Reinforcement Learning Models

Omer Faruk YILDIRAN

February 2023

1 Problem 1: Rescorla-Wagner Model

By the presence or absence of a conditioned stimulus, an animal learns to predict the presence of an unconditioned stimulus (UCS), which is explained by the Rescorla-Wagner model (CS). A parameter termed "w" in this model represents the animal's forecast of the UCS's existence. The Rescorla-Wagner learning rule, which modifies the weight based on the prediction error (the discrepancy between the actual outcome and the projected outcome) and a learning rate parameter, is used to update this parameter after each trial.

(a) Basic stimuli-reward relation

To generate a set of stimuli and rewards for this problem, we can create two arrays 'u' and 'r' of length 50 each. The first 25 elements of both arrays can be set to 1, indicating that both the stimuli and reward are present. The next 25 elements of 'stimuli(u)' can be set to 1, while the next 25 elements of 'reward(r)' can be set to 0, indicating that only the stimuli are present during these trials.

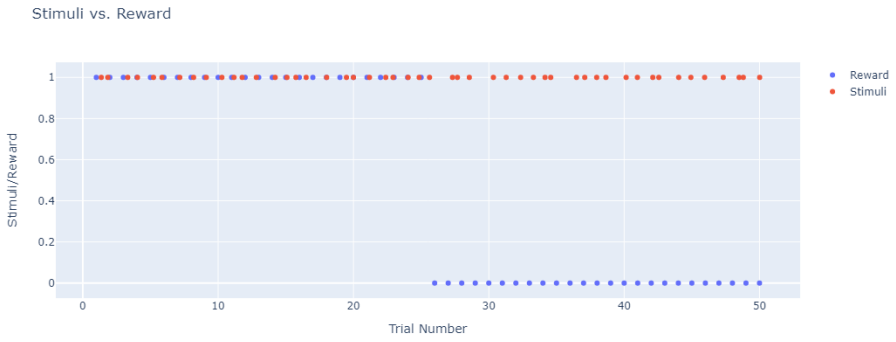


Figure 1: Presence or abscence of stimuli and reward throught 50 trials

1.1 (b) Using the learning rule

The formula for updating the parameter "w" after each trial is "w = w + e * u * d," where "e" stands for the learning rate (set to 0.1), "u" stands for the stimulus, and "d" stands for the prediction error, which is defined as "r - v," where "r" stands for the reward and "v" stands for the predicted outcome.

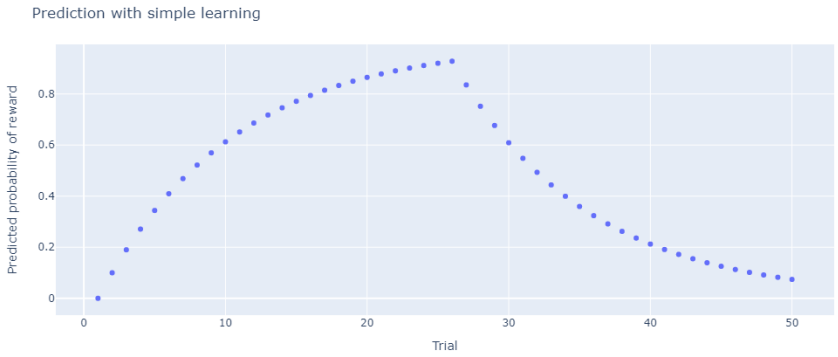
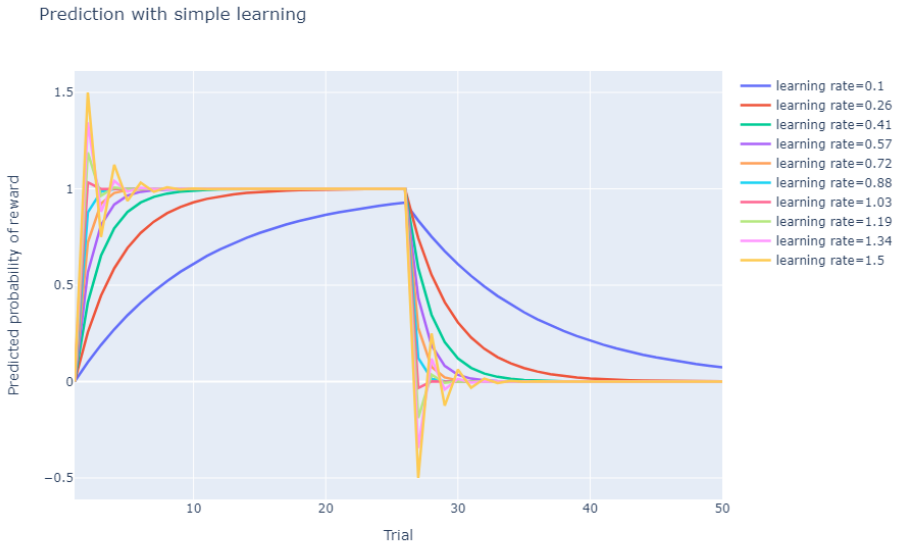


Figure 2: Prediction with simple learning when we take e as 0.1

c) What happens if we change the learning parameter?

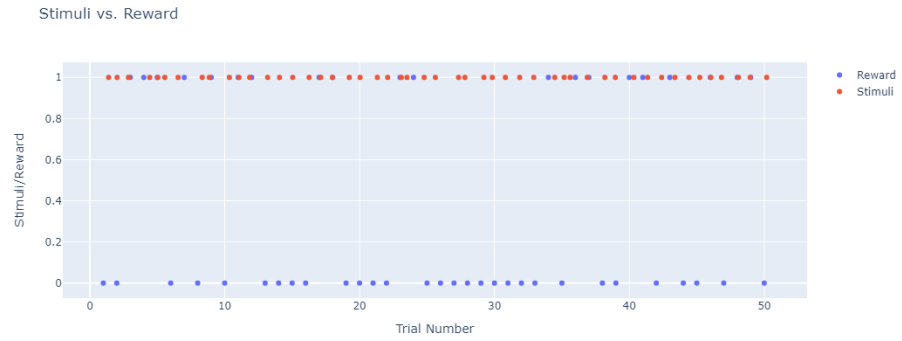
When we increase the learning rate from 0.1 to higher numbers such as 0.7 or 1.0, the prediction value quickly reaches 1 and stays flat as long as the reward is present. Once the reward is absent, the prediction value sharply decreases to 0. Moreover, when the learning rate is set to 1, the prediction value immediately reaches 1 and stays flat until the reward is no longer present. Then, when the reward is absent, the prediction value quickly drops to 0. Even more interestingly if we take extreme values for the learning rate, it sometimes goes beyond 1 and below 0 (e¿1). This pattern of learning is consistent with the Rescorla-Wagner model and demonstrates how the animal is able to learn to predict the occurrence of the UCS based on the presence or absence of the CS.



Prediction with simple learning, in case learning rate is from various numbers

d) Partial Learning

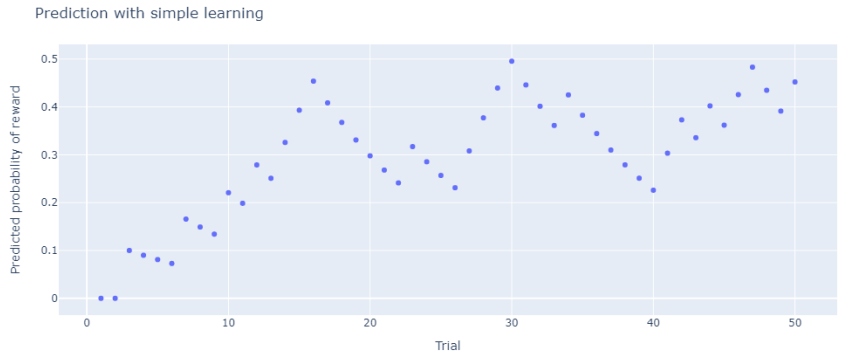
With partial conditioning, the reward may not always be present in a trial. In this instance, the reward's existence is a chance occurrence with a probability of 0.4. The rewards will be produced randomly based on this probability, but we can still generate the same collection of stimuli as before. The resulting plot will show the presence/absence of the stimuli and rewards as a function of trial index.



Stimuli/Reward plot in case the presence of reward order is random and probability of reward is 40 percent

The anticipated results in this simulation are lower and more unpredictable than in the preceding simulation with complete conditioning, as seen by the

plot. This is due to the randomization in the reward schedule, which prevents the animal from accurately predicting the existence of the UCS based on the stimuli alone. Although the projected results increasingly improve with each trial, the model still demonstrates some learning over time. This is due to the fact that the Rescorla-Wagner model, even in the face of noise or uncertainty in the reward schedule, allows for incremental adjustments to the weight parameter based on the prediction error.



Prediction with simple learning, in case the presence of reward order is random and probability is 40 percent

e) Blocking

In the Rescorla-Wagner model, the prediction of the animal is based on the sum of the products of each stimulus and its corresponding learned parameter. In the case of two stimuli, u_1 and u_2, the animal's prediction is given by $v = w_1u_1 + w_2u_2$.

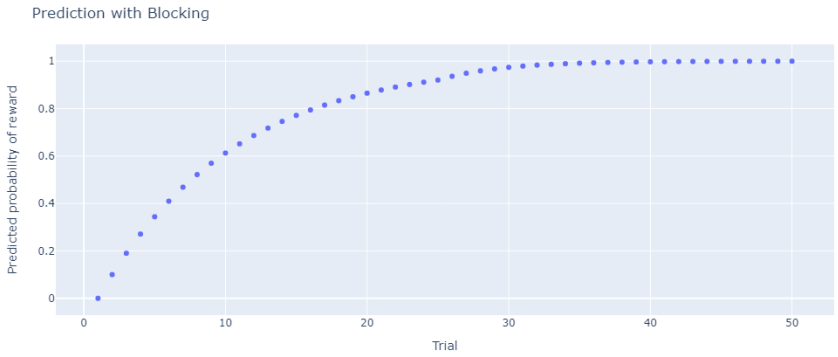
Assume that there is only one stimulus (u1) and one reward (r) for the first 25 trials. The animal's prediction is therefore v = w1u1. We use the Rescorla-Wagner learning rule to update the value of w1 following each trial: w1 = w1 + e * u1 * d where d = r - v is the prediction error, u1 is the stimulus, and e is the learning rate.

The reward and the stimulus are both present for the subsequent 25 trials. The forecast of the animal is v = w1u1 + w2u2. We can set the initial value of w2 to 0 and update it similarly to w1:

w2 = w2 + e * u2 * d

where d = r - v is the prediction error, which now depends on both stimuli, and u2 is the new stimulus.

Based on the data of this experiment, when both stimuli are present, the value of w1 will be learned during the first 25 trials and will not change over the next 25 trials. Hence, regardless of the availability of the second stimulus u2, the animal will make the same prediction during the second set of trials. Because the animal has already learnt to identify the reward with the initial stimulus, this phenomenon—known as "blocking"—occurs because it prevents the animal from updating the value of w2 during the second series of trials.



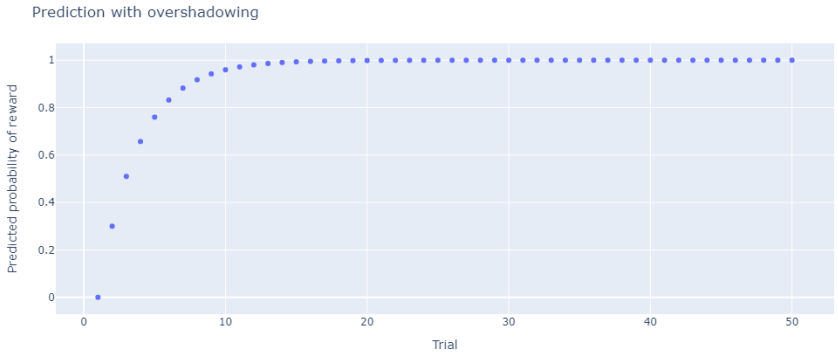
Prediction with block of two stimuli

By introducing blocking, we can see that the animal learns to associate the reward with the first stimulus and ignores the second stimulus during the first 25 trials. When both stimuli are presented during the next 25 trials, the animal already has a strong association between the first stimulus and the reward, so it learns more slowly to associate the second stimulus with the reward. As a result, we see a smaller increase in the predicted reward for the second stimulus during the second half of the experiment.

1.2 g) Overshadowing

In the case of overshadowing, one stimulus may have a larger impact on the animal's prediction than the other stimulus, even though both stimuli are present during the conditioning phase.

Let's assume that there are two stimuli, u_1 and u_2, and two corresponding parameters to learn, w_1 and w_2. The animal's prediction is given by $v = w_1u_1 + w_2u_2$. Let's also assume that both stimuli and the reward are present from the beginning of the experiment.



Prediction with overshadowing of two stimuli

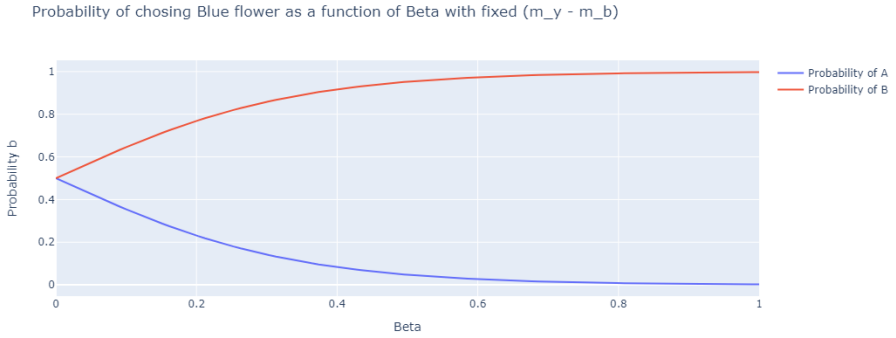
If we set the learning rate for u_1 to be larger than the learning rate for u_2, then w_1 will be updated more rapidly than w_2. This means that u_1 will have a greater impact on the animal's prediction than u_2, since w_1 will be updated more frequently and to a greater extent than w_2. As a result, the animal may learn to associate the reward more strongly with u_1 than with u_2, even though both stimuli are present during the conditioning phase.

In summary, if one stimulus has a larger learning rate than the other, it may overshadow the other stimulus and have a greater impact on the animal's prediction.

2 Problem 2: Simple decision strategy for flower sampling by bees

.

The phrase "exploitation-exploration trade-off" refers to the idea that when making decisions, an agent must choose through using their present knowledge to maximise short-term gains and exploring new possibilities to learn more and maybe increase long-term gains (exploration). The best decision-making technique focuses on balancing the risk of losing out on prospective high-reward possibilities against the cost of putting resources into studying unproven options. This trade-off is frequently encountered in situations where the environment is unpredictable or dynamic.



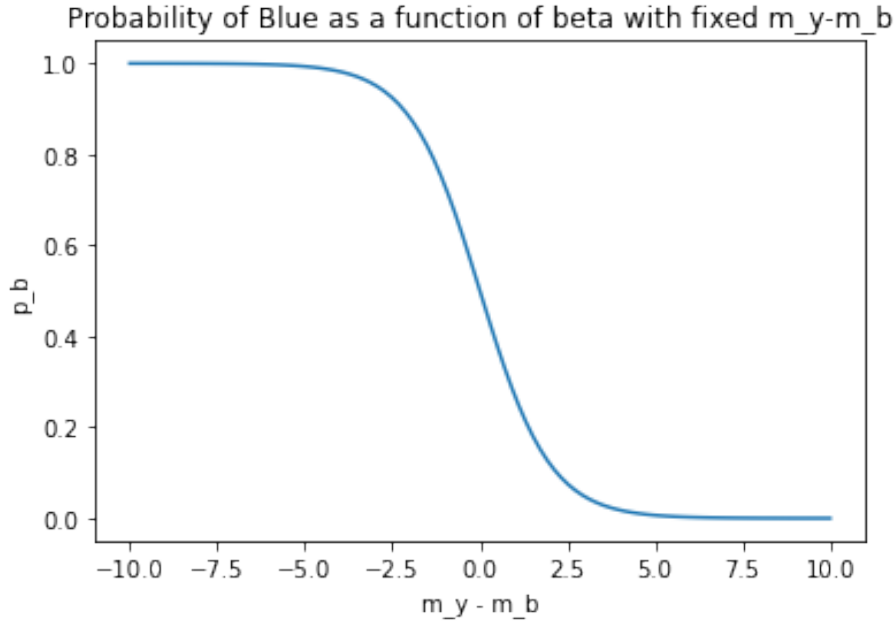
Given a fixed value for the parameter Beta, we may plot the probability of selecting a blue flower as a function of the distinction between the estimated rewards for blue and yellow flowers to visualize this trade-off. We utilize the softmax function specifically:

$$p_b = \frac{1}{1 + e^{\beta(m_y - m_b)}}$$

where m_y and m_b are the estimates of nectar reward for yellow and blue flowers, respectively, and β is a parameter that controls the sensitivity of the bee to the difference between these estimates.

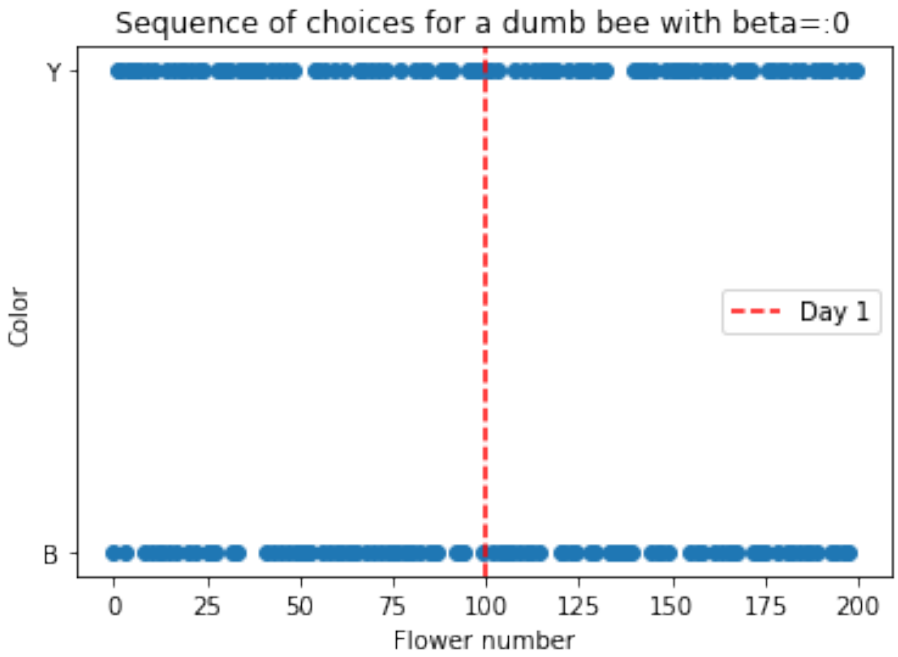
2.1 a)Exploring Bee estimate-reward behaviour with soft-max model

To explore the trade-off between exploitation and exploration, we can vary the values of $m_y - m_b$ and observe how the probability of choosing a blue flower changes. Similarly, we can fix the value of $m_y - m_b$ and vary the value of β to see how the sensitivity of the bee to reward differences affects its decision-making.



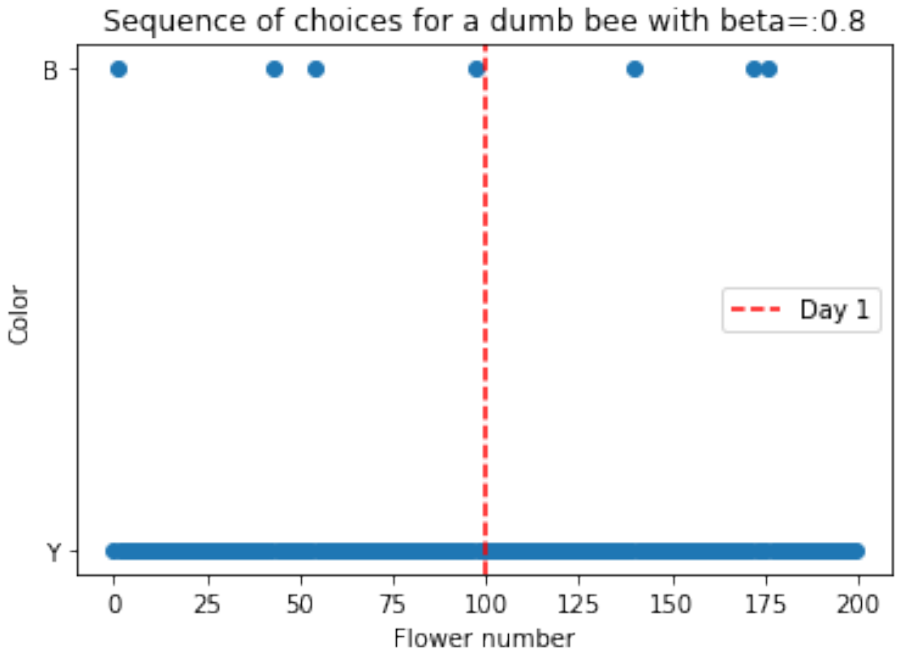
2.2 b) Dumb bee

The behavior of the dumb bee can be explained in terms of its decision-making strategy based on the internal estimates of the nectar rewards. Now let us assume that our bee is dumb and doesn't learn from experience. This means that bee will have same estimate of reward everyday and lets say m_y=0.5 and m_b=0. Also we assume that in day 1 reward of blue is 8 and yellow is 2 and in second day its vise-versa. However actual rewards are not in our interest as our bee doesn't learn from rewards.



Behaviour of dumb through 2 days, Beta=0

When Beta is set to 0, the bee will choose the flower with the higher estimated reward with a probability of 0.5 for each flower if the estimates are equal, since pb = 1 / (1 + exp(0 * $(m_y - m_b)$)) = 0.5. This means that the bee will alternate between yellow and blue flowers without any preference, and its choices will be random.



Behaviour of dumb through 2 days, Beta=0.8

But, when Beta is set to 0.8, the bee will be more inclined to take advantage of the flower with the greater expected reward. The likelihood of selecting the flower with the bigger reward likewise rises as the disparity between the predicted rewards widens. This indicates that the bee's decisions will be biased toward exploitation and that it will typically select the bloom with the highest anticipated reward. The bee may, however, continue to forage by occasionally selecting a bloom with a lower likelihood.

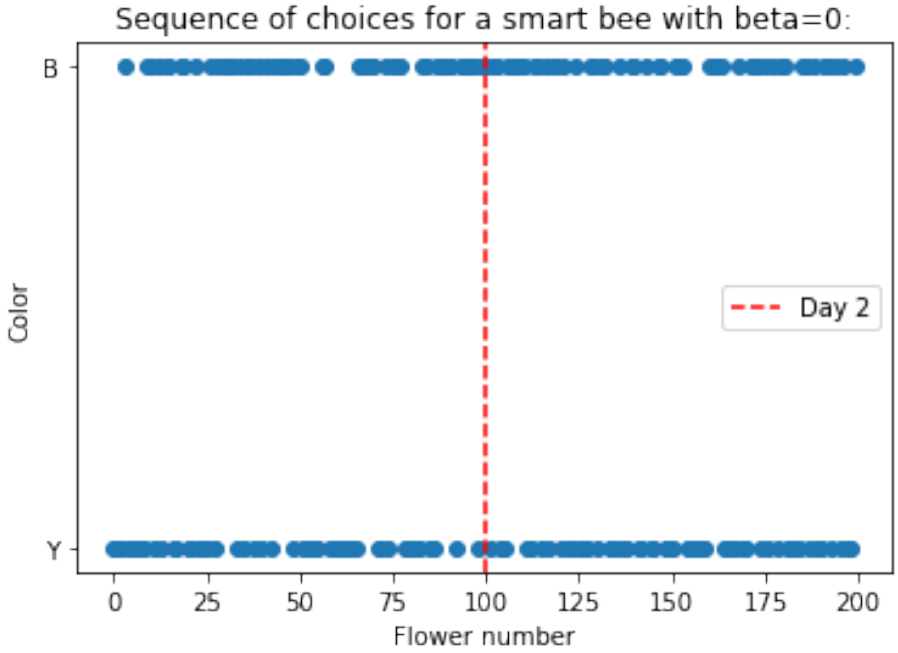
2.3 C) Smart Bee

We are simulating a "smart" bee's behavior in this task, one that can gain knowledge from its encounters. Based on the reward it receives and the learning parameter e, the bee modifies its estimate of the payoff of blue or yellow flowers.

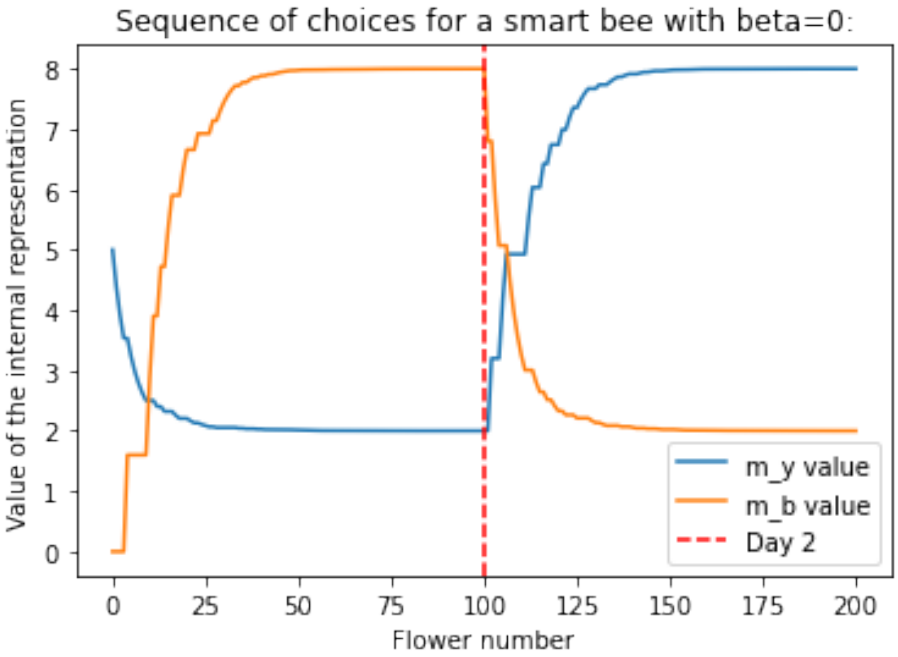
The initial flower reward presumptions are my = 5 and fireDown = 0, which serve as our starting point. The bee will select blue flowers on the first day with a probability set by the softmax function with Beta = 0 or Beta = 1, and yellow flowers on the second day with the same Beta values. The bee adjusts its reward estimate based on the online update rule after each flower visit.

We can track the changes in the reward estimates over time to see how they converge to the true rewards. We can also observe how the bee's behaviour changes as it gains experience and updates its reward estimates. For purely explorative behavior with Beta = 0, the bee's choices are equally likely between

blue and yellow flowers, and the reward estimates converge to the true rewards over time as the bee samples more flowers.

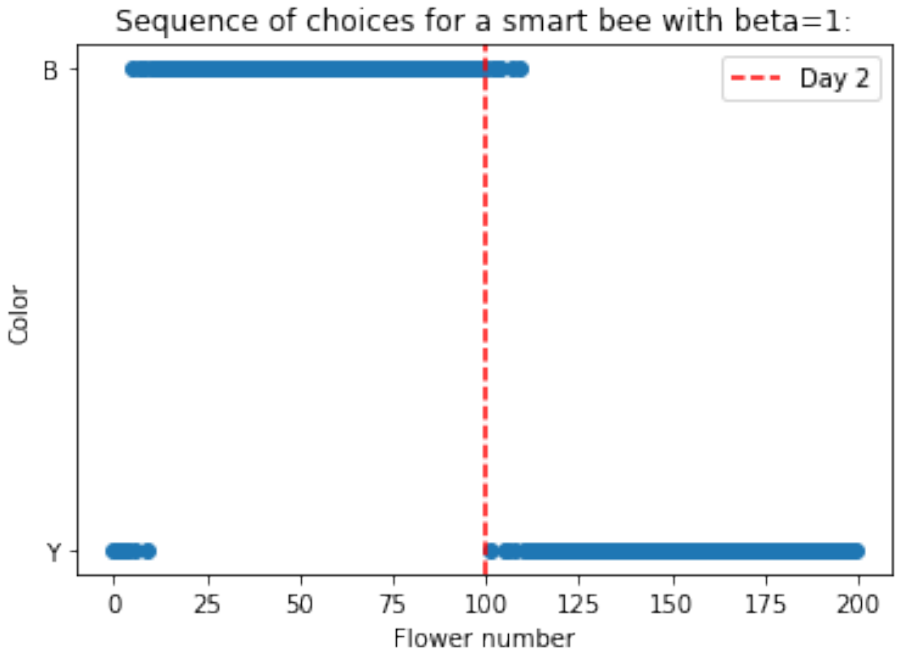


In purely explorative situation, smart bee flower choices, Beta=0

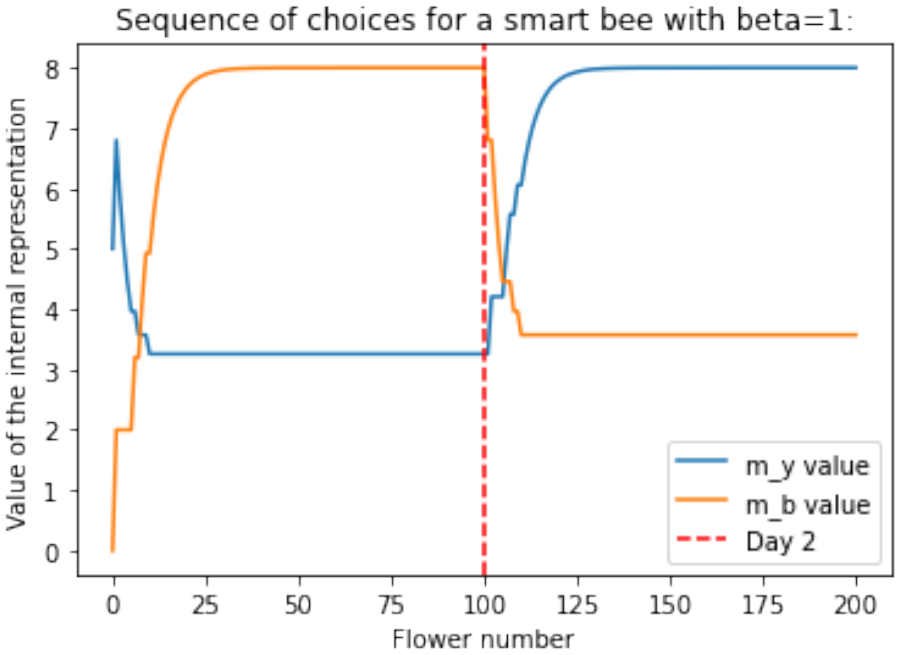


In purely explorative situation, smart bee internal representation of flowers, Beta=0

For purely exploitative behavior with Beta = 1, the bee always chooses the flower with the higher estimated reward, regardless of its actual reward. If the bee initially overestimates the reward of a certain bloom, it may end up being forced to make less-than-ideal decisions. Depending on the bee's starting estimations and the flower reward probabilities, the reward estimates in this scenario might not converge to the genuine rewards as quickly or at all.



In purely exploitative situation, smart bee flower choices, Beta=1



In purely exploitative situation, smart bee internal representation of flowers,

Beta=1

3 Problem 3: The drift diffusion model of decision-making.

The drift diffusion model depicts decision-making by integrating noisy sensory input over time until a decision threshold is achieved. We will replicate a 2AFC-task in which the subject must identify whether a visual motion stimulus is traveling upwards or downwards in this problem. The drift-diffusion model will be used to illustrate how the subject integrates noisy sensory input to reach a choice. The Euler method, a numerical method for solving ordinary differential equations, will be used to simulate the drift-diffusion model. To imitate the noisiness of actual neurons, we will additionally include a Gaussian white noise component.

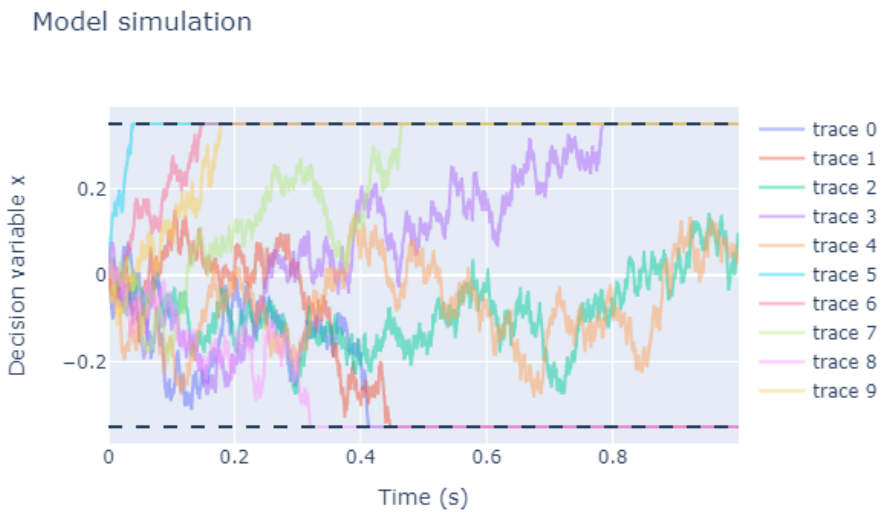
$$x(t + \Delta t) = x(t) + (m_A - m_B)\,\Delta t + \sigma\eta(t)\sqrt{\Delta t}$$

, where x(t) is the integration variable, fireUp and fireDown are the firing rates of the upward-motion and downward-motion sensitive neurons, respectively, sigma is the noise level, $\eta(t)$ is a Gaussian white noise term, and Δt is the time step.

3.1 a) Plot basic runs of DDM model assuming mA=1 and mB=0.95

We are requested to plot many iterations of the drift-diffusion model with varying initial circumstances, using mA = 1 and mB = 0.95. We are also given the time step Dt=0.1ms, the noise level sigma=$0.5ms^{-1/2}$, and are instructed to execute the Euler technique for 10000 time steps until we reach the time t=1s. We must select a suitable value for the threshold Mu.

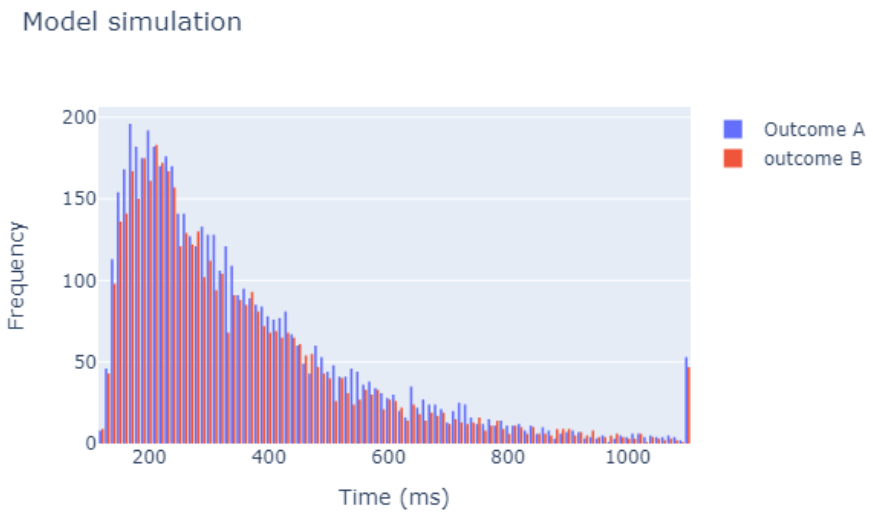
For selecting the threshold we need to have explore the data a bit to not to be ver conservative or liberate about our decision. So when we did this we found that 0.25 was a reasonably good number for the threshold.



Drift Diffusion Model Simulation, mA=1, mB=0.95 and Mu=0.25. Traces are number of trials.

3.2 b) Outcome and time thresholds

When we run the model 1000 times we can see more clearly the decision values and their occurrence frequencies. Before doing that, by just looking at firing rates, we can see that as mA is slightly greater then mB, giving a final decision on A(up) would occur more in out plot.



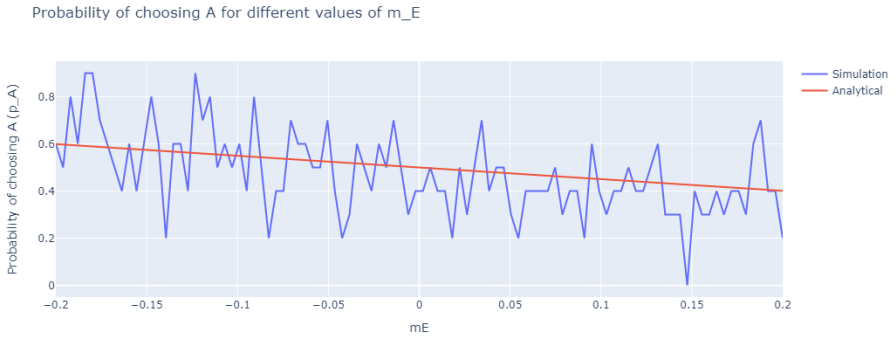
Decision and reaction times

Here you can also see this pattern in which A decision is given more frequently and also this decision is given a bit faster.

3.3 c) Comparison of analytical and simulation results

We will denote the evidence for outcome A versus outcome B as $m_E = m_A - m_B$. Plot the probability of outcome A for values of m_E ranging from -0.2 to 0.2 . Compare the results of your simulation with the analytical formula, which states that the probability of choosing B is given by

$$p_b = \frac{1}{1 + \exp\left(\beta\left(m_A - m_B\right)\right)}$$



Comparison of analytical and simulation results

After simulating the model for each value of m_E, we plot the results and compare them to the analytical formula for the probability of choosing B, which is given by the logistic function with slope parameter β. We calculate the theoretical probability of choosing B (pB theory) using this formula and plot it on the same graph as the simulated probabilities (pA values). You can see that our simulation is consistent with analytical probablity model decisionsi which shows our drift diffusion model shows is calculating the results correctly.

3.4 Conclusion

In this report, we explored several reinforcement learning models and their applications in different scenarios. We started with the basic conditional learning algorithm with Rescorla-Wagner Model. We then moved on to the bee-reward problem, where we examined the behavior of a "dumb" bee that has no learning ability and a "smart" bee that can learn from its experiences. Through simulations, we observed that the smart bee can learn to associate blue or yellow flowers with higher rewards and adjust its behavior accordingly. In the drift-diffusion model, we studied decision-making in a two-alternative forced choice task where a subject needs to decide between two alternative actions based on sensory evidence. Through simulations, we examined the distribution of reaction times for different outcomes and the probability of choosing one outcome over the other as a function of evidence.

In conclusion, these reinforcement learning models showcase the power of trial-and-error learning and decision-making under uncertainty. By using mathematical models and simulations, we can gain insights into the learning and decision-making processes of humans and animals, and develop algorithms that can be applied to various real-world problems. The plots and visualizations created throughout these exercises provide a useful way to understand and interpret the results of simulations, and can be helpful in communicating complex ideas to others.