. The computation of reaction time distributions is not avoidable, as all methods investigating the goodness of fit of the proposed solution involve the comparison of the reaction time distributions of the problem dataset and the proposed solution dataset. How and what information from the reaction time distributions is used when comparing them is usually the focus of investigations of different fitting approaches (White et al., 2018), rather than different global optimization algorithms. Comparison of different global optimization algorithms would allow to find what approach of finding a solution to the diffusion model of conflict tasks is the most suited to the problem.

Instead of diving into the comparison of global optimization algorithms, this thesis first examined the solution space of the diffusion model for conflict tasks resulting from comparing reaction time distributions. When studies looking into parameter recovery from the model find that some parameters are hard to recover (White et al., 2018), conclusions cannot be made on whether the methods used in the study struggled to recover the parameters, or whether the parameters themselves are not recoverable, therefore any method would struggle. Investigation of the solution space allowed to answer this question specifically, by just looking at the surface of the seven-dimentional solution space (resulting from seven parameters from the diffusion model for conflict tasks). Section 2 showed that changes in five out of seven diffusion model for conflict tasks parameters can be mimicked by changes in other parameters, meaning that any global optimization algorithm will struggle to find a solution to the objective function of the diffusion model for conflict tasks, when reaction time distributions are compared in the fitting procedure.

The solution space of the diffusion model for conflict tasks contains many local minima. This results from some of the parameters of the diffusion model for conflict tasks having the same effects on the reaction time distribution as a combination of other parameters, therefore making the reverse inference very hard (given this reaction time distribution, what are the diffusion model for conflict tasks parameters that made it?). However different global optimization algorithms might be better suited to the problem due to the different approaches they employ to deal with the local minima. To examine how well different classes of methods perform, section 3 compared the recovery of parameters from the diffusion model for conflict tasks from five optimization methods. All five algorithms were constrained in their runtime, which was limited by the High Performance Computing cluster in Cardiff University. The results revealed that some algorithms, like the basin hopping algorithm, could not solve the problem in the given time, while the differential evolution algorithm outperformed all other algorithms in all experimental conditions. The superior performance of differential evolution is consistent with Hawkins et al. (2015), who found that differential evolution provided better parameter recovery from drift diffusion model than the particle optimization and the simplex algorithms. The result is also consistent with research outside of the decision making modelling field, as Civicioglu and Besdok (2013) found that differential evolution outperformed other global optimization algorithms, such as particle swarm optimization and artificial bee colony, in 50 benchmark numerical optimization functions. Even though section 2 suggested that limiting the number of parameters recovered from the diffusion model for conflict tasks could help with the recovery, this was not the case when investigating the performance of the global optimization algorithms. This finding has implications for applying global optimization algorithms as a method for parameter recovery from diffusion model for conflict tasks in experimental situations, in that there is no need to constrain parameter space as it does not improve parameter recovery.

Even though differential evolution recovered parameters very well when the resulting reaction time distributions were compared, the recovery of each individual parameter from the diffusion model for conflict tasks varied significantly. Only some parameters were recovered well, as indicated by high and significant correlations between recovered and actual parameter values, while other parameters were not recovered well, as the correlations were low and not significant. This is an issue with applying differential evolution (or any other global optimization algorithm as they all displayed similar performance of only recovering some parameters well) to recover parameters from behavioural studies, as no conclusions about a whole set of parameters, or some specific parameters of interest, could be made.

The biggest issue with all global optimization algorithms for parameter recovery is the excessive computational time it takes to apply them. Even when considering recovering parameters from distributions with a small number of trials, and limiting the runtime to 24 hours, applying global optimization algorithms to studies with even modest number of participants takes a very long time. For example, studies with 80 participants, and six different reaction time distributions per participant (assuming six different experimental manipulations), would take 640 days - that is almost two years of computational time. If researchers do not have access to high performance computing clusters, this makes application of non-differentiable decision making models to psychological research unfeasible.

For this reason, investigation into deep learning as an alternative to global optimization algorithms was started. Deep learning has become very popular in the recent years as the algorithm started beating records in many artificial intelligence problems (LeCun et al., 2015) and perform as well as humans on some problems that once seemed unfeasible for machines. The biggest limitation for applying deep learning is the vast number of training samples it requires, which can easily exceed millions of samples. The training data also needs to be labelled for supervised problems, which can be hard to achieve in some fields. For this reason, applying deep learning models in some fields, like human health research, has been very difficult (Miotto et al., 2017), because the sample sizes can be very limited. This however is one of the advantages of decision making models, as the data can be generated by the model very easily. Data generation is what happens in the recovery of parameters with global optimization algorithms, as the optimization algorithms compute reaction time distributions; but then they are discarded. This is a very wasteful approach, as if these reaction time distributions can be stored, and if we know their ground truth labels (the parameter values that were used to make them), they can be used as training data for deep learning models that try to predict parameter values from reaction time distributions.

Deep learning also avoids another pitfall of global optimization algorithms, as deep learning never actually compares reaction time distributions. Deep learning tries to minimize the error between predicted and actual parameter values by changing the weights of the deep learning model, which get multiplied by the input features (that do come from reaction time distributions), but the reaction time distributions are never compared themselves. This approach, coupled with the fact that thousands of reaction time distributions are being recovered at once (rather than individual reaction time distributions), might avoid the pitfalls of the local minima in the solution space of the diffusion model for conflict tasks. The results in section 4 and section 5 showed that deep learning was capable of recovering parameters from the diffusion model for conflict tasks exceptionally well, even from reaction time distributions with a small number of trials. The recovery of the trickier parameters, as identified by (White et al., 2018) and in section 3, was not great, but also not worse than the best performing global optimization algorithm, suggesting that deep learning can be a credible and, very importantly, a fast alternative to global optimization algorithms.

Some people would argue that deep learning is famed for the computational complexities, which would make the implementation of deep learning as computationally expensive as the global optimization algorithms, however this is not the case in practice. The biggest time cost in deep learning comes from creating the training data, but it only needs to be done once, and then the data can also be shared between researchers. Even though deep learning seems to have a reputation of being “hard to do”, in actual fact the implementation of deep neural networks is very straightforward if the basic principles of the algorithm are well understood, mainly due to the high level packages available in PYTHON, such as KERAS and PYTORCH. Both of the packages are very “pythonic” and make the implementation of even the very complicated deep neural network architectures quite straightforward - the tag line of the KERAS package is “Deep learning for humans”. The training time of the deep neural networks implemented in section 4 and section 5 was hours, compared to days for a recovering parameters from a single reaction time distribution using global optimization algorithms. Once deep learning models are trained, their application in parameter recovery takes seconds for hundreds of thousands of reaction time distributions AT ONCE, making them much faster overall than the global optimization algorithms.

Deep learning models are generally known to not be a good choice for problems where interpretability of the model is of high priority. As deep learning models contain hundreds of thousands if not millions of trainable weights, and can create very complicated non-linear features from the input data, this makes it really hard to then explain why certain decisions were made by the models. This is usually a high priority in fields like medicine or the legal field, where decisions not made by humans need to be interpreted and understood by humans. However this is not the case with decision making models. Even though it would be interesting to see what features are used for parameter prediction, and how they interact, this is not a legal or medical requirement and therefore can be omitted. Another major critical point about deep learning is whether it actually learns anything about the input data, or whether it just learns direct input-output matching without learning anything about the underlying qualities of the input data. This in general is a huge point of discussion, see (C. Zhang, Bengio, Hardt, Recht, & Vinyals, 2016). However it is unclear whether in parameter recovery, learning representations are actually needed, or whether input-output mapping is sufficient for the task of parameter recovery itself.

One issue that the researchers could face when trying to apply the deep learning models in parameter recovery could be the specificity of the inputs required for the models. For example, both section 4 and section 5 used congruent, incongruent, and neutral trials in model training. If the same pre-trained models were to be used in recovering parameters from experimental studies, all the experimental studies will need to have those three types of trials, as deep learning models cannot deal with missing inputs. In order for the deep learning models to be more general, they should be trained without the neutral trials in inputs, as they are not always included in the experimental studies.

The number of samples used to train the models in section 4 and section 5 should be reviewed as well. The number of 150,000 samples was chosen because models trained with data from distributions with a high number of trials seemed to perform exceptionally well with this number of samples and did not need any more training data. This is evident from deep learning models performing very well, even with quite shallow architectures.

But overall, 150,000 samples is a very small number for deep learning models, when the number of trainable weights is in hundreds of thousands. Increasing the training dataset at least ten times could benefit the performance of the models predicting parameters from reaction time distributions with low number of trials.

One of the most important issues to consider moving forward should be further shaping of the deep learning model architecture to predict parameters better from data with low number of trials, especially the impulsivity parameters, as they could not be predicted very well from data with low number of trials. In this thesis, data with low number of trials consisted of 600 trials, which is still quite high considering that if each decision takes 3 seconds to make, it would take half an hour to collect that data, which could be challenging with clinical or younger populations. We need to see how much the number of trials in the training distributions can be reduced to predict the parameters well from datasets with few hundred, or even below a hundred trials.

Even though learning from data with low number of trials is difficult due to the abundance of non-numerical inputs, further smoothing or filtering of the data is not an answer to this problem. This is because one of the the biggest advantages of the deep learning methods over shallow machine learning is that feature engineering is unnecessary for good model performance. Feature engineering can bring researcher bias to the models. In the ideal circumstances, if the models are provided with a good amount of training data and appropriate architecture that allows the models to learn the features, inputs of raw data should be sufficient for good model performance. This is quite tricky with reaction time data. As discussed in section 4, unlike in convolutional neural networks, reaction time distribution data is not spatially correlated. This means that if the trial sequence in a reaction time distributions were reshuffled, the distribution itself would not change. This makes neural networks with convolutions unsuitable. Secondly, even though we know there are temporal dependencies in reaction times in decision making, decision making models like the diffusion model for conflict tasks treat each trial as independent, therefore recurrent network architectures are also unsuitable. Reaction time distributions, in broad terms, are not bound by a fixed number of trials (e.g., a reaction time distribution with 1000 trials would overall be the same as a reaction time distribution with 995 trials or a reaction time distribution with 1005 trials). However the varied input size is a problem for deep learning, as most models require a fixed input size. This is usually solved with some kind of empty padding, for example, images can be framed with white pixels, word embeddings with non-words, and so on. It is yet unknown how to pad reaction time data so that it does not change the properties of the distributions. One way to avoid that would be using sequence models instead of fixed input sizes.

In the second part of the thesis, we looked at the stability of response inhibition tasks, and whether it was improved when speed and accuracy pressures were manipulated in the experimental studies, as well as whether looking at parameters from diffusion model for conflict tasks improved the stability between tasks and between the experimental sessions. The only reason we were able to investigate model parameters was because using deep learning allowed us to apply the diffusion model for conflict tasks in a very short period of time. If global optimization algorithms were used instead, recovering parameters from 1100 different reaction time distributions (the amount of different sets of parameters that were recovered in section 7 and section 8) would have taken that long - 1100 days. This approach is also very inflexible and allows no space for errors with either the data preparation (e.g., changing your mind about data cleaning criteria), or the algorithm hyperparameters (e.g. changing mutation and crossover parameter values in the differential evolution algorithm), meaning that refitting the data for any reason is computationally unfeasible. This is completely avoidable with deep learning as even though retraining the deep learning models can take a few days, depending on the size of the architecture and the training dataset, the application pipelines take seconds, meaning that refitting full sets of experimental data from hundreds of participants can be done in seconds.

To conclude, deep learning is a very promising approach in parameter recovery from the diffusion model for conflict tasks, as it performs as well as differential evolution, which came out as the best global optimization algorithm for the problem in our benchmarking in section 3, but in a fraction of time.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., … Zheng, X. (2015). *TensorFlow: Large-scale machine learning on heterogeneous systems.* Retrieved from `http://tensorflow.org/` (Software available from tensorflow.org)

Agam, Y., Joseph, R. M., Barton, J. J., & Manoach, D. S. (2010). Reduced cognitive control of response inhibition by the anterior cingulate cortex in autism spectrum disorders. *Neuroimage*, *52*(1), 336–347.

Aichert, D. S., Wöstmann, N. M., Costa, A., Macare, C., Wenig, J. R., Möller, H.-J., … Ettinger, U. (2012). Associations between trait impulsivity and prepotent response inhibition. *Journal of clinical and experimental neuropsychology*, *34*(10), 1016–1032.

Alessi, S., & Petry, N. (2003). Pathological gambling severity is associated with impulsivity in a delay discounting procedure. *Behavioural Processes*, *64*(3), 345–354.

Ambrosi, S., Servant, M., Blaye, A., & Burle, B. (2019). Conflict processing in kindergarten children: New evidence from distribution analyses reveals the dynamics of incorrect response activation and suppression. *Journal of experimental child psychology*, *177*, 36–52.

Anokhin, A. P., Golosheykin, S., Grant, J. D., & Heath, A. C. (2017). Heritability of brain activity related to response inhibition: A longitudinal genetic study in adolescent twins. *International Journal of Psychophysiology*, *115*, 112–124.

Aron, A. R., & Poldrack, R. A. (2005). The cognitive neuroscience of response inhibition: Relevance for genetic research in attention-deficit/hyperactivity disorder. *Biological psychiatry*, *57*(11), 1285–1292.

Baca-Garcia, E., Diaz-Sastre, C., Basurte, E., Prieto, R., Ceverino, A., & de Leon, J. (2001). A prospective study of the paradoxical relationship between impulsivity and lethality of suicide attempts. *The Journal of clinical psychiatry*, *62*(7), 1–478.

Beck, R. C., & Triplett, M. F. (2009). Test–retest reliability of a group-administered paper–pencil measure of delay discounting. *Experimental and clinical psychopharmacology*, *17*(5), 345.

Behnel, S., Bradshaw, R., Citro, C., Dalcin, L., Seljebotn, D., & Smith, K. (2011). Cython: The best of both worlds. *Computing in Science Engineering*, *13*(2), 31 -39. doi: 10.1109/MCSE.2010.118

Bellman, R. (1961). *Adaptive control process: A guided tour.*

Bengio, Y., et al. (2009). Learning deep architectures for AI. *Foundations and trends® in Machine Learning*, *2*(1), 1–127.

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, *13*, 281–305.

Bergstra, J., Yamins, D., & Cox, D. (2013). Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *International conference on machine learning* (pp. 115–123).

Bickel, W. K., Odum, A. L., & Madden, G. J. (1999). Impulsivity and cigarette smoking: Delay discounting in current, never, and ex-smokers. *Psychopharmacology*, *146*(4), 447–454.

Black, D. W., Shaw, M., McCormick, B., Bayless, J. D., & Allen, J. (2012). Neuropsychological performance, impulsivity, ADHD symptoms, and novelty seeking in compulsive buying disorder. *Psychiatry Research*, *200*(2), 581–587.

Booth, J. R., Burman, D. D., Meyer, J. R., Lei, Z., Trommer, B. L., Davenport, N. D., . . . Marsel Mesulam, M. (2005). Larger deficits in brain networks for response inhibition than for visual selective attention in attention deficit hyperactivity disorder (ADHD). *Journal of Child Psychology and Psychiatry*, *46*(1), 94–111.

Bottou, L. (2012). Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade* (pp. 421–436). Springer.

Bousardt, A. M., Hoogendoorn, A. W., Noorthoorn, E. O., Hummelen, J. W., & Nijman, H. L. (2016). Predicting inpatient aggression by self-reported impulsivity in forensic psychiatric patients. *Criminal behaviour and mental health*, *26*(3), 161–173.

Britten, K. H., Shadlen, M. N., Newsome, W. T., & Movshon, J. A. (1992). The analysis of visual motion: A comparison of neuronal and psychophysical performance. *Journal of Neuroscience*, *12*(12), 4745–4765.

Carlisi, C., Norman, L., Murphy, C., Christakou, A., Chantiluke, K., Giampietro, V., . . . others (2017). Comparison of neural substrates of temporal discounting between youth with autism spectrum disorder and with obsessive-compulsive disorder. *Psychological medicine*, *47*(14), 2513–2527.

Cheung, A. M., Mitsis, E. M., & Halperin, J. M. (2004). The relationship of behavioral inhibition to executive functions in young adults. *Journal of Clinical and Experimental Neuropsychology*, *26*(3), 393–404.

Chollet, F., et al. (2015). *Keras.* `https://github.com/fchollet/keras`. GitHub.

Civicioglu, P., & Besdok, E. (2013). *A conceptual comparison of the cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms* *artif intell rev (2013) 39: 315-346, doi 10.1007/s10462-011-9276-0.* DOI.

Costa, P. T., & MacCrae, R. R. (1992). *Revised NEO personality inventory (NEO PI-R) and NEO five-factor inventory (NEO FFI): Professional manual.* Psychological Assessment Resources.

Črepinšek, M., Liu, S.-H., & Mernik, M. (2013). Exploration and exploitation in evolutionary algorithms: A survey. *ACM Computing Surveys (CSUR)*, *45*(3), 35.

Cruz, A. C., Luvisi, A., De Bellis, L., & Ampatzidis, Y. (2017). Vision-based plant disease detection system using transfer and deep learning. In *2017 asabe annual international meeting* (p. 1).

Cummins, T., Hawi, Z., Hocking, J., Strudwick, M., Hester, R., Garavan, H., ... Bellgrove, M. (2012). Dopamine transporter genotype predicts behavioural and neural measures of response inhibition. *Molecular Psychiatry*, *17*(11), 1086.

Cyders, M. A., & Coskunpinar, A. (2011). Measurement of constructs using self-report and behavioral lab tasks: Is there overlap in nomothetic span and construct representation for impulsivity? *Clinical psychology review*, *31*(6), 965–982.

Dalley, J. W., Everitt, B. J., & Robbins, T. W. (2011). Impulsivity, compulsivity, and top-down cognitive control. *Neuron*, *69*(4), 680–694.

Dambacher, M., & Hübner, R. (2015). Time pressure affects the efficiency of perceptual processing in decisions under conflict. *Psychological research*, *79*(1), 83–94.

de Bruin, A., & Della Sala, S. (2017). Effects of age on inhibitory control are affected by task-specific features. *The Quarterly Journal of Experimental Psychology*(just-accepted), 1–40.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

de Wit, H., Flory, J. D., Acheson, A., McCloskey, M., & Manuck, S. B. (2007). IQ and nonplanning impulsivity are independently associated with delay discounting in middle-aged adults. *Personality and Individual Differences*, *42*(1), 111–121.

Dick, D. M., Smith, G., Olausson, P., Mitchell, S. H., Leeman, R. F., O'malley, S. S., & Sher, K. (2010). Understanding the construct of impulsivity and its relationship to alcohol use disorders. *Addiction biology*, *15*(2), 217–226.

Dillon, D. G., Wiecki, T., Pechtel, P., Webb, C., Goer, F., Murray, L., ... others (2015). A computational analysis of flanker interference in depression. *Psychological medicine*, *45*(11), 2333–2344.

Durana, J., Barnes, P., Johnson, J., & Shure, M. (1993). A neurodevelopmental view of impulsivity and its relationship to the superfactors of personality. *The impulsive* *client*, 23–37.

Encyclopedia of mathematics, Kolmogorov–Smirnov test. (n.d.). Retrieved from `https://www.encyclopediaofmath.org/index.php/Kolmogorov\OT1\textendashSmirnov_test`

Enticott, P. G., Ogloff, J. R., & Bradshaw, J. L. (2006). Associations between laboratory measures of executive inhibitory control and self-reported impulsivity. *Personality and Individual Differences*, *41*(2), 285–294.

Eriksen, B. A., & Eriksen, C. W. (1974). Effects of noise letters upon the identification of a target letter in a nonsearch task. *Perception & psychophysics*, *16*(1), 143–149.

Eysenck, H., & Eysenck, M. (1985). *Personality and individual differences: A natural science perspective.* Plenum, New York, NY, USA.

Fears, S. C., Schür, R., Sjouwerman, R., Service, S. K., Araya, C., Araya, X., … others (2015). Brain structure–function associations in multi-generational families genetically enriched for bipolar disorder. *Brain*, *138*(7), 2087–2102.

Fogel, L. J., Owens, A. J., & Walsh, M. J. (1966). *Artificial intelligence through simulated evolution.* New York, USA: John Wiley.

Forstmann, B., Ratcliff, R., & Wagenmakers, E.-J. (2016). Sequential sampling models in cognitive neuroscience: Advantages, applications, and extensions. *Psychology*, *67*(1), 641.

Forstmann, B. U., Anwander, A., Schäfer, A., Neumann, J., Brown, S., Wagenmakers, E.-J., … Turner, R. (2010). Cortico-striatal connections predict control over speed and accuracy in perceptual decision making. *Proceedings of the National Academy of Sciences*, *107*(36), 15916–15920.

Fortgang, R., Hultman, C., van Erp, T., & Cannon, T. (2016). Multidimensional assessment of impulsivity in schizophrenia, bipolar disorder, and major depressive disorder: Testing for shared endophenotypes. *Psychological medicine*, *46*(7), 1497–1507.

Friedman, N. P., & Miyake, A. (2004). The relations among inhibition and interference control functions: A latent-variable analysis. *Journal of experimental psychology: General*, *133*(1), 101.

Gebru, T., Krause, J., Wang, Y., Chen, D., Deng, J., Aiden, E. L., & Fei-Fei, L. (2017). Using deep learning and Google street view to estimate the demographic makeup of the US. *arXiv preprint arXiv:1702.06683*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., … Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems* (pp. 2672–2680).

Grassi, G., Pallanti, S., Righi, L., Figee, M., Mantione, M., Denys, D., … Stratta, P. (2015). Think twice: Impulsivity and decision making in obsessive–compulsive disorder. *Journal of Behavioral Addictions*, *4*(4), 263–272.

Gudjonsson, G. H. (1989). Compliance in an interrogative situation: A new scale. *Personality and Individual Differences*, *10*(5), 535–540.

Gudjonsson, G. H., Sigurdsson, J. F., Einarsson, E., & Einarsson, J. H. (2008). Personal versus impersonal relationship compliance and their relationship with personality. *The Journal of Forensic Psychiatry & Psychology*, *19*(4), 502–516.

Gvion, Y., & Apter, A. (2017). Impulsivity, decision-making and their role in suicidal behaviour. In *Handbook of suicidal behaviour* (pp. 91–101). Springer.

Hajek, T., Alda, M., Hajek, E., & Ivanoff, J. (2013). Functional neuroanatomy of response inhibition in bipolar disorders–combined voxel based and cognitive performance meta-analysis. *Journal of psychiatric research*, *47*(12), 1955–1966.

Hawkins, G. E., Forstmann, B. U., Wagenmakers, E.-J., Ratcliff, R., & Brown, S. D. (2015). Revisiting the evidence for collapsing boundaries and urgency signals in perceptual decision-making. *Journal of Neuroscience*, *35*(6), 2476–2484.

Hedge, C., Powell, G., Bompas, A., Vivian-Griffiths, S., & Sumner, P. (2018). Low and variable correlation between reaction time costs and accuracy costs explained by accumulation models: Meta-analysis and simulations. *Psychological bulletin*, *144*(11), 1200.

Hedge, C., Powell, G., & Sumner, P. (2018). The reliability paradox: Why robust cognitive tasks do not produce reliable individual differences. *Behavior Research Methods*, *50*(3), 1166–1186.

Heitz, R. P. (2014). The speed-accuracy tradeoff: history, physiology, methodology, and behavior. *Frontiers in neuroscience*, *8*, 150.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Horn, N., Dolan, M., Elliott, R., Deakin, J., & Woodruff, P. (2003). Response inhibition and impulsivity: An fMRI study. *Neuropsychologia*, *41*(14), 1959–1966.

Huang, F. J., Boureau, Y.-L., LeCun, Y., et al. (2007). Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *Computer vision and pattern recognition, 2007. cvpr'07. ieee conference on* (pp. 1–8).

Iaboni, F., Douglas, V. I., & Baker, A. (1995). Effects of reward and response costs on inhibition in ADHD children. *Journal of Abnormal Psychology*, *104*(1), 232.

Jones, E., Oliphant, T., Peterson, P., et al. (2001–). *SciPy: Open source scientific tools for Python.* Retrieved from `http://www.scipy.org/` ([Online; accessed

2016-08-16])

Kana, R. K., Keller, T. A., Minshew, N. J., & Just, M. A. (2007). Inhibitory control in high-functioning autism: Decreased activation and underconnectivity in inhibition networks. *Biological psychiatry*, *62*(3), 198–206.

Karalunas, S. L., & Huang-Pollock, C. L. (2013). Integrating impairments in reaction time and executive function using a diffusion model framework. *Journal of abnormal child psychology*, *41*(5), 837–850.

Kennedy, J. (2011). Particle swarm optimization. In *Encyclopedia of machine learning* (pp. 760–766). Springer.

Kennedy, J. F., Kennedy, J., Eberhart, R. C., & Shi, Y. (2001). *Swarm intelligence.* Morgan Kaufmann.

Kertzman, S., Vainder, M., Aizer, A., Kotler, M., & Dannon, P. N. (2017). Pathological gambling and impulsivity: Comparison of the different measures in the behavior inhibition tasks. *Personality and Individual Differences*, *107*, 212–218.

Kim, M., Lee, T. H., Choi, J.-S., Kwak, Y. B., Hwang, W. J., Kim, T., ... others (2017). Neurophysiological correlates of altered response inhibition in internet gaming disorder and obsessive-compulsive disorder: Perspectives from impulsivity and compulsivity. *Scientific Reports*, *7*, 41742.

Krajbich, I., & Rangel, A. (2011). Multialternative drift-diffusion model predicts the relationship between visual fixations and choice in value-based decisions. *Proceedings of the National Academy of Sciences*, *108*(33), 13852–13857.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097–1105).

Lecrubier, Y., Braconnier, A., Said, S., & Payan, C. (1995). The impulsivity rating scale (IRS): Preliminary results. *European Psychiatry*, *10*(7), 331–338.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444.

Leotti, L. A., & Wager, T. D. (2010). Motivational influences on response inhibition measures. *Journal of Experimental Psychology: Human Perception and Performance*, *36*(2), 430.

Lerche, V., & Voss, A. (2017). Retest reliability of the parameters of the ratcliff diffusion model. *Psychological research*, *81*(3), 629–652.

Lewandowsky, S., & Farrell, S. (2010). *Computational modeling in cognition: Principles and practice.* Sage.

Li, H. (2017). Deep learning for natural language processing: Advantages and challenges.

National Science Review.

Logan, G. D. (1994). On the ability to inhibit thought and action: A users' guide to the stop signal paradigm. In T. H. Carr (Ed.), *Inhibitory processes in attention, memory, and language* (p. 189-239). San Diego, CA, US: Academic Press.

Lu, X., Tsao, Y., Matsuda, S., & Hori, C. (2013). Speech enhancement based on deep denoising autoencoder. In *Interspeech* (pp. 436–440).

Lynam, D., Smith, G., Whiteside, S., & Cyders, M. (2006). The UPPS-P: Assessing five personality pathways to impulsive behavior. *West Lafayette, IN: Purdue University*.

Marsh, R., Steinglass, J. E., Gerber, A. J., O'Leary, K. G., Wang, Z., Murphy, D., . . . Peterson, B. S. (2009). Deficient activity in the neural systems that mediate self-regulatory control in bulimia nervosa. *Archives of general psychiatry*, *66*(1), 51–63.

Masci, J., Meier, U., Cireşan, D., & Schmidhuber, J. (2011). Stacked convolutional auto-encoders for hierarchical feature extraction. In *International conference on artificial neural networks* (pp. 52–59).

Matzke, D., & Wagenmakers, E.-J. (2009). Psychological interpretation of the ex-gaussian and shifted wald parameters: A diffusion model analysis. *Psychonomic bulletin & review*, *16*(5), 798–817.

Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., & Khudanpur, S. (2010). Recurrent neural network based language model. In *Interspeech* (Vol. 2, p. 3).

Miletić, S., Turner, B. M., Forstmann, B. U., & van Maanen, L. (2017). Parameter recovery for the leaky competing accumulator model. *Journal of Mathematical Psychology*, *76*, 25–50.

Min, S., Lee, B., & Yoon, S. (2017). Deep learning in bioinformatics. *Briefings in bioinformatics*, *18*(5), 851–869.

Miotto, R., Wang, F., Wang, S., Jiang, X., & Dudley, J. T. (2017). Deep learning for healthcare: Review, opportunities and challenges. *Briefings in bioinformatics*.

Močkus, J. (1975). On bayesian methods for seeking the extremum. In *Optimization techniques ifip technical conference* (pp. 400–404).

Mulder, M. J., Bos, D., Weusten, J. M., van Belle, J., van Dijk, S. C., Simen, P., . . . Durston, S. (2010). Basic impairments in regulating the speed-accuracy tradeoff predict symptoms of attention-deficit/hyperactivity disorder. *Biological psychiatry*, *68*(12), 1114–1119.

Myung, I. J. (2003). Tutorial on maximum likelihood estimation. *Journal of mathematical Psychology*, *47*(1), 90–100.

Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., & Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. *Journal of Big Data*, *2*(1), 1.

Newcorn, J. H., Halperin, J. M., Jensen, P. S., Abikoff, H. B., Arnold, L. E., Cantwell, D. P., ... others (2001). Symptom profiles in children with ADHD: Effects of comorbidity and gender. *Journal of the American Academy of Child & Adolescent Psychiatry*, *40*(2), 137–146.

Newman, A. L., & Meyer, T. D. (2014). Impulsivity: Present during euthymia in bipolar disorder? A systematic review. *International journal of bipolar disorders*, *2*(1), 2.

Oades, R. D., Lasky-Su, J., Christiansen, H., Faraone, S. V., Sonuga-Barke, E. J., Banaschewski, T., ... others (2008). The influence of serotonin and other genes on impulsive behavioral aggression and cognitive impulsivity in children with attention-deficit/hyperactivity disorder (ADHD): Findings from a family-based association test (FBAT) analysis. *Behavioral and Brain Functions*, *4*(1), 48.

Oquab, M., Bottou, L., Laptev, I., & Sivic, J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 1717–1724).

Patton, J. H., Stanford, M. S., & Barratt, E. S. (1995). Factor structure of the barratt impulsiveness scale. *Journal of clinical psychology*, *51*(6), 768–774.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Peirce, J. (2009). Generating stimuli for neuroscience using psychopy. *Frontiers in Neuroinformatics*, *2*, 10. Retrieved from `http://journal.frontiersin.org/article/10.3389/neuro.11.010.2008` doi: 10.3389/neuro.11.010.2008

Petry, N. M. (2001). Substance abuse, pathological gambling, and impulsiveness. *Drug and alcohol dependence*, *63*(1), 29–38.

Pote, I., Torkamani, M., Kefalopoulou, Z.-M., Zrinzo, L., Limousin-Dowsey, P., Foltynie, T., ... Jahanshahi, M. (2016). Subthalamic nucleus deep brain stimulation induces impulsive action when patients with Parkinson's disease act under speed pressure. *Experimental brain research*, 1–12.

Rae, B., Heathcote, A., Donkin, C., Averell, L., & Brown, S. (2014). The hare and the tortoise: Emphasizing speed can change the evidence used to make decisions. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *40*(5), 1226.

Ratcliff, R. (1978). A theory of memory retrieval. *Psychological review*, *85*(2), 59.

Ratcliff, R., Smith, P. L., Brown, S. D., & McKoon, G. (2016). Diffusion decision model: Current issues and history. *Trends in cognitive sciences*, *20*(4), 260–281.

Ratcliff, R., Thapar, A., & McKoon, G. (2010). Individual differences, aging, and IQ in two-choice tasks. *Cognitive psychology*, *60*(3), 127–157.

Reynolds, B., Ortengren, A., Richards, J. B., & de Wit, H. (2006). Dimensions of impulsive behavior: Personality and behavioral measures. *Personality and individual differences*, *40*(2), 305–315.

Richards, C., Moss, J., Nelson, L., & Oliver, C. (2016). Persistence of self-injurious behaviour in autism spectrum disorder over 3 years: A prospective cohort study of risk markers. *Journal of neurodevelopmental disorders*, *8*(1), 21.

Robbins, T. W., Gillan, C. M., Smith, D. G., de Wit, S., & Ersche, K. D. (2012). Neurocognitive endophenotypes of impulsivity and compulsivity: Towards dimensional psychiatry. *Trends in cognitive sciences*, *16*(1), 81–91.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

Servant, M., van Wouwe, N., Wylie, S. A., & Logan, G. D. (2018). A model-based quantification of action control deficits in Parkinson's disease. *Neuropsychologia*, *111*, 26–35.

Servant, M., White, C., Montagnini, A., & Burle, B. (2016). Linking theoretical decision-making mechanisms in the Simon task with electrophysiological data: A model-based neuroscience study in humans. *Journal of cognitive neuroscience*.

Sharma, L., Markon, K. E., & Clark, L. A. (2014). Toward a theory of distinct types of "impulsive" behaviors: A meta-analysis of self-report and behavioral measures. *Psychological bulletin*, *140*(2), 374.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., ... others (2017). Mastering the game of Go without human knowledge. *Nature*, *550*(7676), 354.

Simon, J. R., & Rudell, A. P. (1967). Auditory SR compatibility: The effect of an irrelevant cue on information processing. *Journal of applied psychology*, *51*(3), 300.

Smits, R. R., Stein, J. S., Johnson, P. S., Odum, A. L., & Madden, G. J. (2013). Test–retest reliability and construct validity of the experiential discounting task. *Experimental and clinical psychopharmacology*, *21*(2), 155.

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of machine learning research*, *15*(1), 1929–1958.

Stanford, M. S., Mathias, C. W., Dougherty, D. M., Lake, S. L., Anderson, N. E., & Patton, J. H. (2009). Fifty years of the Barratt Impulsiveness Scale: An update and review. *Personality and individual differences*, *47*(5), 385–395.

Steel, Z., & Blaszczynski, A. (1998). Impulsivity, personality disorders and pathological gambling severity. *Addiction*, *93*(6), 895–905.

Storn, R., & Price, K. (1997). Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, *11*(4), 341–359.

Stroop, J. R. (1935). Studies of interference in serial verbal reactions. *Journal of experimental psychology*, *18*(6), 643.

Suarez, I., Burle, B., Tobon, C., Pineda, D., Lopera, F., Hasbroucq, T., & Casini, L. (2015). Deciphering interference control in adults with ADHD by using distribution analyses and electromyographic activity. *Acta psychologica*, *159*, 85–92.

Thompson, C. A., Ratcliff, R., & McKoon, G. (2016). Individual differences in the components of children's and adults' information processing for simple symbolic and non-symbolic numeric decisions. *Journal of Experimental Child Psychology*, *150*, 48–71.

Ulrich, R., Schröter, H., Leuthold, H., & Birngruber, T. (2015). Automatic and controlled stimulus processing in conflict tasks: Superimposed diffusion processes and delta functions. *Cognitive psychology*, *78*, 148–174.

Uziela, K., Menéndez Hurtado, D., Shu, N., Wallner, B., & Elofsson, A. (2017). ProQ3D: Improved model quality assessments using deep learning. *Bioinformatics*, *33*(10), 1578–1580.

Van Wouwe, N., van den Wildenberg, W., Claassen, D., Kanoff, K., Bashore, T., & Wylie, S. (2014). Speed pressure in conflict situations impedes inhibitory action control in Parkinson's disease. *Biological psychology*, *101*, 44–60.

Vieira, S., Pinaya, W. H., & Mechelli, A. (2017). Using deep learning to investigate the neuroimaging correlates of psychiatric and neurological disorders: Methods and applications. *Neuroscience & Biobehavioral Reviews*, *74*, 58–75.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., & Manzagol, P.-A. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, *11*(Dec), 3371–3408.

Voss, A., Rothermund, K., & Voss, J. (2004). Interpreting the parameters of the diffusion model: An empirical validation. *Memory & Cognition*, *32*(7), 1206–1220.

Wagenmakers, E.-J., Van Der Maas, H. L., & Grasman, R. P. (2007). An EZ-diffusion model for response time and accuracy. *Psychonomic bulletin & review*, *14*(1), 3–22.

Weafer, J., Baggott, M. J., & de Wit, H. (2013). Test–retest reliability of behavioral measures of impulsive choice, impulsive action, and inattention. *Experimental and Clinical Psychopharmacology*, *21*(6), 475.

Weintraub, D., David, A. S., Evans, A. H., Grant, J. E., & Stacy, M. (2015). Clinical spectrum of impulse control disorders in Parkinson's disease. *Movement Disorders*, *30*(2), 121–127.

White, C., Ratcliff, R., Vasey, M., & McKoon, G. (2009). Dysphoria and memory for emotional material: A diffusion-model analysis. *Cognition and Emotion*, *23*(1), 181–205.

White, C. N., Ratcliff, R., & Starns, J. J. (2011). Diffusion models of the flanker task: Discrete versus gradual attentional selection. *Cognitive psychology*, *63*(4), 210–238.

White, C. N., Servant, M., & Logan, G. D. (2018). Testing the validity of conflict drift-diffusion models for use in estimating cognitive processes: A parameter-recovery study. *Psychonomic bulletin & review*, *25*(1), 286–301.

White, T. L., Lejuez, C. W., & de Wit, H. (2008). Test-retest characteristics of the balloon analogue risk task (BART). *Experimental and clinical psychopharmacology*, *16*(6), 565.

Wiecki, T. V., Sofer, I., & Frank, M. J. (2013). HDDM: Hierarchical bayesian estimation of the drift-diffusion model in python. *Frontiers in neuroinformatics*, *7*, 14.

Winkel, J., Van Maanen, L., Ratcliff, R., Van der Schaaf, M. E., Van Schouwenburg, M. R., Cools, R., & Forstmann, B. U. (2012). Bromocriptine does not alter speed–accuracy tradeoff. *Frontiers in neuroscience*, *6*, 126.

Winstanley, C. A., Eagle, D. M., & Robbins, T. W. (2006). Behavioral models of impulsivity in relation to ADHD: Translation between clinical and preclinical studies. *Clinical psychology review*, *26*(4), 379–395.

Wodka, E. L., Mark Mahone, E., Blankner, J. G., Gidley Larson, J. C., Fotedar, S., Denckla, M. B., & Mostofsky, S. H. (2007). Evidence that response inhibition is a primary deficit in ADHD. *Journal of clinical and experimental neuropsychology*, *29*(4), 345–356.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, *1*(1), 67–82.

Wöstmann, N. M., Aichert, D. S., Costa, A., Rubia, K., Möller, H.-J., & Ettinger, U. (2013). Reliability and plasticity of response inhibition and interference control.

Brain and cognition, *81*(1), 82–94.

Wu, M., Hartmann, M., Skunde, M., Herzog, W., & Friederich, H.-C. (2013). Inhibitory control in bulimic-type eating disorders: A systematic review and meta-analysis. *PloS one*, *8*(12), e83412.

Wylie, S., Van Den Wildenberg, W., Ridderinkhof, K., Bashore, T., Powell, V., Manning, C., & Wooten, G. (2009). The effect of speed-accuracy strategy on response interference control in Parkinson's disease. *Neuropsychologia*, *47*(8), 1844–1853.

Xie, J., Xu, L., & Chen, E. (2012). Image denoising and inpainting with deep neural networks. In *Advances in neural information processing systems* (pp. 341–349).

Yarkoni, T., & Westfall, J. (2016). Choosing prediction over explanation in psychology: Lessons from machine learning. *Unpublished manuscript. Retrieved from http://jakewestfall. org/publications/Yarkoni_Westfall_choosing_prediction. pdf*.

Zhan, L., Piwowar, B., Liu, W.-K., Hsu, P., Lai, S., & Chen, J. Z. (2004). Multicanonical basin hopping: A new global optimization method for complex systems. *The Journal of chemical physics*, *120*(12), 5536–5542.

Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2016). Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*.

Zhang, J., Rittman, T., Nombela, C., Fois, A., Coyle-Gilchrist, I., Barker, R. A., ... Rowe, J. B. (2015). Different decision deficits impair response inhibition in progressive supranuclear palsy and Parkinson's disease. *Brain*, *139*(1), 161–173.

Zhang, J., & Rowe, J. B. (2014). Dissociable mechanisms of speed-accuracy tradeoff during visual perceptual learning are revealed by a hierarchical drift-diffusion model. *Frontiers in Neuroscience*, *8*, 69.