北京大学

本科生毕业论文

题目： Model-Free and Model-Based Algorithms in Human Sequential Decision Making

人类序列决策过程中的模型无关和模型依赖的算法

姓　　名：陈乐天

学　　号：1400013706

院　　系：心理与认知科学学院

专　　业：心理学

指导教师：张航

二〇一八 年 五 月

北京大学本科毕业论文导师评阅表

<table>
<tr><td>学　　号</td><td>1400013706</td><td>学生姓名</td><td>陈乐天</td><td>论文成绩</td><td></td></tr>
<tr><td>学　　院
（系）</td><td colspan="3">心理与认知科学学院</td><td>专 业</td><td>心理学</td></tr>
<tr><td>导师姓名</td><td>张航</td><td>导师单位</td><td>心理与认知科学学院，麦戈文脑研究所，生命科学联合中心</td><td>职称</td><td>研究员</td></tr>
<tr><td colspan="2">论文题目</td><td colspan="4">Model-Free vs Model-Based Algorithms in Human Sequential Decision Making</td></tr>
<tr><td colspan="2">导师评语
（包含对论文的性质、难度、分量、综合训练等是否符合培养目标的目的等评价）</td><td colspan="4">在毕业论文中，陈乐天同学设计完成了一个序列决策实验任务，用计算建模方法结合强化学习理论探讨了人类进行序列决策时的学习过程。这个项目具有相当的挑战性，需要对计算机科学中的强化学习理论及其在心理学和神经科学中的应用拥有全面的理解，对计算建模方法能够灵活运用。陈乐天同学独立完成了文献综述、实验设计、数据分析和论文写作的全过程，论文逻辑清晰，写作规范，达到了本科毕业的标准。

导师签名：</td></tr>
</table>

（此表格供院系参考用，各院系可根据实际情况制定和使用原有本单位的论文评阅表）

版权声明

PEKING UNIVERSITY

BACHELOR THESIS

Model-Free vs Model-Based Algorithms in Human Sequential Decision Making

Author:
Letian CHEN

Supervisor:
Dr. Hang ZHANG

A thesis submitted in fulfillment of the requirements for the degree of Bachelor of Science

in the

Computation & Decision Lab
School of Psychological and Cognitive Sciences

May 20, 2018

摘要

近年来，有关强化学习的研究快速发展，研究者提出多种强化学习模型来解释人类的学习行为，其中最经典的两类分别是“模型依赖”与“模型无关”。目前已有二者均存在的证据，但二者如何合作/对抗却始终无法得到解答。本文设计了一个较前人而言任务复杂度大幅提升的实验环境，采取了多任务同时训练，多任务区块训练两种实验条件，观察记录并分析被试在此环境中的学习过程。结果表明，被试在复杂环境中依然进行了良好的学习，而现有的模型无法完全解释被试数据。因此本文进一步提出了带遗忘因子的 Q 价值学习，相比于之前没有遗忘因子的状态-动作-奖励-状态-动作学习能够更好地拟合该实验数据。此外，本文还提出了模型依赖帮助模型无关，模型无关帮助模型依赖等新模型，实现了混合模型等，并分析了其与最优的带遗忘因子的 Q 价值学习间的关系。

关键词

模型依赖 模型无关 强化学习 混合模型 阶段决策

PEKING UNIVERSITY

Abstract

School of Psychological and Cognitive Sciences

Bachelor of Science

Model-Free vs Model-Based Algorithms in Human Sequential Decision Making

by Letian CHEN

Recently, there has been large advancement in researches about reinforcement learning. Researchers introduce several reinforcement learning models to explain human's learning behavior. The most classic two of the algorithms are "Model-Based" and "Model-Free". It has been proved that both of them co-exist in human brain, but till now there is no answer regarding how two learning systems assist or compete. We design a relatively complex experiment environment in this thesis. In order to figure out human's multi-tasking learning process, we also design two types of learning: randomized learning or block learning. Results show that despite the difficulty of new environment, human participants learn very well in all performance metrics. However, existing model could not fully explain participants' data. Therefore, we introduce "Q-learning with forget rate" model, which is a much better fit of participants' data than classic "SARSA learning without forget rate" model. Besides, we also introduce new "Model-Based Help Model-Free" model and "Model-Free Help Model-Based" model and implement a hybrid model. The results also shows that hybrid and MF model are two best models fitting participants' data in this task.

Keyword: Model-Based, Model-Free, Reinforcement Learning, Hybrid Model, Sequential Decision Making

Contents

List of Figures

List of Tables

Chapter 1

Chapter 1: Introduction

1.1 Introduction to Sequential Decision Making

We are making all kinds of decisions in our daily life: whether bringing an umbrella, which canteen shall we go for today's lunch, doing homework or housework, etc. Some daily decisions are sequential, which means that the consequence of an action may reveal its effect in the future, and several sequential actions are required to gain the reward. For example, if we are driving a car on a highway, suddenly we spot a car accident ahead. At this time, we must hit the break and then rotate steering wheel to avoid another car accident. Only hitting the break will result in crashing, though with a lower speed. Only rotating steering wheel on a high speed situation will make us lose control of the car, most likely making another disaster. Only when we hit the break to slow down the vehicle and then change car's direction can we solve the puzzle. This is a typical sequential decision making problem. Every action we take may cause a consequence, but most importantly, it will influence the consequence of the following actions.

Another representative example is chess. When it is your turn, you can take your time to think about the situation currently on the board. After you have made the decision, for example moving a rook, you could take your action. This action may have a direct consequence, like eating your opponent's queen. But it also has subsequent consequences, such as your rook being eaten by your opponent's knight. It could also have a far-away consequence, for instance, you lose the game.

1.2 Introduction to Markov Decision Process

Markov decision process (MDP) provides a formalized mathematical framework for modeling sequential decision making with some restricts. To introduce MDP, we first need to define several terms. At each time step in a MDP, the process is in some state s, and the decision maker may choose any action a that is available in state s (Bellman, 1957). The process responds at the next time step by randomly moving into a new state s', and gives the decision maker a corresponding reward $R(s')$. The process is shown in Fig. 1.1.

Taken the chess example again, each time the player receives a state s, in this case, the chess board situation. The player could choose an action that is viable under current state s. Assume the player chooses action a, the environment, in this case, the chess rule, will transit the state s into a new state s', also giving the player a reward r. The reward may be small when you eat a pawn while it may be big when you win the game.

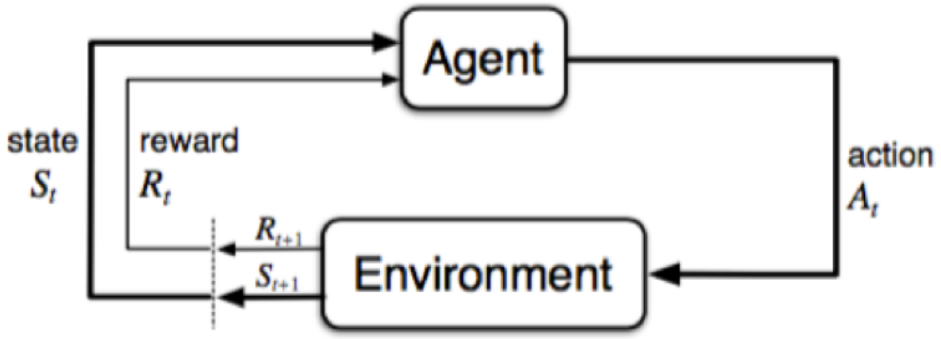


FIGURE 1.1: Markov Decision Process (Sutton and Barto, 1998)

One core property of Markov decision process is that the transition obeys Markov property. Markov property means the probability that the process moves into its new state s' is influenced only by the chosen action a and current state s. Specifically, it is given by the state transition function $T(s, a, s')$. Given s and a, it is conditionally independent of all previous states and actions.

Markov property makes it possible to analyze the action selection by only looking at the current state because current state contains all information the decision maker need to know, otherwise the Markov property is unsatisfied. It is obvious that the chess example has Markov property since the information on board, s, is sufficient for the player to make its action and transit to a new state (ignore the influence to board caused by opponents).

This thesis will only focus on Markov decision process because of the simplicity of Markov property.

1.3 Thesis Structure

Sequential decision making are so essential that our life may fall apart without such ability. Hence, there has been much work inspecting the process of sequential decision making, especially Markov decision process. In this thesis, I also try to design an experiment to explore several possible decision methods under Markov devision process.

We have given introduction of the sequential decision making problem and its formalized framework Markov Decision Process in Chapter 1. We will introduce several existing work in this topic in Chapter 2. After that, we will present our experiment in Chapter 3. Chapter 4 will be basic statistical analyses and in Chapter 5 we will put forward several models that are theoretically possible in our task and compare them with classic ones, both in model fitting and model simulation. Finally, we will discuss and conclude what we have discovered with some future research directions.

Chapter 2

Chapter 2: Background Theory

2.1 Reinforcement Learning

From Pavlov's classical conditioning, Skinner's operant conditioning and Thorndike's Law of Effect, people are trying to connect the consequence to the behavior (Thorndike, 1927). Reinforcement learning (RL) is initially a psychology term to explain the phenomenon that people will increase the probability of doing something after positive stimulation such as reward and will decrease the probability after negative stimulation such as electric shock (similar to operant conditioning) (Dayan and Balleine, 2002). In the 1980s, reinforcement learning is introduced to machine learning area, concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward.

It is widely accepted that reinforcement learning is the proper solution to Markov decision process. Most recently, reinforcement learning in computer science has been experiencing a boost due to its great success on several tasks such as Go and Atari games (Mnih et al., 2015).

At the same time, reinforcement learning in psychology and neuroscience have made great advancement. Hollerman and Schultz (1998) discovered that dopamine neurons in midbrain have the corresponding activity with reward prediction error, which is the core element of Model-Free learning. This pattern is also frequently reported in later work (Bayer and Glimcher, 2005; Waelti, Dickinson, and Schultz, 2001). Human fMRI results also prove that stratum does have stronger activity when reward prediction error is positive (Garrison, Erdeniz, and Done, 2013; Kishida et al., 2016).

Therefore, reinforcement learning has been proven effective both empirically and theoretically. What's more, it is viable in human brain via midbrain dopamine system. Taken all pieces together, reinforcement learning seems to become the most convincing theory in modeling sequential decision making. Thus, we will introduce several reinforcement learning algorithms in this section. We will also build our models on top of classic reinforcement learning algorithms in Chapter 5.

2.2 Model-Free Algorithm Family

Model-Free means the participant or agent does not build a model for the environment explicitly but to maintain a value estimation for each state-action pair.

The Model-Free algorithm family mainly includes SARSA and Q-learning. To understand Model-Free algorithm, first we need to define "Q value" as in Equation 2.1. The Q value represents how much value can I get in the long run after taking action a in state s. Note that the value considers long run outcome rather than immediate outcome. According to this definition, if we could get the true Q value, we should just always choose the action with maximum Q value to gain the best performance. So, how do we calculate Q value?

$$Q(s,a) = \sum_{t=0}^{\infty} \gamma^t R(s_t) \tag{2.1}$$

There are two classic methods to estimate Q value in the model-free manner: SARSA and Q-learning (Daw and Dayan, 2014). They are both motivated by reward prediction error (RPE). Reward prediction error is defined as the difference between the expected value and the real encountered value. If a participant receives a positive RPE, he will know that his prediction is underestimated and should be increased. If a participant receives a negative RPE, he will inversely decrease his value estimation. SARSA and Q-learning formulation are shown in Equation 2.2 and 2.3.

The difference between SARSA and Q-learning is that when calculating reward prediction error (RPE), SARSA uses the true action chosen by participants in the next timestep, while Q-learning uses the action with maximum Q value on the next state.

$$\begin{aligned} \delta_{RPE} &= r(s') + \gamma Q_{SARSA}(s',a') - Q_{SARSA}(s,a) \\ Q_{SARSA}(s,a) &= Q_{SARSA}(s,a) + \alpha\delta_{RPE} \end{aligned} \tag{2.2}$$

$$\begin{aligned} \delta_{RPE} &= r(s') + \gamma \max_{a' \in a(s)} Q_Q(s',a') - Q_Q(s,a) \\ Q_Q(s,a) &= Q_Q(s,a) + \alpha\delta_{RPE} \end{aligned} \tag{2.3}$$

Model-Free algorithms requires little computation but needs fair amount of storage because it need to store all the state-action pair values. However, Model-Free algorithms suffer from environment change because they could only re-learn the Q value when either transition T or reward R has been changed.

2.3 Model-Based Algorithm Family

Unlike Model-Free algorithms, Model-Based algorithm maintain an explicit representation of the environment, i.e. the transition matrix T and reward function R. The learning method of T and R will be explained in Chapter 5. Here we will first explain several methods to calculate value estimation using estimated T and R.

The first intuitive method is to use the definition of value function to calculate v by regarding the estimated T and R as true T and R.

$$Q(s,a) = \hat{r} + \gamma\hat{T}\hat{R} + \gamma^2\hat{T}^2\hat{R} + ... \tag{2.4}$$

The second way is to use bellman equation (Sutton and Barto, 1998) $Q = R + \gamma TQ$. Specifically, we could build an iteration algorithm to do value iteration by Equation 2.5, in which Q_t represents t-th iteration Q.

$$Q_{t+1} = \hat{R} + \gamma \hat{T} Q_t \tag{2.5}$$

In order to decrease the computation complexity, we could also do Monte-Carlo sampling for transition matrix T.

Model-based method requires a lot of computation but are able to adjust to environment changes. For instance, when reward function changes, as long as the participant has learned the transition structure of the environment, he will still be able to obtain the reward.

2.4 Other Algorithms

There has been much work about Model-Free and Model-Based combination method. Daw, Niv, and Dayan (2005) and Doya et al. (2002) suggests that Model-Free and Model-Based value estimation may be combined in proportion to their certainty about their value estimation. Gläscher et al. (2010) introduces a hybrid model whose hybrid weight is a exponential function. Daw et al. (2011) claims that Model-Based and Model-Free work simultaneously in our decision making.

Except MF model, MB model and their combination, their still exists many other computational models. Gershman and Daw (2017) argues that reinforcement learning is tightly bound to episodic memory and use a kernel function to make new predictions. Botvinick et al. (2015) and Botvinick (2012) bring forward a hierarchical model of reinforcement learning. The main idea of hierarchical RL is to combine some action sequence as a new action for participants to choose.

Chapter 3

Chapter 3: Experiment

3.1 Experiment Intuition

Existing experiments are generally simple both in state space and temporal structure, such as Gläscher et al. (2010), Daw et al. (2011). They only examine the RL process in a two-stage decision making task. There is no evidence which RL algorithm will work when the problem becomes complicated.

Besides, it seems that less is known about multi-task decision making. Previous work mostly use one reward function to see the learning process. But we do not know whether multiple-task simultaneous learning will cause tasks to assist or interfere with each other.

Finally, the orders of tasks may further change the assistance or interference.

With these questions, we defined our own experiment environment.

3.2 Experiment Environment

3.2.1 Environment Structure

We create an abstract "labyrinth" environment for human to play. The labyrinth structure is displayed in Fig. 3.1. Each circle and the corresponding character represents a "state" participants may be in. The lines between states represents action's primary resulting states (we will explain the word "primary" in Section 3.4. Right now, you could just regard where the line points is its resulting state). It is obvious that the labyrinth consists of 6 states and each state has 3 actions, which point to 3 different resulting states. The goal for participants is to move from one state to another. For example, the participants may be asked to migrate from state A to state B. In this situation, the optimal solution is firstly move from A to E, and then go straight to B. As a result, he successfully transmit himself from A to B in two steps. It is worth noting that participants know nothing about the structure of this environment before acting. Therefore, they could only know the structure by trying and learning.

In this task, each states is represented by a fractal image, as shown in Fig. 3.2. We chose 6 dissimilar fractal images to represent 6 different states.

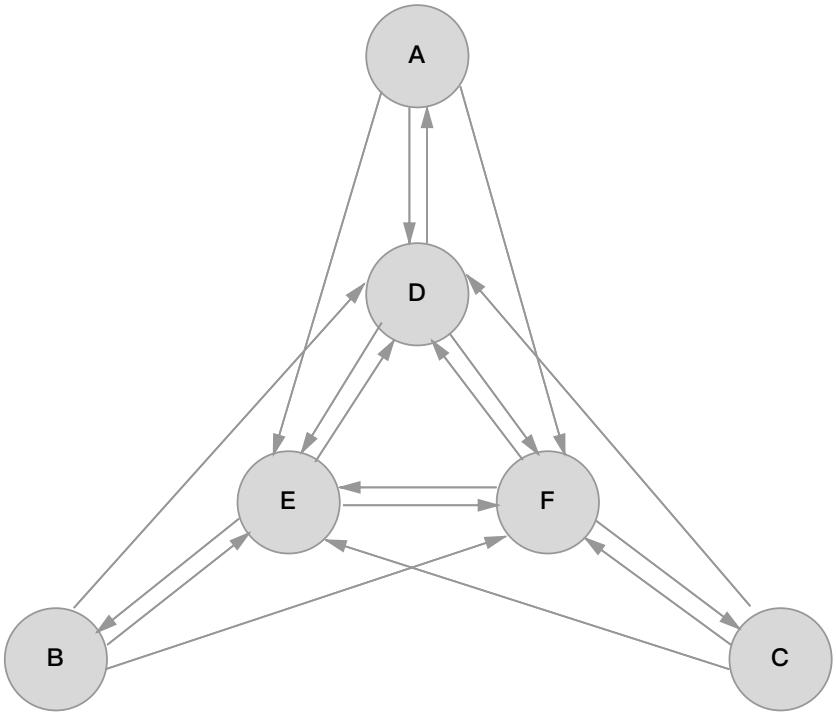


FIGURE 3.1: Environment Structure: circle indicates states (including *A*, *B*, *C*, *D*, *E*, *F*), line indicates action's primary consequence.

3.2.2 Trial Procedure

The Trial Procedure is demonstrated in Fig. 3.3.

In the beginning of each trial, a fixation is displayed at the screen center for 1 second. After that, the trial start state and goal state are shown (start state at center, end state on the right) for 1.5 seconds. This is called preparation phase. Then, the free choice phase begins.

At each timestep of free-choice phase, the state which the participant is now in is displayed in the center, being surrounded by three arrows indicating three kinds of actions. The goal state is always displayed on the right as in the preparation phase. Besides, there is a reward indication below the state, showing how much tokens participant could get if he reaches the end state after the next action. After the participant has made his decision and pressed one of three buttons, the center image will transit to the outcome state image with a fade-in fade-out fashion (previous image's α decreased gradually and new image's α increased gradually). In the meantime, reward indication will decrease by 1 (initially 20). But if the reward has been 1, it will not decrease but remain 1. There is no maximum reaction time, which means the screen will remain unchanged unless the participant press a button. When the transition ends, a new timestep begins. This is repeated until the state that participant is in reaches the end state.

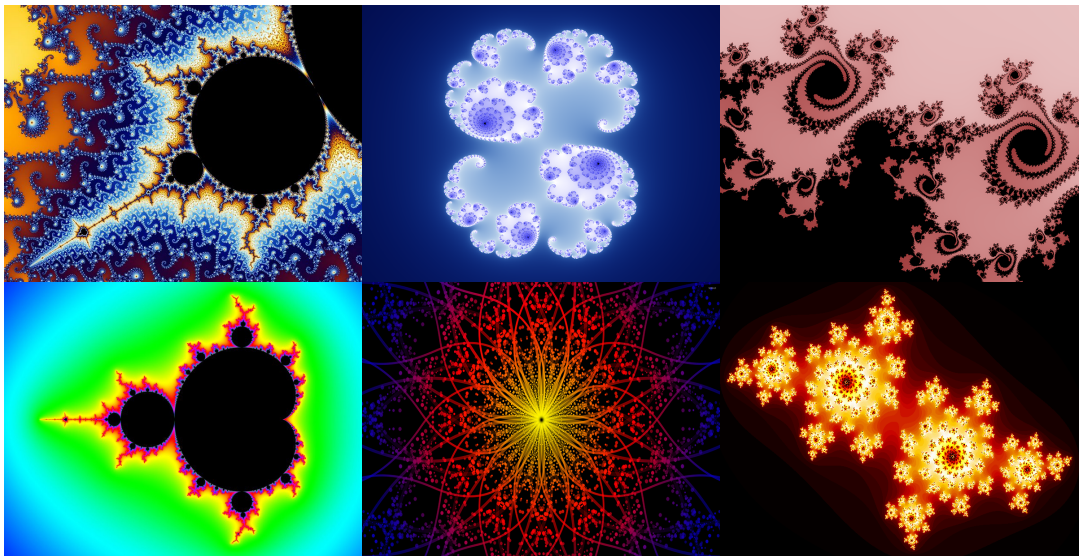


FIGURE 3.2: States Fractal Image

If participants have reached the end state, the trial enters feedback phase, in which total steps the participant takes to move to the end state and tokens earned are shown for 3 seconds. After feedback phase, a new trial begins with fixation.

3.3 Experiment Design

For every participant, there are totally 144 trials. each trial follows the procedure described in Section 3.2.2. Till now, we haven't described how the start state and end state are chosen for each trial, and that's the key point of experiment design.

The experiment is a between-subject design. The between-subject independent variable is the order of tasks and have two levels: randomized and block. Participants with odd participant id are assigned to block condition while participants with even participant id are assigned to randomized condition. For all conditions, whether randomized or block, the experiment consists of three kinds of start-end pairs: A to B, B to C and C to A. In short, the "inner" state D, E, F will not be used as start or end state. The reason why we choose such tasks is that every task has an optimal solution of two steps. Just as the example shown in 3.2.1, if the participant is asked to transit from A to B, he should take the path of A -> E -> B, which takes two steps. You may also be wondering why we do not choose the opposite direction tasks, such as B to A as well. It is because we want to examine the possibility of knowledge transfer from previous tasks to newly occurred goals. For example, after learning how to move from A to B and from B to C, the participants are asked to do the task of C to A. In this manner, we want to test whether participants could transfer any knowledge from previous tasks, for instance, the structure of of environment, to the new tasks. That is exactly the difference between model-free and model-based algorithms. The overall trial number for each participant is 144. We divide 144 trials averagely into three tasks (A to B, B to C and C to A), making each one 44 trials.

The experiment has 4 blocks, each of which contains 36 trials. For randomized condition, participants' tasks are assigned pseudo-randomly. For block condition,

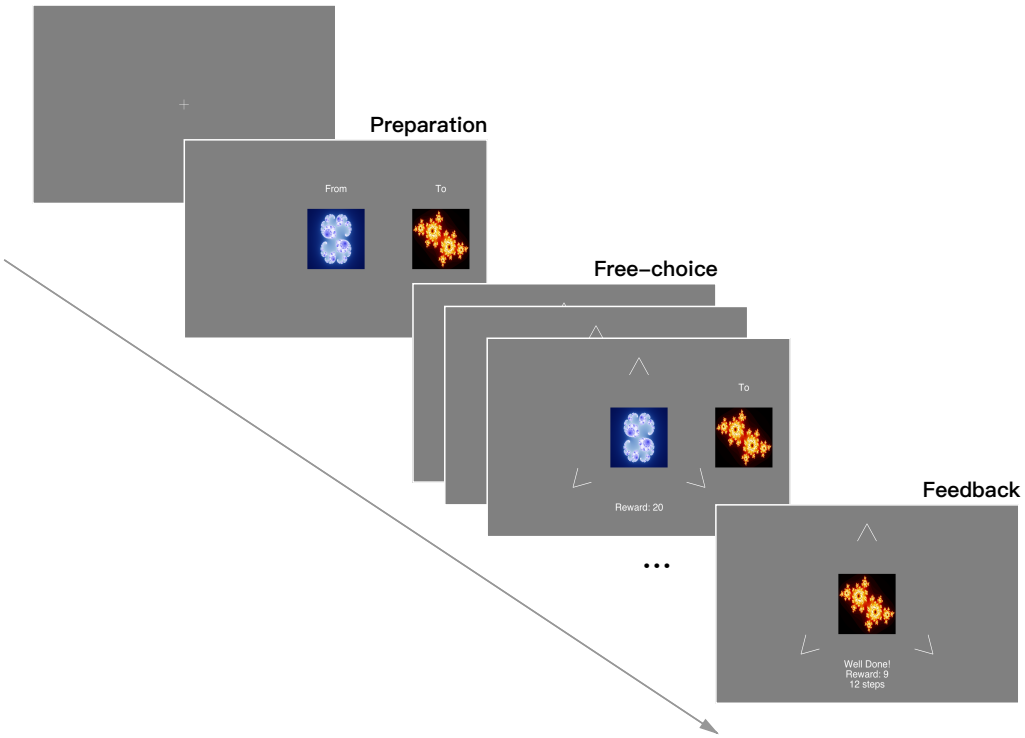


FIGURE 3.3: Experiment Procedure

participants will only learn A to B in the first block, and then learn B to C in the second block, learn C to A in the third block. The final block of block condition consists of 12 trials for each task and the order is randomly assigned. In short, under randomized condition participants will learn all three tasks simultaneously while under block condition participants will learn the task one by one and finally test in the last block.

3.4 Experiment Details

Now it is clear how the trial tasks are assigned and how a single trial proceeds. We still have several experiment details to explain.

For the mapping from abstract state characters (such as A) to fractal images, each participants received a randomly assigned mapping. In this case, the effect that specific images may cause can be balanced.

For the mapping from state's action to "primary" consequence, it is also randomly assigned for each participant to avoid the possibility that specific optimal solution is preferred and relatively easy to learn for participants.

The action's "primary" consequence mentioned in Section 3.2.1 means that in fact, action's consequence is stochastic in this environment. For a specific action of a state, it will have 0.7 probability to transit to the primary consequence. However, it still have 0.2 and 0.1 probability to transit to other action's primary consequence of the

same state. For example, the state A's first action's primary consequence is E, while second action's primary consequence is D and third action's primary consequence is F. In this case, when participants chose the first action in state A, it will transit to E with probability 0.7 and will transit to D or F with probability 0.2 or 0.1 (the one with 0.2 probability is called "secondary" consequence, which is determined together with "primary" consequence when the environment is created). Thus, the action's consequence is stochastic actually. The reason why we add stochasticity here is to increase difficulty of the task, making it hard for participants to find the structure of our environment. As we mentioned in Section 1, the existing work is mainly focused on laboratory toy problems, rather than real-world complex decision making. Though the complexity is still not comparable to real-world situations, it has been much more complex than previous tasks.

The data our experiment records is the whole decision process, including every timestep's data of every trial. Each timestep data consists of current state, action chosen by participants, transition resulting state and reaction time. We will show how we analyze data in the following chapters.

As for the reward, all the rewards participants collected in the experiment are tokens. The participant fee is calculated by $fee = 30 + tokens/100$. An average participant may gain a fee of approximately 50 CNY.

The data is collected on 36 Peking University students. They all have normal vision or corrected normal vision, including color vision. All participants had given informed consent before the experiment.

Chapter 4

Chapter 4: Statistical Analysis

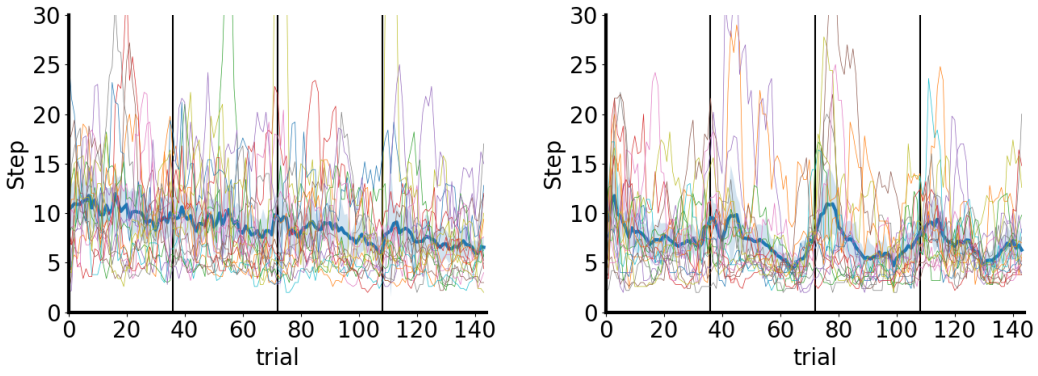
In this chapter, we will firstly calculate several metrics according to the participants' data and show the descriptive statistics. After that we will use statistical test to prove that learning does occur in both randomized and block conditions.

4.1 Metrics Definition

As we mentioned in Section 3.4, we record each timestep's data of each trial, including current state, action chosen by participants, transition resulting state and reaction time. It is hard to directly analyze sequential data as in this experiment, thus we calculate several metrics for each trial, and analyze them on the trial dimension.

4.1.1 Steps

Number of steps is the most straightforward metric to measure a trial. It is obvious that the less steps participants takes, the better his policy is. This metric is named "step". Participants' step data is shown in Fig. 4.1.

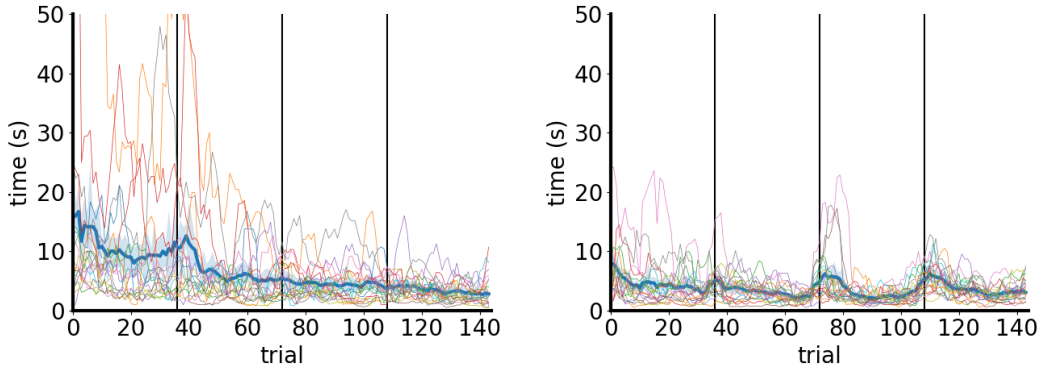


(a) Randomized Condition (b) Block Condition

FIGURE 4.1: Step data of participants in (a) randomized condition (b) block condition. The thick blue line is the mean of all participants (blue shadow indicating standard error), while the thin colorful lines are data of each participant. Vertical lines on 36, 72, 108 indicate block separation.cData line is filtered by Savitzky-Golay method.

4.1.2 Reaction Time and Normalized Reaction Time

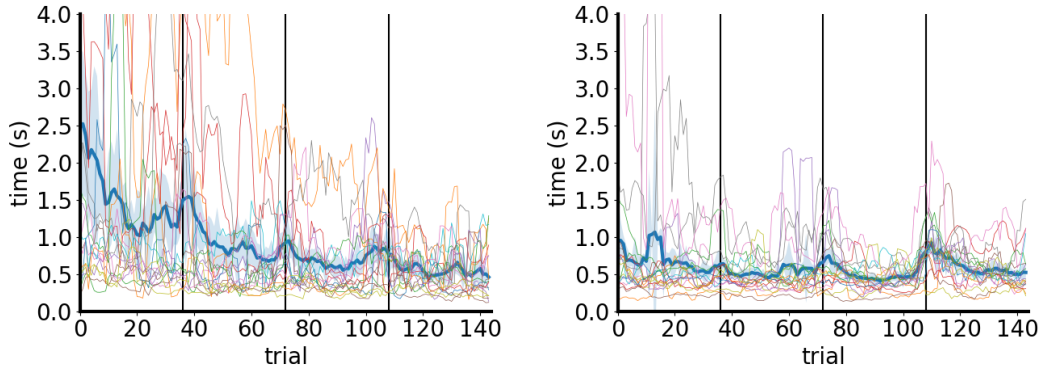
The reaction time of each timestep is recorded in data. Hence, we could sum all the reaction time of all the timesteps in one trial to represent participants' overall decision making time in a single trial. This metric is named "Reaction Time". Participant's reaction time data is shown in Fig. 4.2.



(a) Randomized Condition (b) Block Condition

FIGURE 4.2: Reaction time data of participants in (a) randomized condition (b) block condition. The thick blue line is the mean of all participants (blue shadow indicating standard error), while the thin colorful lines are data of each participant. Vertical lines on 36, 72, 108 indicate block separation. Data line is filtered by Savitzky-Golay method.

It is obvious that the "Reaction Time" metrics is strongly influenced by "Step" because of the sum operation. Thus, we create a new metric named "Normalized Reaction Time" to decrease the influence of "Step" by dividing "Reaction Time" with "Steps". Fig 4.3 shows "Normalized Reaction Time" in participants.



(a) Randomized Condition (b) Block Condition

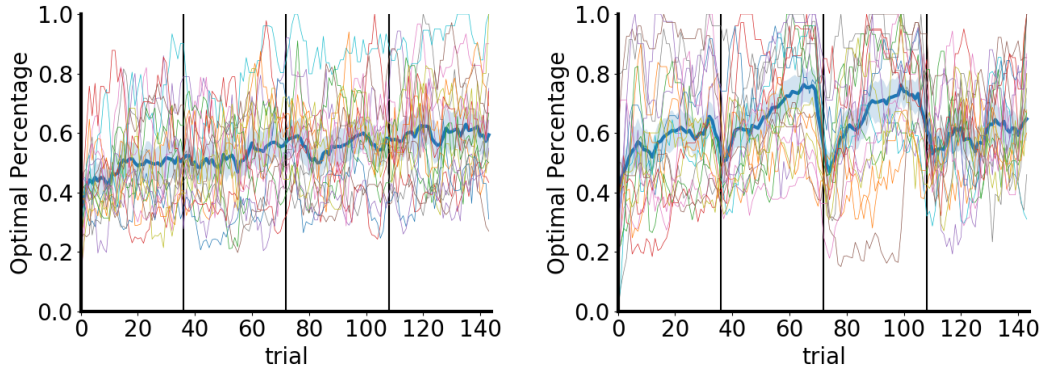
FIGURE 4.3: Normalized Reaction time data of participants in (a) randomized condition (b) block condition. The thick blue line is the mean of all participants (blue shadow indicating standard error), while the thin colorful lines are data of each participant. Vertical lines on 36, 72, 108 indicate block separation. Data line is filtered by Savitzky-Golay method.

4.1.3 Optimal Percentage

All three metrics mentioned above are all straightforward from raw data. But we are most interested in how participants learn to play this task well. For this purpose, we defined several Optimal-related metrics to indicate how participants learn this task step by step.

First we need to define "optimal". At each state the participant is in, there exists an optimal action choose. For instance, if the goal is B, the optimal action in A, D, F, C will all be the one whose primary consequence is E, while the optimal action in E is the one whose primary consequence is B. This action is called "optimal action".

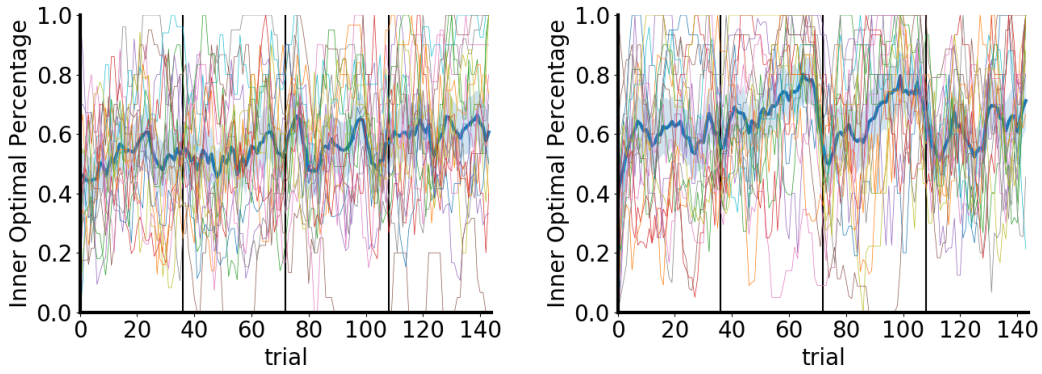
Thus, we could calculate whether at each timestep the participant choose the optimal action. Then we are able to calculate the optimal action percentage in each trial, making it a trial metric. This metric is called "Optimal Percentage" and its trend with trial is shown in Fig. 4.4.



(a) Randomized Condition (b) Block Condition

FIGURE 4.4: Optimal Percentage data of participants in (a) randomized condition (b) block condition. The thick blue line is the mean of all participants (blue shadow indicating standard error), while the thin colorful lines are data of each participant. Vertical lines on 36, 72, 108 indicate block separation. Data line is filtered by Savitzky-Golay method.

In order to examine participants' learning more precisely, we divide the state into three categories: "inner", "outer", "last". For each trial with a end state X, the "last" category includes only the one that can directly move to end state X. For instance, if the end state is B, then the "last" category includes only E (the environment structure can be found in Fig. 3.1). The "inner" category includes two of three inner states D, E, F except the one included in the "last" category, which in this example means the "inner" category includes D and F. Finally the "outer" category includes two of three outer states A, B, C except the end state, which in this instance means A and C. Optimal Percentage in three state categories are named as "Inner Optimal Percentage", "Outer Optimal Percentage" and "Last Optimal Percentage". Participants' data on these three metrics are shown in Fig. 4.5, Fig. 4.6 and Fig. 4.7.



(a) Randomized Condition (b) Block Condition

FIGURE 4.5: Inner Optimal Percentage data of participants in (a) randomized condition (b) block condition. The thick blue line is the mean of all participants (blue shadow indicating standard error), while the thin colorful lines are data of each participant. Vertical lines on 36, 72, 108 indicate block separation. Data line is filtered by Savitzky-Golay method.

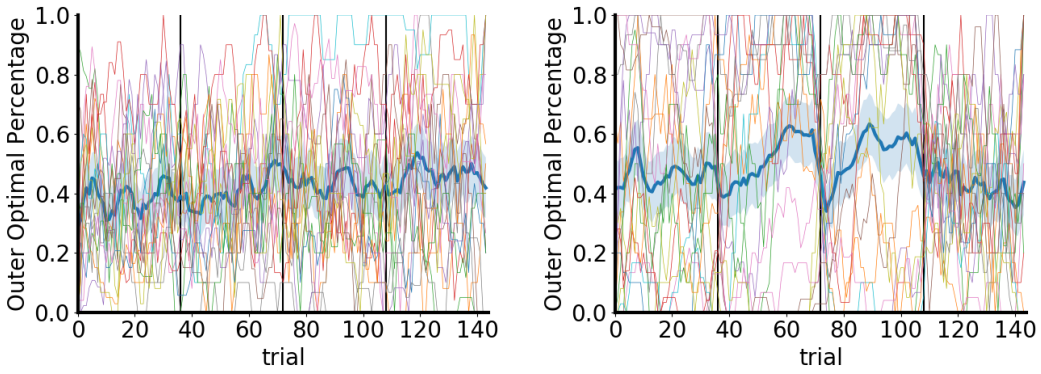
TABLE 4.1: All Metrics in Data Analysis

Metric Name	**Bound**	**Simulation Reflected**
Step	[2, Inf)	√
Reaction Time	(0, Inf)	×
Normalized Reaction Time	(0, Inf)	×
Optimal Percentage	[0, 1]	√
Inner Optimal Percentage	[0, 1]	√
Outer Optimal Percentage	[0, 1]	√
Last Optimal Percentage	[0, 1]	√

The advantage of "Optimal Percentage" over "Step" is that it will not be influenced by the stochasticity of the transition (action stochastic consequence), while "Step" is strongly influenced by the transition randomness even if all the actions chosen by participants are optimal. What's more, "Optimal Percentage" is a variable between 0 and 1 while "Step" does not have an upper bound, making "Optimal Percentage" metric having better statistical properties.

The reason why we further divide "Optimal Percentage" into three categories is that we assume participants may take different learning process for each different states. For example, it is obvious that "last" category is the most simple to learn. In the meantime, it seems that participants learn "inner" category much better than "outer" category. In this way, we may divide participants' learning in three different kinds of states.

All 7 metrics are summarized in Table 4.1, and the last column indicates whether or not it can be reflected in model simulation in Section 5.3.



(a) Randomized Condition (b) Block Condition

FIGURE 4.6: Outer Optimal Percentage data of participants in (a) randomized condition (b) block condition. The thick blue line is the mean of all participants (blue shadow indicating standard error), while the thin colorful lines are data of each participant. Vertical lines on 36, 72, 108 indicate block separation. Data line is filtered by Savitzky-Golay method.

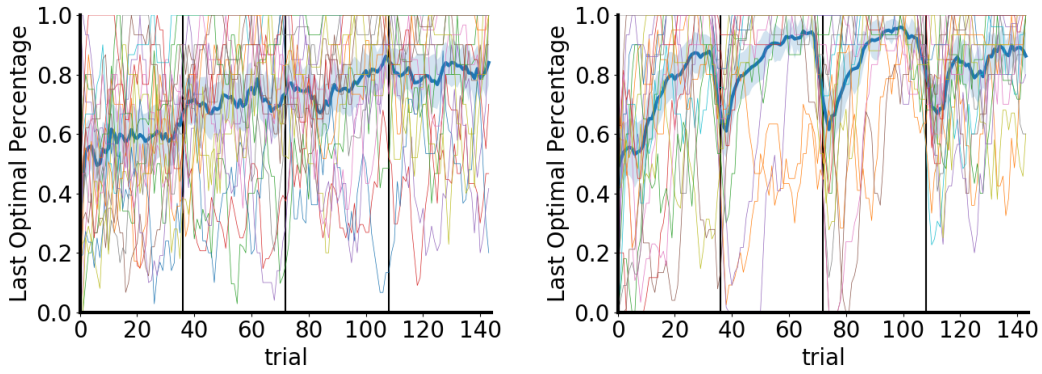
4.2 Metrics Under Randomized Condition

In this section, we will analyze the learning of participants in randomized condition. We built a linear model for each metric with one independent variable: trial number. We assume that participants learn not only within-blocks but also between-blocks. The linear model $metric = \beta_0 + \beta_1 \times trial$ was fit by generalized least square method. The regression results are summarized in Table 4.2.

It is clear from the result that participants do learn something about the task and their performance metrics, such as "Step", all four "Optimal Percentage", are getting better as trial number getting bigger. We could also see the trends in Fig. 4.1, 4.4, 4.5, 4.6, 4.7. In the meantime, "Reaction Time" and "Normalized Reaction Time" decreases as trial number increases, indicating that participants having more confidence and making their decision quicker. It is worth noting that both "Reaction Time" and "Normalized Reaction Time" could not be reflected in the analysis of Chapter 5 in either model simulation or model fit. Therefore, it remains some space to explore in the future about these two metrics, especially when modeling uncertainty.

4.3 Metrics Under Block Condition

In this section, we will analyze the learning of participants in block condition. Unlike randomized condition, linear model for block condition has two independent variables: timestep and block. Note that timestep means the trial number within a block, while block represents block number. It is because in the block condition, participants will do the same task for each one of first three blocks. We assume that participants learn not only within-blocks but also between-blocks. Therefore, the linear model becomes $metric = \beta_0 + \beta_1 \times timestep + \beta_2 \times block$. It was fit by generalized least square method. The regression results are summarized in Table 4.3.



(a) Randomized Condition (b) Block Condition

FIGURE 4.7: Last Optimal Percentage data of participants in (a) randomized condition (b) block condition. The thick blue line is the mean of all participants (blue shadow indicating standard error), while the thin colorful lines are data of each participant. Vertical lines on 36, 72, 108 indicate block separation. Data line is filtered by Savitzky-Golay method.

TABLE 4.2: Linear Model for Metrics on Trial Number in Randomized condition

Metric	β_1	**Significance of** β_1	β_0
Step	-0.0299	<.001	10.7862
Reaction Time	-0.0726	<.001	11.6328
Normalized Reaction Time	-0.0090	<.001	1.5710
Optimal Percentage	0.0011	<.001	0.4603
Inner Optimal Percentage	0.0008	<.001	0.4941
Outer Optimal Percentage	0.0007	<.001	0.3692
Last Optimal Percentage	0.0020	<.001	0.5662

From the results we could tell that within-block performance is just like the randomized condition. When the timestep gets bigger, the performance gets better. We could more clearly see the trends than randomized condition in Fig. 4.1, 4.4, 4.5, 4.6, 4.7. Just as randomized condition, "Reaction Time" and "Normalized Reaction Time" decreases as timestep increases, indicating that participants having more confidence and making their decision quicker within the block. But most importantly, there seems to show little evidence for learning between blocks. Actually, there only are "Step" ($p < .05$) and "Last Optimal Percentage" ($p < .001$) that are statistical significant. It is most likely that "Step"'s decrease is because of the increase of "Last Optimal Percentage". The data seems to convey the message that in block condition, less is learnt between-block, indicating that each task does not assist or interfere with other tasks. We also did the same linear model of block condition for randomized condition data for comparison, and it is summarized in Table 4.4. It shows that in randomized condition, between-block learning is significant on all the performance metrics. But combining the previous analysis in 4.2, we would conclude that the between-block learning is just the side-product of trial learning in randomized condition.

TABLE 4.3: Linear Model for Metrics on Timestep and Block in Block condition

Metric	β_1	p **of** β_1	β_2	p **of** β_2	β_0
Step	-0.1231	<.001	-0.2915	.034	9.8876
Reaction Time	-0.0917	<.001	-0.2017	.011	5.6542
Normalized Reaction Time	-0.0068	<.001	-0.0289	.071	0.7583
Optimal Percentage	0.0057	<.001	0.0042	.369	0.5137
Inner Optimal Percentage	0.0049	<.001	0.0009	.896	0.5477
Outer Optimal Percentage	0.0034	.002	-0.0106	.172	0.4395
Last Optimal Percentage	0.0092	<.001	0.0284	<.001	0.6065

TABLE 4.4: Linear Model for Metrics on Timestep and Block in Randomized condition

Metric	β_1	p **of** β_1	β_2	p **of** β_2	β_0
Step	-0.0593	.001	-1.0054	<.001	11.1950
Reaction Time	-0.1085	<.001	-2.5262	<.001	12.1324
Normalized Reaction Time	-0.0161	<.001	-0.3059	<.001	1.6695
Optimal Percentage	0.0011	.021	0.0393	<.001	0.4599
Inner Optimal Percentage	0.0005	.460	0.0308	<.001	0.4982
Outer Optimal Percentage	0.0008	.288	0.0257	<.001	0.3676
Last Optimal Percentage	0.0012	.073	0.0752	<.001	0.5774

Chapter 5

Chapter 5: Model Fitting and Model Simulation

5.1 Possible Models

We have reviewed several classic models in Chapter 2. But what kind of model is possible in this specific task? We will firstly give some intuitions, detailly describe their algorithms and make hypotheses in this section, then do model fitting in Section 5.2. Finally, we will see whether the best model could capture several features in participants' data in Section 5.3 with Model Simulation.

5.1.1 Model-Free Models

Model-Free algorithms in this task need to maintain three different Q value tables for three tasks because the goal state has three possibilities: A, B and C. Therefore, we should create 3 Q value tables for two kinds of classic Model-Free algorithm, SARSA and Q-learning. The difference between SARSA and Q-learning is that when calculating reward prediction error (RPE), SARSA uses the true action chosen by participants in the next timestep, while Q-learning uses the action with maximum Q value on the next state. We rewrite the update rule for SARSA and Q-learning here in Equation 5.1 and 5.2, in which α is learning rate, γ is temporal discounting factor.

$$\begin{aligned} \delta_{RPE} &= r(s') + \gamma Q_{SARSA}(s', a') - Q_{SARSA}(s, a) \\ Q_{SARSA}(s, a) &= Q_{SARSA}(s, a) + \alpha \delta_{RPE} \end{aligned} \tag{5.1}$$

$$\begin{aligned} \delta_{RPE} &= r(s') + \gamma \max_{a' \in a(s)} Q_Q(s', a') - Q_Q(s, a) \\ Q_Q(s, a) &= Q_Q(s, a) + \alpha \delta_{RPE} \end{aligned} \tag{5.2}$$

What's more, considering the difficulty of this task, we add forget rate into the model as Equation 5.3 in which f represents forget rate ranging from 0 to 0.05. Note that the reason why the maximum f is only 0.01 is that we assume forget takes place at every timestep in every trial. If it takes the participants averagely 10 steps to get to the end state, then after 10 trials the participants will almost totally forget the Q value of that state ($(1 - 0.05)^{10 \times 45} \approx 0.005$).

$$Q(s, a) = Q(s, a) \times (1 - f) \tag{5.3}$$

The decision probability is determined by softmax's transformation of Q value according to Equation 5.4, in which τ is inverse temperature determining how much the participants will prefer the action with higher Q value. If $\tau = 0$, each action will

receive a same probability to be chosen. If $\tau = \infty$, participants will definitely choose the action with highest Q value.

$$P(a) = \frac{e^{\tau Q(s,a)}}{\sum_a e^{\tau Q(s,a)}} \tag{5.4}$$

5.1.2 Model-Based Models

Unlike Model-Free algorithm, Model-Based algorithm only maintain one set of transition matrix. In this task, we assume that participants do not need to learn reward function, since the reward function is rather simple: rewards in the end state (depend on how many steps it takes) and 0 elsewhere. The transition learning method we apply is similar to Gläscher et al., 2010 with a state prediction error (SPE). The update rule is shown in Equation 5.5, in which η is learning rate, $s' is the observed new state$. It is obvious that after every update of T, $\sum_{s''} T(s,a) = 1$, keeping the distribution normalized.

$$\begin{aligned} \delta_{SPE} &= 1 - T(s,a,s') \\ T(s,a,s') &= T(s,a,s') + \eta\delta_{SPE} \\ T(s,a,s'') &= T(s,a,s'') \times (1-\eta) \quad \forall s'' \neq s' \end{aligned} \tag{5.5}$$

In the Model-Based methods, participants need to do extra calculation to gain value estimation. Gläscher et al. (2010) use a dynamic programming (see Equation 5.6) to calculate the Q value using transition matrix and reward information. We apply the same method in our pure Model-Based methods, but it is worth noting that dynamic programming is viable in their experiment because it is only a simple two stage decision with now states shared between stages. However, this task of ours is much complicated and the same state may be visited more than once in a trial, which makes dynamic programming much harder to converge. Actually, in Gläscher et al.'s experiment it is guaranteed to be convergent with two iterations. But in our experiment settings, it seems there is no clear guarantee of convergence. Considering the computation limit of human brain, we still iterate the dynamic programming equation two times, but reader should be aware of the limit of this method. This method is actually a simplified value iteration algorithm in Computer Science Sutton and Barto, 1998. There does exist other algorithms such as policy iteration, but they suffer from the same problem of computation complexity. Therefore, we only consider value iteration in pure Model-Based method in this thesis.

$$Q(s,a) = \sum_{s'} T(s,a,s') \times (r(s') + \arg\max_{a'} Q(s',a')) \tag{5.6}$$

Similar to Model-Free method, we add forget rate in Model-Based Models as well. The difference between Model-Free forget and Model-Based forget is that Model-Free forget is toward 0 while Model-Based forget in our experiment is toward $\frac{1}{6}$, since the uniform distribution of transition will result in $\frac{1}{6}$. The forget process is given by Equation 5.7, in which f is the forget rate.

$$T(s,a,s') = (\frac{1}{6} - T(s,a,s')) \times f + T(s,a,s') \tag{5.7}$$

After the Model-Based value has been calculated, action selection follows the same softmax transformation as Equation 5.4.

5.1.3 Hybrid Model

A Hybrid learner combines state-action value estimates from both Model-Free and Model-Based. This model assumes that human brain has two kinds of learning mechanism, model-free and model-based, and when making decisions they produce two value estimation simultaneously. Following Bucci, Holland, and Gallagher (1998), we characterize the weight with an exponential function (see Equation 5.8).

$$w_t = I \times e^{-kt} \tag{5.8}$$

Q value calculation for Hybrid model is shown in Equation 5.9.

$$Q_{HYB}(s,a) = w_t \times Q_{MB}(s,a) + (1 - w_t) \times Q_{MF}(s,a) \tag{5.9}$$

After the hybrid value has been calculated, action selection follows the same softmax transformation as Equation 5.4.

5.1.4 Model-Based Help Model-Free Model

Model-Free models suffer from incorrect inference. For example, the participant is now at E and his goal is B. At this time the participant choose the action whose primary consequence is D but with 0.2 probability it is transited to B, and he win the reward. In this situation, Model-Free models will credit the action whose primary consequence is D, which is obviously unwise. However, since Model-Based methods is aware of which action's primary consequence is the reward state, it should be able to "help" Model-Free methods to fix this kind of wrong credit. Thus, the Model-Based Help Model-Free Model, or in short MBHMF Model, does Model-Free and Model-Based learning simultaneously as Hybrid model, but it only relies on Model-Free prediction to do decision making. On the other hand, when Model-Free is updating its value, it will check whether the action chosen by participant is the most likely transit-to-the-new-state one. If it is not, and another action has more than 0.7 probability to transit to this new state, it will fix the mistake. The mathematical formulation of mistake fixing is shown in Equation 5.10.

$$\begin{aligned}
&\delta_{RPE} = r(s') + \gamma \max_{a' \in a(s)} Q_Q(s',a') - Q_Q(s,a'') \\
&Q(s,a'') = Q(s,a'') + \alpha\delta_{RPE} \\
&T(s,a'',s') > T(s,a''',s') \quad \forall a''' \neq a'' \\
&T(s,a'',s') > 0.7
\end{aligned} \tag{5.10}$$

Action selection of MBHMF model follows the same softmax transformation as Equation 5.4.

5.1.5 Model-Free Help Model-Based Model

Model-Based could fix mistakes in Model-Free learning. Can Model-Free help Model-Based method as well? The answer is yes.

First of all, Model-Free could help reducing the extreme computational burden when calculating value estimation. In fact, Model-Free value estimation could help Model-Based Model doing successor matrix calculation, which could easily be turned

to value by right dot product reward vector. This method is named Model-Free Help Model-Based (MFHMB) model. Its value estimation is totally determined by Model-Based calculation.

Specifically, the successor matrix S is given by Equation 5.11. Successor matrix represents from current state, taking action a, how much probability I will be in every state. I_t in the equation means the probability of each state after t step transitions. The first equation in 5.11 shows that in one step, the successor matrix is just equal to the transition matrix on current state and specific action. The second equation represents that after first transition, the action chosen by algorithm will be the action that has maximum Model-Free Q value, resulting in a greedy policy. After the action have been chosen, the second equation could calculate the probability of each state's probability after two steps transition from current state. Then the third, fourth transition can be calculated in the same manner. Successor matrix $S(a,s)$ is just a sum of all I_t with temporal discounting. As the basic Model-Based method, we could just calculate two steps from current state to match the computational ability of human brain. Previous Model-Based value estimation's computation complexity is $\mathcal{O}(S^2AI)$, in which S is the number of all states, A is the number of all actions and I is iteration time. However, successor matrix calculation's computation complexity is decrease to $\mathcal{O}(SAI)$.

$$\begin{aligned} &I_1(a,s') = T(s,a,s') && \text{s is the current state} \\ &I_t(a,s') = I_{t-1}(a,s) \times T(s,a',s') \quad a' = \arg\max_a Q_{MF}(s,a) && t = 2,3,... \\ &S(a,s) = \sum_{t=1}^{\infty} \gamma^{t-1} I_t && \text{s is sucessor state} \end{aligned} \tag{5.11}$$

Successor matrix could easily be transformed to value by Equation 5.12, in which R is the reward function. Note that s could only be the current state since we do not calculate successor matrix from other states.

$$Q(s,a) = \sum_{s'} S(a,s')R(s') \tag{5.12}$$

Action selection of MFHMB model follows the same softmax transformation as Equation 5.4.

5.1.6 Conclusion of Models

In this section we introduced five models, including MF model, MB model, Hybrid model, MBHMF model, MFHMB model. They are summarized in Table 5.1 and their parameters are summarized in Table 5.2. Readers could easily observe that MF model and MB model is two base models while other two models are built upon them. In the meantime, the idea of Hybrid, MBHMF, MFHMB are not mutual-exclusive, which means they could exist as a whole. Due to the time limit, we could not examine models combining two or three ideas but leave this as a feature work direction.

TABLE 5.1: Possible Models

Model Name	Abbreviation	Parameters
Model-Free	MF	$\alpha, \tau, \gamma, f_{MF}$
Model-Based	MB	$\eta, \tau, \gamma, f_{MB}$
Hybrid	Hybrid	$\alpha, \eta, \tau, \gamma, f_{MF}, f_{MB}, I, k$
Model-Based Help Model-Free	MBHMF	$\alpha, \eta, \tau, \gamma, f_{MF}, f_{MB}$
Model-Free Help Model-Based	MFHMB	$\alpha, \eta, \tau, \gamma, f_{MF}, f_{MB}$

TABLE 5.2: Model Parameters

Parameter Name	Meaning	Bound
α	learning rate of MF	[0, 1]
η	learning rate of MB	[0, 1]
τ	inverse temperature in softmax	[0, ∞)
γ	temporal discounting rate	[0, 1]
f_{MF}	forget rate of MF	[0, 0.05]
f_{MB}	forget rate fo MB	[0, 0.05]
I	hybrid exponential function parameter	[0, 1]
k	hybrid exponential function parameter	[0, 0.1]

It is also worth mentioning that MB and MF could also assist or interfere with each other in many other ways. We have tried several other possibilities, such as MF limits MB learning (MB only learns when it is valuable). Since it has been plenty of evidence that MB and MF co-exist in our decision system, readers should be aware that there could exist pretty much possibilities that MB and MF cooperate or compete with each other we can't cover in this thesis.

5.2 Model Fitting

All the data fitting procedures were conducted on the individual level using maximum likelihood estimates. We use a function `optimize.minimize` in SciPy(Jones, Oliphant, and Peterson, 2001–) to search for the parameters that minimized minus log likelihood. To verify that we had found the global minimum, we repeated the search process using 1000 different starting points.

We used Akaike information criterion with a correction for sample sizes (AICc) to do model comparison (Akaike, 1974; Bucci, Holland, and Gallagher, 1998).

For each participant, we gained 5 AICc for 5 models. We then calculate the least AICc model for each participant. Firstly we compare Model-Free's 2 kinds of algorithms: SARSA and Q-learning. There are 33/36 participants that Q-learning has smaller AICc, indicating that Q-learning is much more possible and SARSA. Therefore, we will use Q-learning to complete the model free part in the Hybrid model, MBHMF model and MFHMB model.

TABLE 5.3: Model Comparison

Model Name	Best count	∑AICc
Model-Free	16	38275
Model-Based	0	42848
Hybrid	17	38222
Model-Based Help Model-Free	3	38369
Model-Free Help Model-Based	0	42941

Secondly we compare whether adding forget will result in a better fit. The data shows that 29/36 participants has smaller AICc when forget is considered. Moreover, we could see that in the block condition of Fig. 4.4, 4.5, 4.6, 4.7, the last block performance is worse than previous three one on the same task, suggesting there exists forget. Therefore, we could conclude that forget is a key part of model for our experiment and we will add forget into the Hybrid model, MBHMF model and MFHMB model.

Finally we compare our five major models. The comparison method is the same as SARSA vs Q-learning, to check which one of 5 models has least AICc on individuals. The result is summarized in Table 5.3. We could see that Hybrid has most best model results of 17, MF 16 and MBHMF 3. The AICc sum over all participants's result is consistent with best count result. We will carry out further analyses in the next section "Model Simulation" towards this result.

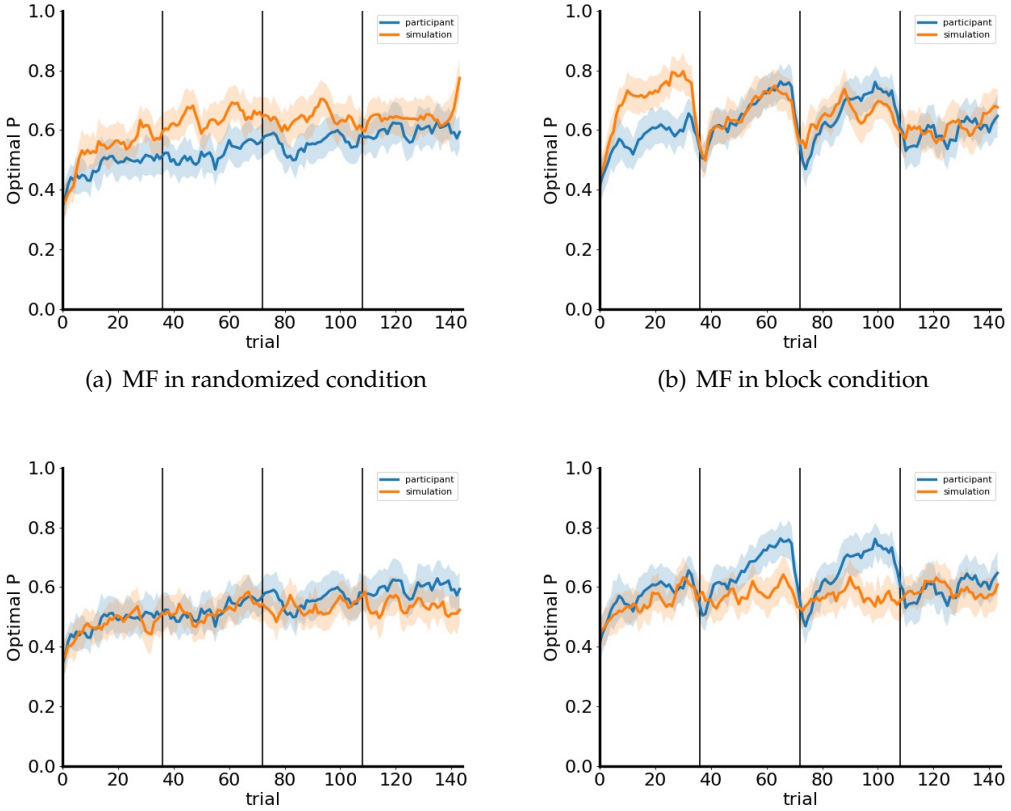
5.3 Model Simulation

In this section, we want to see whether our two best models, Hybrid and Q-learning with forget, could recreate patterns participants show in both randomized and block conditions.

All data simulation uses the params from model fitting maximum likelihood method. Therefore, we create one simulation for each participant using their parameters under Model-Free Q-learning with forget model and Model-Based with forget model. The reason why we choose MB rather than hybrid is that hybrid decision has much more stochasticity that could not reveal clear data pattern, but combining MF and MB features we could easily imagine how hybrid model performs. It is also worth noting that one simulation has strong stochasticity both due to action selection and environment transition randomness. The reason why we do not repeat the simulation several times to use the mean value is that participant's data is also only one observation. Mean values will be much smoother than participant's data because of averaging operation.

The comparison between simulation and participant data in all four optimal percentage metrics is shown in Fig. 4.4, 4.5, 4.6 and 4.7.

It is clear that MF model could explain data in block condition while MB seems more fit to randomized condition in all four metrics. No wonder the hybrid model



(a) MF in randomized condition

(b) MF in block condition

(c) MB in randomized condition

(d) MB in block condition

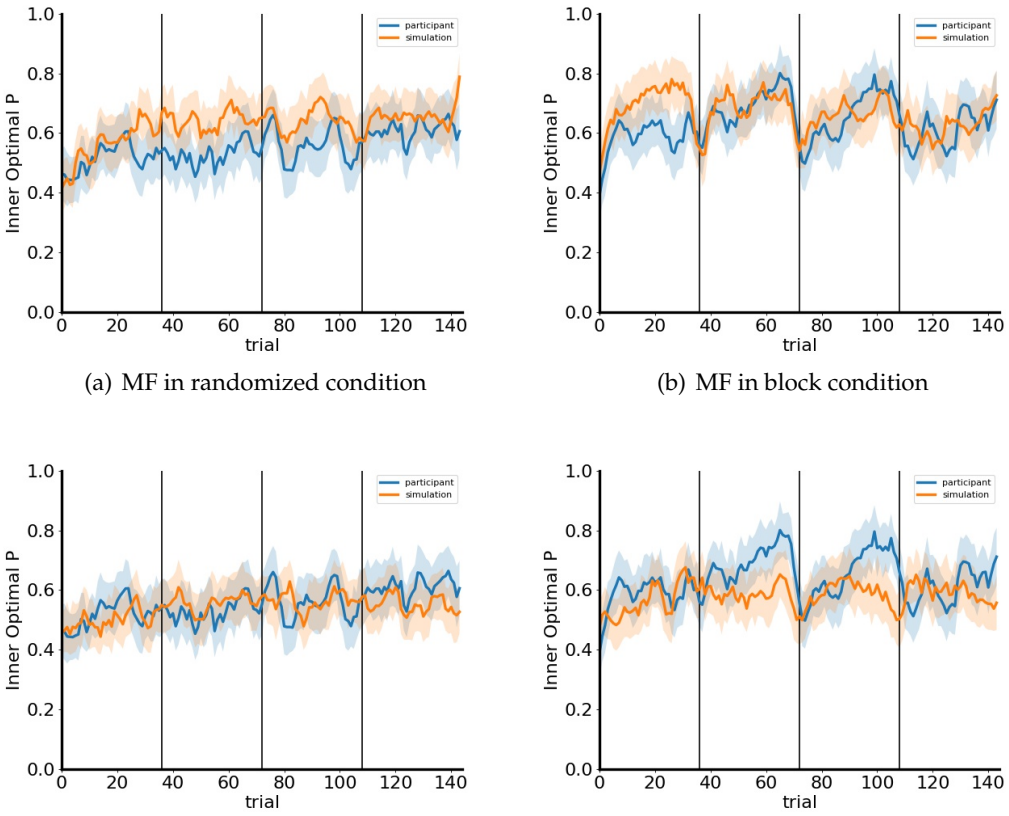
FIGURE 5.1: Optimal Percentage Simulation Participants Comparison. Blue line indicates participant mean data, orange line indicates simulation mean data. Shadow indicates standard error.

could get the most best count and lowest ∑AICc since it could combine the advantages of both MF and MB method.

However, there still exist features that MF could not explain in the block condition. The first block's learning for participants seems much slower and worse than simulation in all four metrics. We will discuss this phenomenon in Section 6. Moreover, as we stated in previous sections, randomized condition is harder than block condition, and the MB-seemingly-fit-better phenomenon could just be a illusion because of the inefficiency in learning for both simulation and participants.

We also did model simulation in different parameters to see how will the data pattern change as the parameters change. Due to space limit, we won't put these graphs here.

First of all, if the learning rate α or τ is close to 1, the performance will fluctuate a lot since it suffer from the stochasticity of environment transition. On the other hand, if the learning rate is close to 0, the learning will be much slower than the participants do. Therefore, a learning rate between 0.3 and 0.5 seems great in this task.



(a) MF in randomized condition

(b) MF in block condition

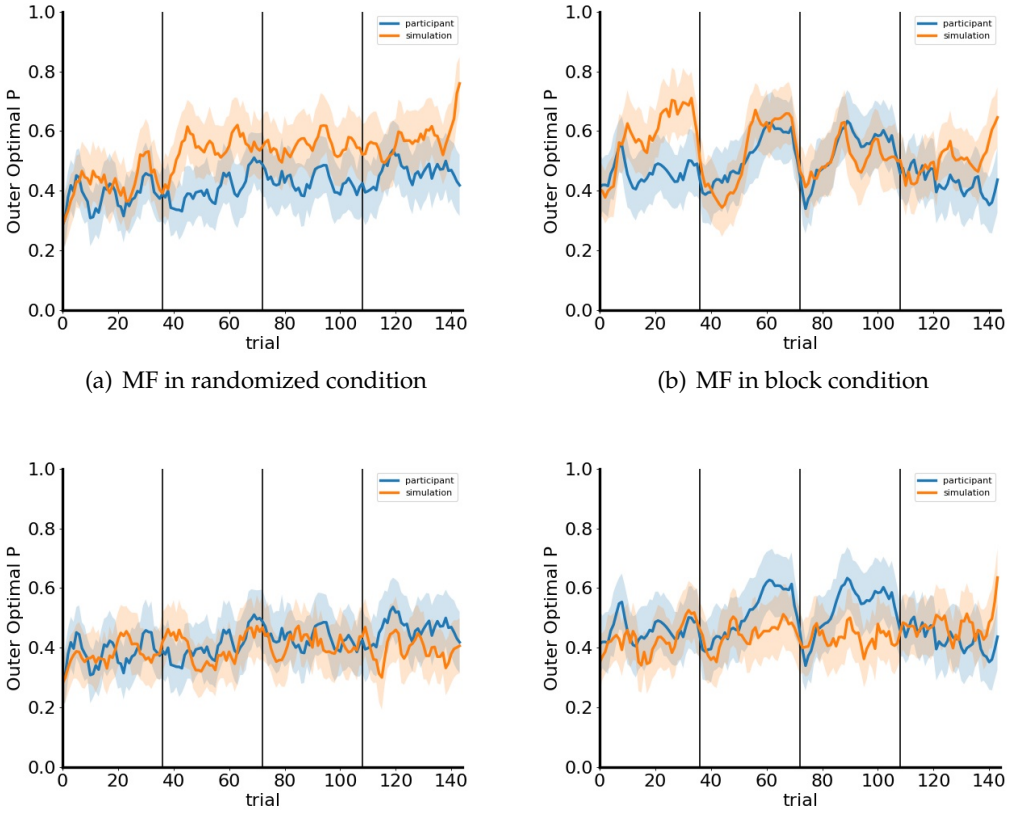
(c) MB in randomized condition

(d) MB in block condition

FIGURE 5.2: Inner Optimal Percentage Simulation Participants Comparison. Blue line indicates participant mean data, orange line indicates simulation mean data. Shadow indicates standard error.

Secondly, the softmax inverse temperature τ should never be too big or too small. If τ is close to 0, then the action selection is uniformly random regardless of value learned. If τ is bigger than 10, then the action selection will be too stable for the player, making him stuck in local optimum value.

Finally, the forget rate. The average forget rate for participants is 0.0123, which means participants will totally forget in about 40 trials. This is reasonable because too small forget rate may result in sticking in local optimum since it can not forget the information, or transfer wrong knowledge to the following tasks.



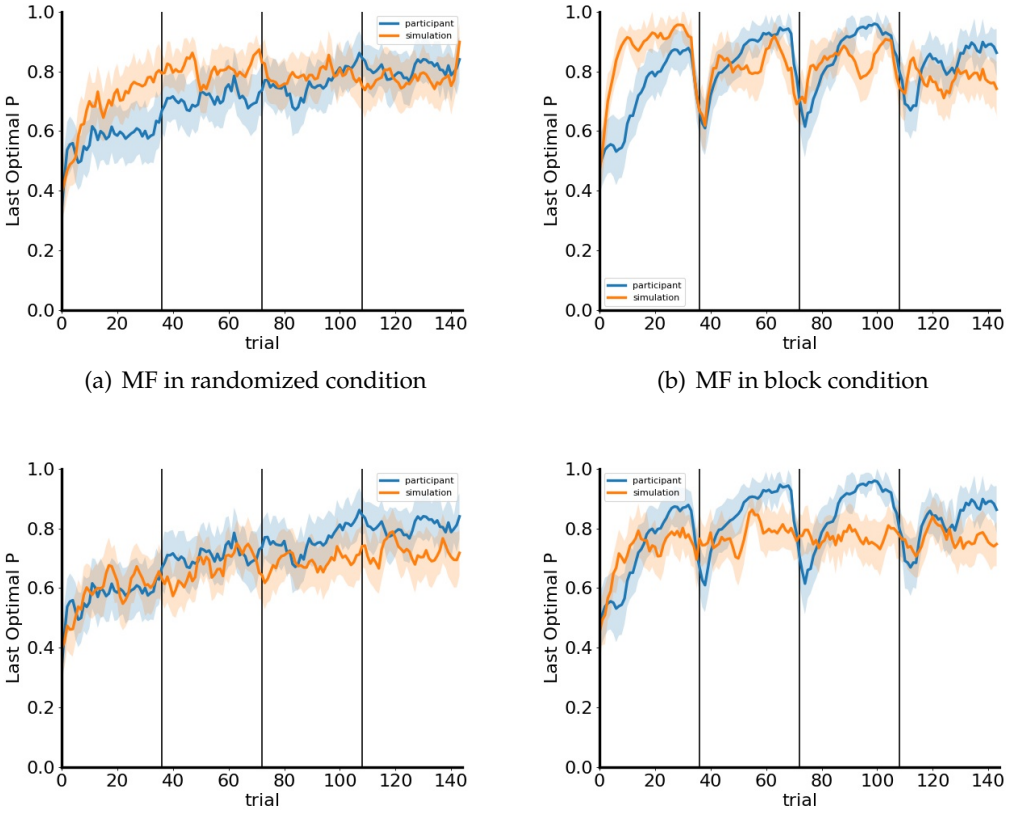
(a) MF in randomized condition

(b) MF in block condition

(c) MB in randomized condition

(d) MB in block condition

FIGURE 5.3: Inner Optimal Percentage Simulation Participants Comparison. Blue line indicates participant mean data, orange line indicates simulation mean data. Shadow indicates standard error.



(a) MF in randomized condition

(b) MF in block condition

(c) MB in randomized condition

(d) MB in block condition

FIGURE 5.4: Inner Optimal Percentage Simulation Participants Comparison. Blue line indicates participant mean data, orange line indicates simulation mean data. Shadow indicates standard error.

Chapter 6

Chapter 6: Discussion and Conclusion

6.1 Discussion

Firstly, in the model comparison results, it is clear that hybrid and Model-Free method beat other models in both best counts and sum AICc. However, there still exists several insights behind the analysis.

It is worth noting that Q-learning fits much better than SARSA, and Q-learning with forget rate fits much better than Q-learning without forget rate. It is common to add forget rate to decision making models, but it seems little RL papers ever adding forget rate to either MF or MB models. Our data has demonstrated clearly that Q value may be forgotten.

Model-Based Centered algorithms including MB and MFHMB have worst model comparison result. The reason may be that even the MFHMB has successfully decreased computation complexity, it is still very hard to compute precisely. What's more, human's ability of probability representation is not so precise because of cognitive bias such as framework effect. These two reasons make the value estimation of Model-Based algorithms unstable and untrustworthy. Future work may design a measure in the experiment to see whether participants learn wrong probability or they learn the correct probability information but mistakenly calculate the Q value.

The two best models are the Hybrid model and MF model, and it seems that Hybrid model and MF model do not have task specification (e.g. Hybrid is not significantly better at one task and neither do MF). This result indicates that human does use both MF and MB method to tackle the sequential decision making task. However, it is still unclear how they might cooperate or compete. Even if MF seems to explain all the data features in the simulation period, it could not explain the first block result. As a matter of fact, we think it may be the evidence of meta-learning since participants do not do what to learn and how to tackle the problem in the first block, they will need meta learning to find a possible solution to the task. It is also possible that the value learning is value driven. There might exist a value threshold below which participants will not learn since human working memory is limited and only a very small amount of information could be saved and updated frequently. Value threshold is also one kind of meta learning since participants use value threshold to capture what is important and should be learnt. Further analysis and experiment are needed to fully address the question.

The third best model is MBHMF model, though it has only 3 best AICc. This may indicate that the Model-Based Help Model-Free to fix mistakes. Actually the AICc of MBHMF is very close to MF model. There exists a problem that we manually set the threshold of confidently replace old action being updated. If we could relax this constraint to make the threshold a free variable, maybe MBHMF model will have a better performance.

There exists another possible model: participants may never build any value or transition matrix in their brain. They just randomly pick actions until a good sequence being spotted and remembered. For example, a participant may happen to choose the optimal action sequence to reach the goal. Then he remembered the action sequence, and carry out this action sequence with 100% probability. If they are transferred to primary sequence he will successfully win 18 tokens with probability $0.7^2 = 0.49$, but if not, he will randomly choose action again because he only knows how to move from start state to end state by pressing an action consequence. Future work could examine the model comparison result for these kinds of heuristic methods as well.

6.2 Conclusion

In this thesis, we introduce Model-Free and Model-Based classic algorithms for Markov Decision Process. Then we design a relatively complex "labyrinth"-like environment to see how human behave in a complicated environment. The result shows participants do learn something and their performance keeps getting better along the trial set. We also fit five models to the participants' data. The result demonstrates that Q-learning is a better fit than SARSA, adding forget rate will become better fit for data, and Hybrid and Model-Free is two most convincing models given data but Model-Based Help Model-Free model are also useful. Randomized task's participant data could almost be explained by MB model and block task's participant data could almost be explained by MF model. Therefore, future research could focus on how to explain the first block learning in the block condition, and the difference in learning procedure for the two different task orders.

Acknowledgements

I would like to thank my mentor Dr. Hang Zhang for her helpful discussions and warm attitude. Doing research is never so happy!

I also want to thank all members of CDLab for your assistance and always-smile greeting. Coding in lab is never so happy!

I would like to thank my senior, Dian Xiong. You made me realize that everything is possible.

I would like to thank all the people that helped me. It is because of your help that makes me go this far.

I would like to thank my parents. You gave me the most parents could ever give.

Finally, I would like to thank my dear Zhuoting. Without you, I will never become who I am.

Bibliography

Akaike, Hirotugu (1974). "A new look at the statistical model identification". In: *IEEE transactions on automatic control* 19.6, pp. 716–723.

Bayer, Hannah M and Paul W Glimcher (2005). "Midbrain dopamine neurons encode a quantitative reward prediction error signal". In: *Neuron* 47.1, pp. 129–141.

Bellman, Richard (1957). "A Markovian decision process". In: *Journal of Mathematics and Mechanics*, pp. 679–684.

Botvinick, Matthew et al. (2015). "Reinforcement learning, efficient coding, and the statistics of natural tasks". In: *Current Opinion in Behavioral Sciences* 5, pp. 71–77.

Botvinick, Matthew Michael (2012). "Hierarchical reinforcement learning and decision making". In: *Current opinion in neurobiology* 22.6, pp. 956–962.

Bucci, David J, Peter C Holland, and Michela Gallagher (1998). "Removal of cholinergic input to rat posterior parietal cortex disrupts incremental processing of conditioned stimuli". In: *Journal of Neuroscience* 18.19, pp. 8038–8046.

Daw, Nathaniel D and Peter Dayan (2014). "The algorithmic anatomy of model-based evaluation". In: *Phil. Trans. R. Soc. B* 369.1655, p. 20130478.

Daw, Nathaniel D, Yael Niv, and Peter Dayan (2005). "Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control". In: *Nature neuroscience* 8.12, p. 1704.

Daw, Nathaniel D et al. (2011). "Model-based influences on humans' choices and striatal prediction errors". In: *Neuron* 69.6, pp. 1204–1215.

Dayan, Peter and Bernard W Balleine (2002). "Reward, motivation, and reinforcement learning". In: *Neuron* 36.2, pp. 285–298.

Doya, Kenji et al. (2002). "Multiple model-based reinforcement learning". In: *Neural computation* 14.6, pp. 1347–1369.

Garrison, Jane, Burak Erdeniz, and John Done (2013). "Prediction error in reinforcement learning: a meta-analysis of neuroimaging studies". In: *Neuroscience & Biobehavioral Reviews* 37.7, pp. 1297–1310.

Gershman, Samuel J and Nathaniel D Daw (2017). "Reinforcement learning and episodic memory in humans and animals: an integrative framework". In: *Annual review of psychology* 68, pp. 101–128.

Gläscher, Jan et al. (2010). "States versus rewards: dissociable neural prediction error signals underlying model-based and model-free reinforcement learning". In: *Neuron* 66.4, pp. 585–595.

Hollerman, Jeffrey R and Wolfram Schultz (1998). "Dopamine neurons report an error in the temporal prediction of reward during learning". In: *Nature neuroscience* 1.4, p. 304.

Jones, Eric, Travis Oliphant, Pearu Peterson, et al. (2001–). *SciPy: Open source scientific tools for Python*. [Online; accessed <today>]. URL: http://www.scipy.org/.

Kishida, Kenneth T et al. (2016). "Subsecond dopamine fluctuations in human striatum encode superposed error signals about actual and counterfactual reward". In: *Proceedings of the National Academy of Sciences* 113.1, pp. 200–205.

Mnih, Volodymyr et al. (2015). "Human-level control through deep reinforcement learning". In: *Nature* 518.7540, p. 529.

Sutton, Richard S and Andrew G Barto (1998). *Introduction to reinforcement learning*. Vol. 135. MIT press Cambridge.

Thorndike, Edward L (1927). "The law of effect". In: *The American Journal of Psychology* 39.1/4, pp. 212–222.

Waelti, Pascale, Anthony Dickinson, and Wolfram Schultz (2001). "Dopamine responses comply with basic assumptions of formal learning theory". In: *Nature* 412.6842, p. 43.

北京大学学位论文原创性声明和使用授权说明

原创性声明

本人郑重声明： 所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含任何其他个人或集体已经发表或撰写过的作品或成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本声明的法律结果由本人承担。

论文作者签名： 日期： 年 月 日

学位论文使用授权说明

本人完全了解北京大学关于收集、保存、使用学位论文的规定，即：

按照学校要求提交学位论文的印刷本和电子版本；

学校有权保存学位论文的印刷本和电子版，并提供目录检索与阅览服务；

学校可以采用影印、缩印、数字化或其它复制手段保存论文；

在不以赢利为目的的前提下，学校可以公布论文的部分或全部内容。

（保密论文在解密后遵守此规定）

论文作者签名： 导师签名：

日期： 年 月 日