Student Name: Mohammad Mohammad Beigi
Student ID: 99102189
Subject: Evidence Accumulation

Advanced Topics in Neuroscience - Dr. Ali Ghazizadeh
Assignment 7

---

part 1:

Q1 & Q2:

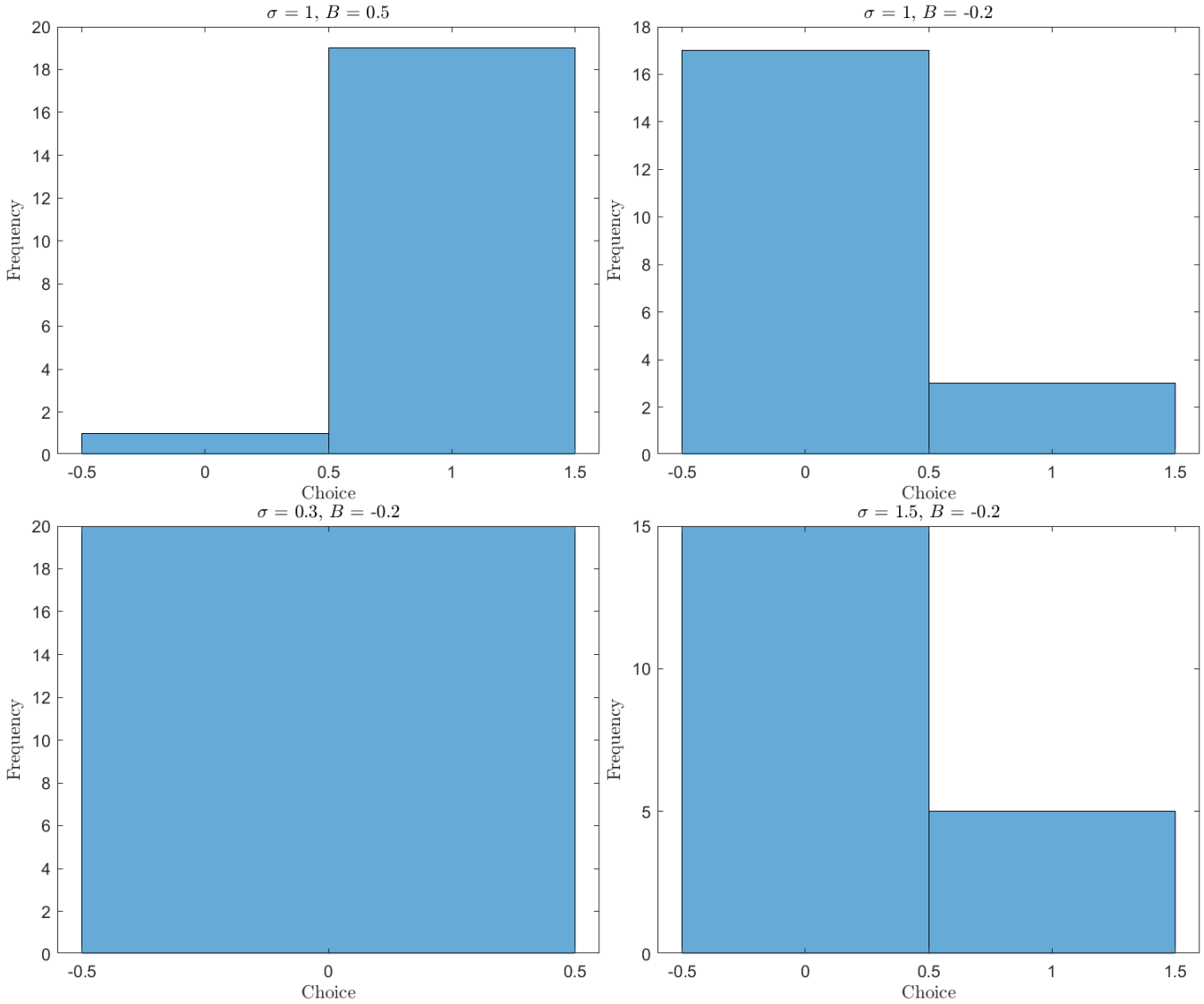
We will write a simple simulation for Go/No Go task with a fixed time interval. This would correspond to presenting a stimulus and requiring the research participant to select a choice at the end of the time interval.

The following function accepts bias term, sigma, dt and time interval as the inputs and gives the choice and decision variable as the outputs.

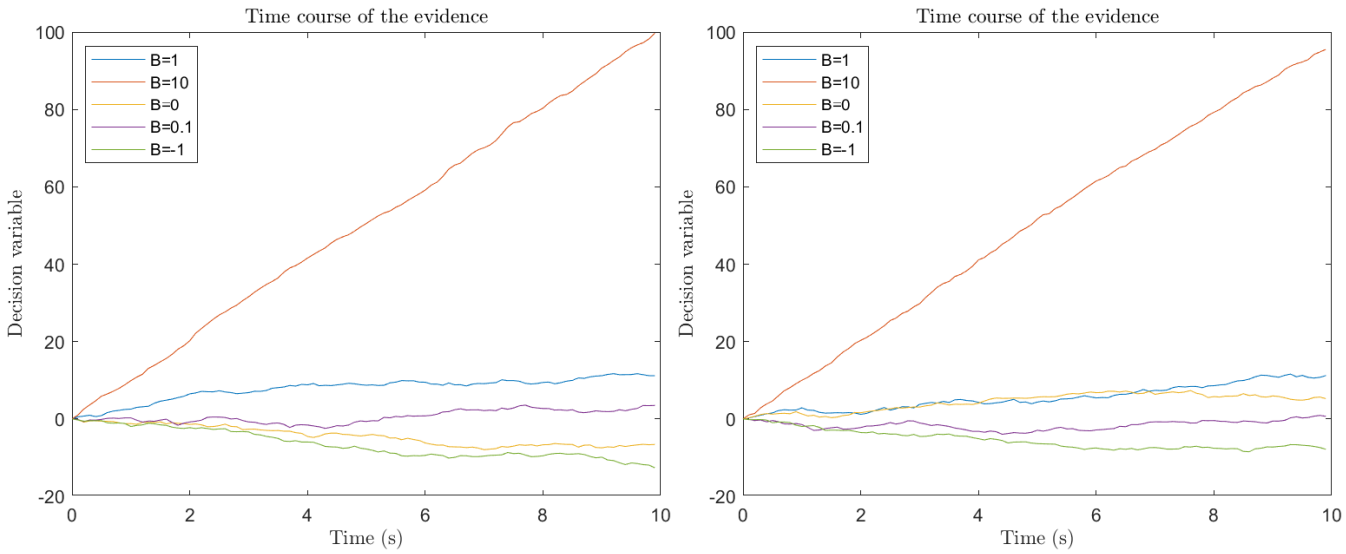
```
function [choice, output_signal] = simple_model(bias, sigma, dt, time_interval)
% Set initial values
x0 = 0; % start point
x = x0; % initialize the decision variable x
N = time_interval / dt; % number of time steps
output_signal = zeros(1, N);
for i = 1:N
    % Update the decision variable
    x = x + bias*dt + sigma*sqrt(dt)*randn();
    output_signal(i) = x;
end
% Determine the response based on the final value of the decision variable
if x > 0
    choice = 1; % Go signal
else
    choice = 0; % No Go Signal
end
end
```

We will simulate 20 choice experiments, each 1 second long, with B=1 per second, σ=1, dt=0.1 second. The results of running the simulation four times is:

```
>> Q2
Distribution of results:
    Go trials: 16 out of 20
    No-go trials: 4 out of 20
>> Q2
Distribution of results:
    Go trials: 20 out of 20
    No-go trials: 0 out of 20
>> Q2
Distribution of results:
    Go trials: 18 out of 20
    No-go trials: 2 out of 20
>> Q2
Distribution of results:
    Go trials: 17 out of 20
    No-go trials: 3 out of 20
```

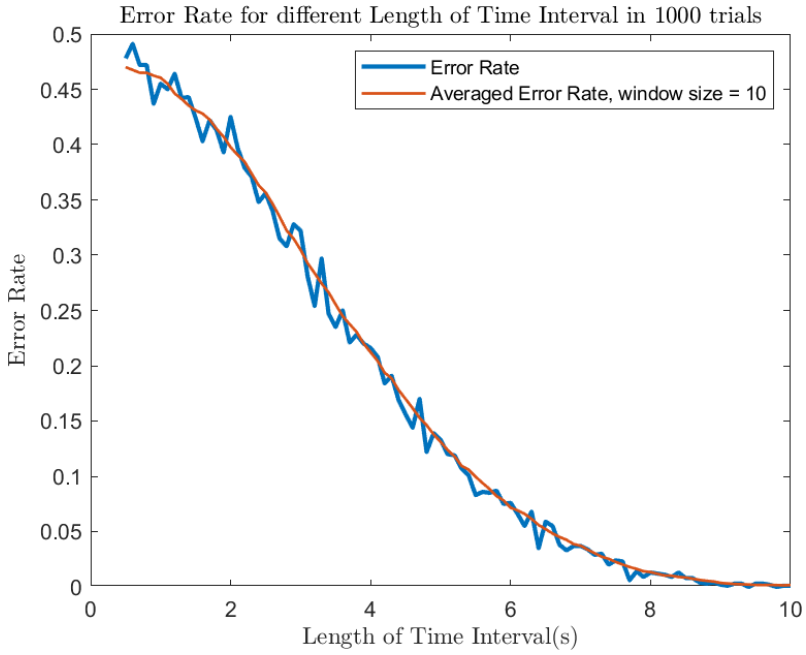


We can see that the distribution of choices obeys a binomial distribution which its parameter is specified by bias and σ. Now we will examine the time course of the evidence for different values of bias:



Q3:

To explore how the time interval influences the error rate, we will Generate 1000 trials each for tasks ranging in duration from 0.5 to 10 seconds with steps equal to 0.1. (parameters B=0.1 per second, σ=1, dt=0.1 second). We will assume that a positive response is correct:



As you see in the figure above, error rate decreases when length of time interval increases.

Q4:

To calculate the expected value and variance of the decision variable in the discrete version of the diffusion drift model (DDM), we start with the formula:

$$dX = Bdt + \sigma dW$$

Taking the expectation on both sides of this equation gives:

$$E[dX] = BE[dt] + \sigma E[dW]$$

Since $E[dW] = 0$, we have:

$$E[dX] = Bdt$$

So the expected change in the decision variable over a time step of dt is simply the bias (B) times the time step (dt).

Now, if we let $X(t)$ represent the value of the decision variable at time t, we can use the above equation to write:

$$E[X(t + dt) - X(t)] = Bdt$$

Rearranging, we get:

$$E[X(t + dt)] = E[X(t)] + Bdt$$

This tells us that the expected value of the decision variable at time t + dt is equal to the expected value of the decision variable at time t plus the expected change in the decision variable over the time step.

To find the expected value of the decision variable at a specific time t given an initial value x_0, we can apply this formula iteratively:

$$E[X(t)] = x_0 + B * t$$

Note that this assumes that the bias (B) and the standard deviation of the random walk (σ) are constant over time.

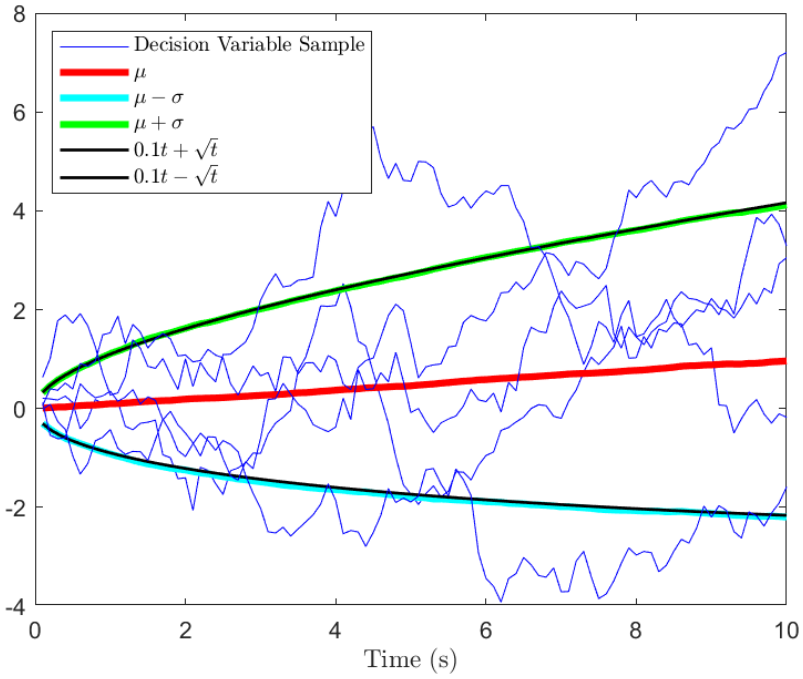
To calculate the variance of the decision variable in the DDM formula, we can use the properties of the Brownian motion term. Let's denote the variance of the decision variable as Var(X(t)). Then, using the three properties of the Brownian motion term mentioned in the question, we can derive the following:

$$Var(dX) = Var(\sigma dW) = \sigma^2 Var(dW) = \sigma^2 dt$$

where Var(dW) = dt is due to the second property of the Brownian motion term. We can then integrate this variance over time to get the total variance of the decision variable:

$$Var(X(t)) = \int_0^t Var(dX) = \int_0^t \sigma^2 dt = \sigma^2 \int_0^t dt = \sigma^2 t$$

Therefore, the variance of the decision variable in the DDM formula is proportional to the square of the standard deviation of the Brownian motion term and to the time elapsed. This means that increasing the uncertainty parameter σ or the time step size dt will increase the variance of the decision variable.



Q5:

The following function uses cdf function of normal distribution to calculate the probability of being above or below the start point without iterating through a trajectory. Once a probability of the decision variable total being above or below the start point is calculated, a draw from a uniform random distribution can be used to choose an option.

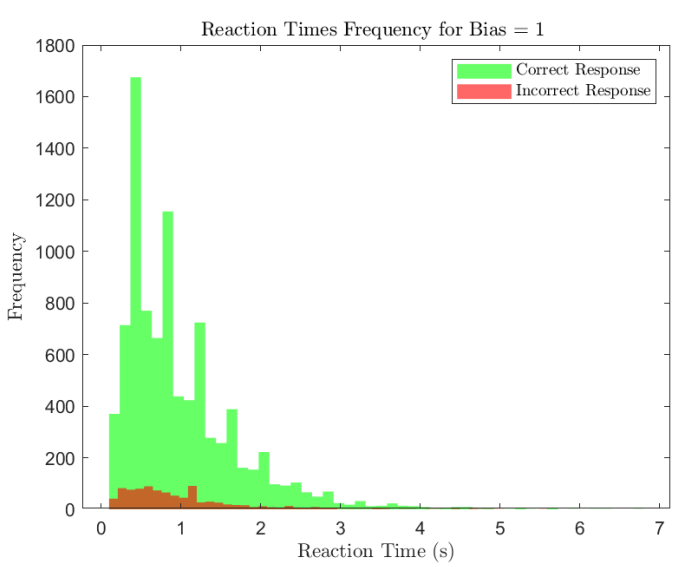
```
function choice = simple_model2(bias, time_limit, start_point)
    % Calculate the standard deviation of the Brownian motion term
    sigma = sqrt(time_limit);
    % Calculate the mean and standard deviation of the normal distribution
    mean = bias * time_limit;
    std_dev = sigma;
    % Calculate the probability of the decision variable total being above the start point
    prob_above_start_point = normcdf(start_point, mean, std_dev);
    % Draw from a uniform random distribution to choose an option
    if rand() < prob_above_start_point
        choice = 1;
    else
        choice = -1;
    end
end
```

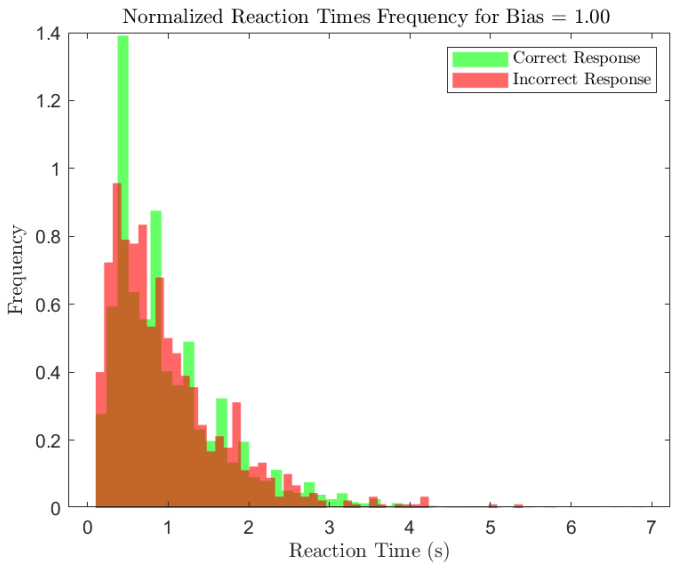
```
Distribution of results:
    Go trials: 8 out of 20
    No-go trials: 12 out of 20
>> Q5
Distribution of results:
    Go trials: 18 out of 20
    No-go trials: 2 out of 20
>> Q5
Distribution of results:
    Go trials: 14 out of 20
    No-go trials: 6 out of 20
>> Q5
Distribution of results:
    Go trials: 12 out of 20
    No-go trials: 8 out of 20
>> Q5
Distribution of results:
    Go trials: 16 out of 20
    No-go trials: 4 out of 20
```

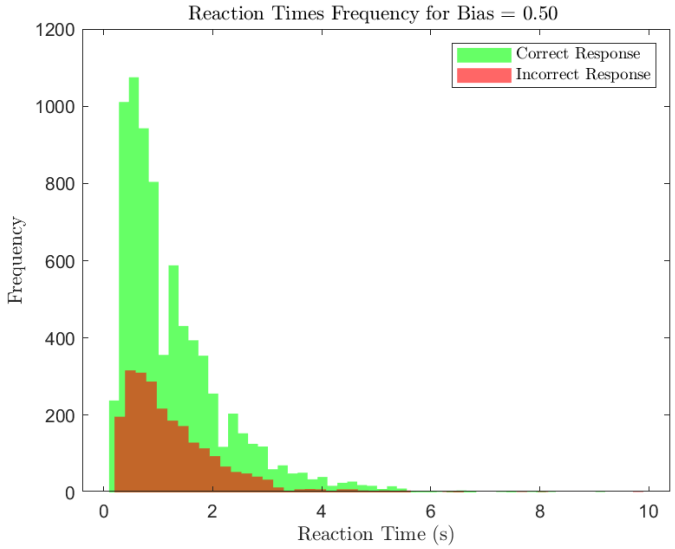
Q6:

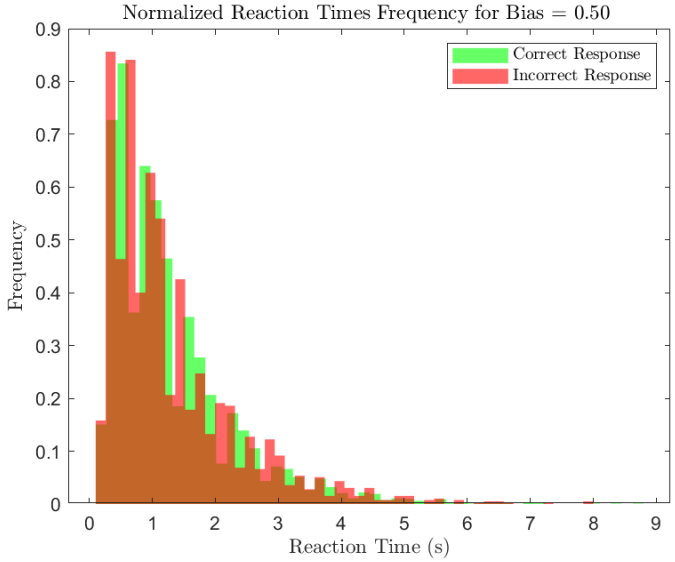
In the previous parts, the time interval was fixed. But, to have this option to calculate the reaction time, here we simulate a situation of free response. We write a function two_choice_trial that accepts five parameters: positive and negative thresholds θ_+ and θ_-, a variance for accumulation error σ, a start value x_0, and a bias B. The function should use the discrete time representation of the decision process by calculating values for the total evidence at successive time intervals until evidence exceeds a threshold. The time at the threshold crossing is the reaction time. The function should return both the reaction time and 1 or -1 for the response. We generate a large number of trials . Plot the histogram of correct (assuming that a response in the direction of the bias is correct) and incorrect response RTs.

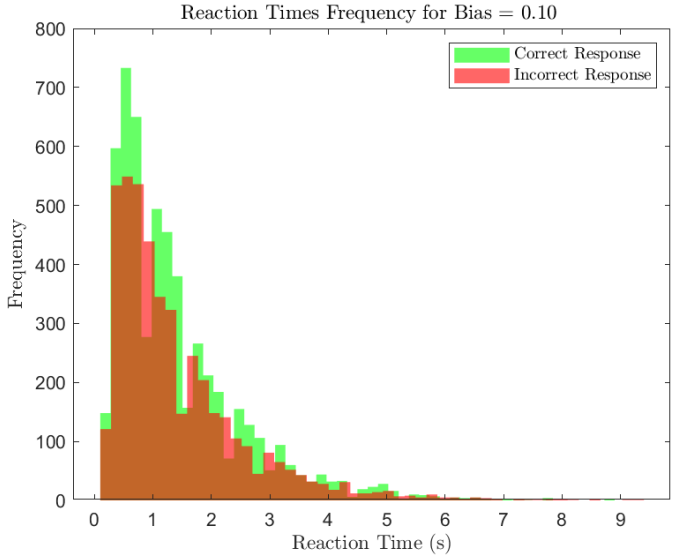
```
function [rt, response] = two_choice_trial(theta_p, theta_n, sigma, x0, bias)
    % Set the time step and initialize the evidence and time variables
    dt = 0.1;
    x = x0;
    t = 0;
    % Initialize the response variable
    response = 0;
    % Loop until the evidence reaches one of the thresholds
    while x > theta_n && x < theta_p
        % Update the evidence by adding a new sample from the random walk
        x = x + bias * dt + sigma * sqrt(dt) * randn();
        % Update the time step
        t = t + dt;
    end
    % Record the response and calculate the reaction time
    if x >= theta_p
        response = 1;
    elseif x <= theta_n
        response = -1;
    end
    rt = t;
end
```

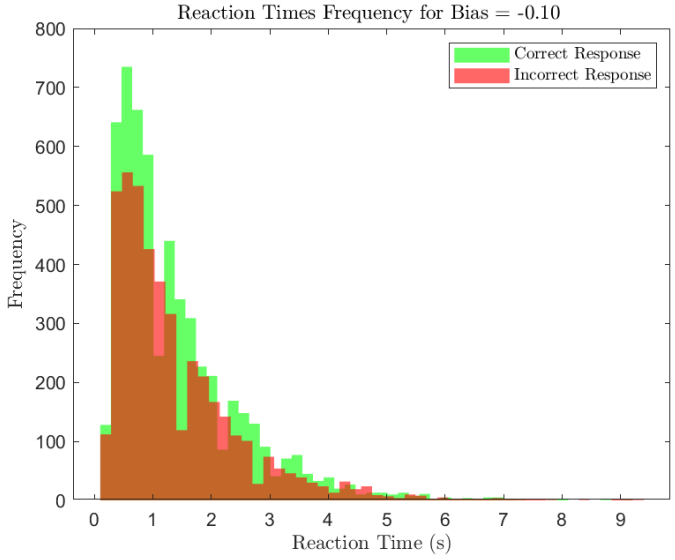

Reaction Times Frequency for Bias = 1
Correct Response
Incorrect Response
Frequency
Reaction Time (s)

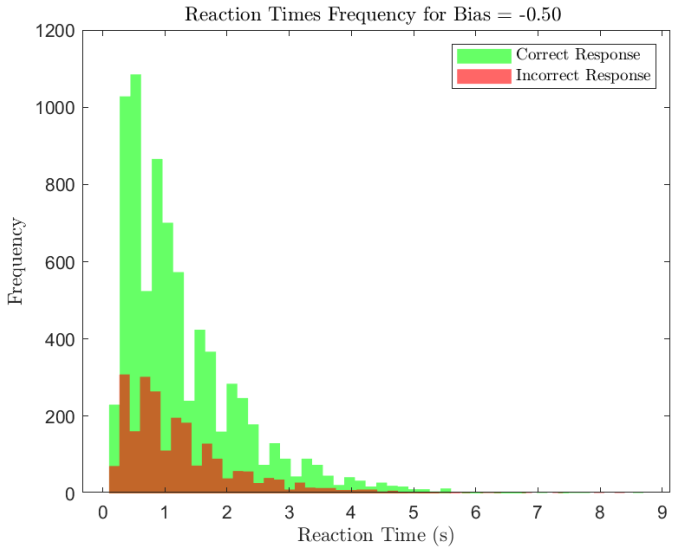

Normalized Reaction Times Frequency for Bias = 1.00
Correct Response
Incorrect Response
Frequency
Reaction Time (s)

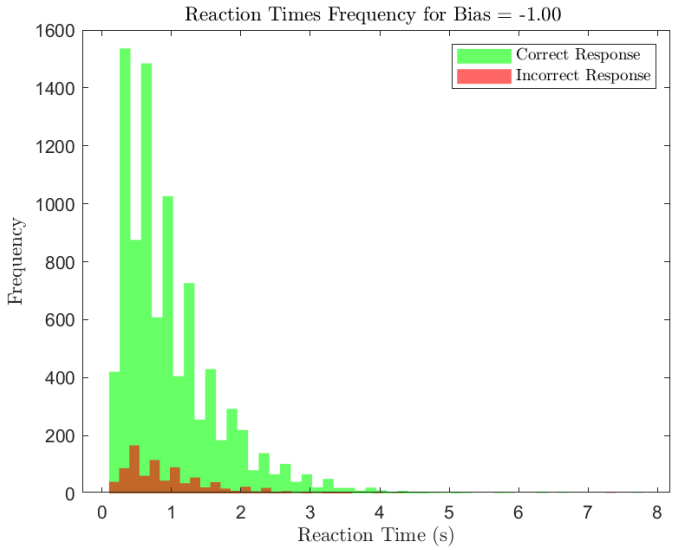

Reaction Times Frequency for Bias = 0.50
Correct Response
Incorrect Response
Frequency
Reaction Time (s)


Normalized Reaction Times Frequency for Bias = 0.50
Correct Response
Incorrect Response
Frequency
Reaction Time (s)


Reaction Times Frequency for Bias = 0.10
Correct Response
Incorrect Response
Frequency
Reaction Time (s)


Reaction Times Frequency for Bias = -0.10
Correct Response
Incorrect Response
Frequency
Reaction Time (s)

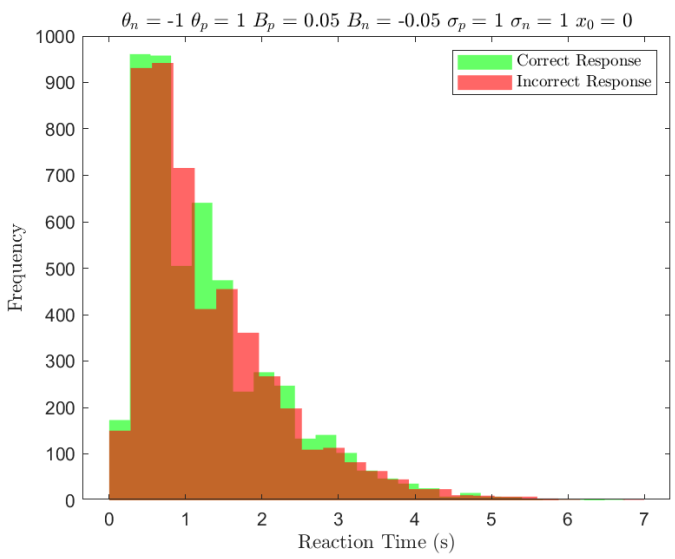


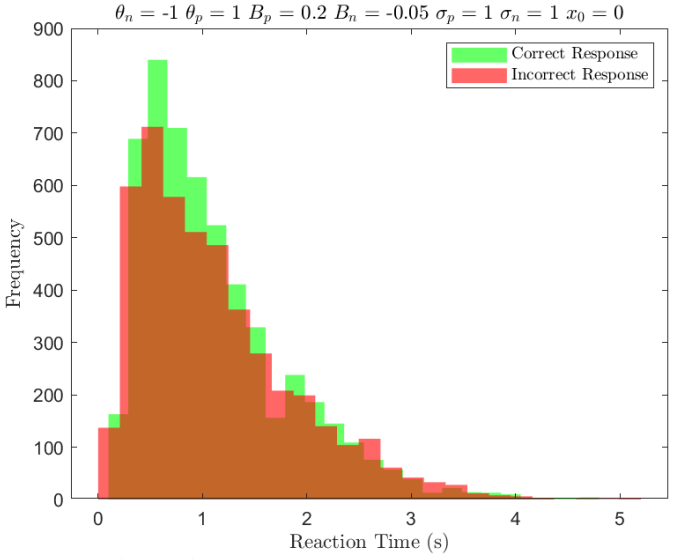


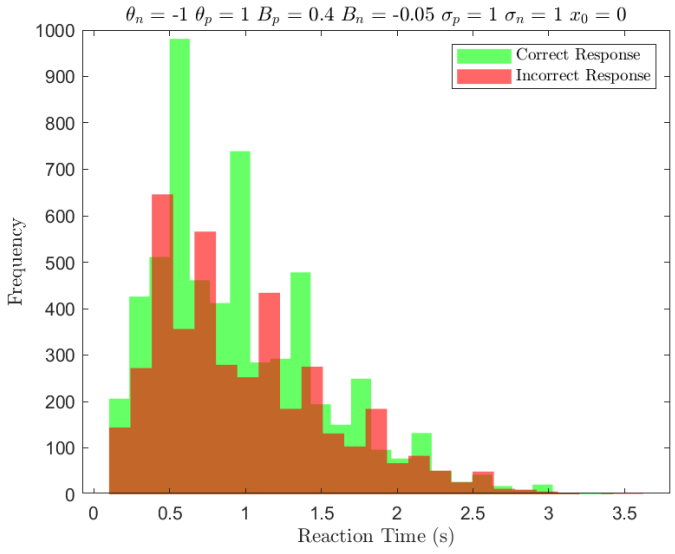
Q7:

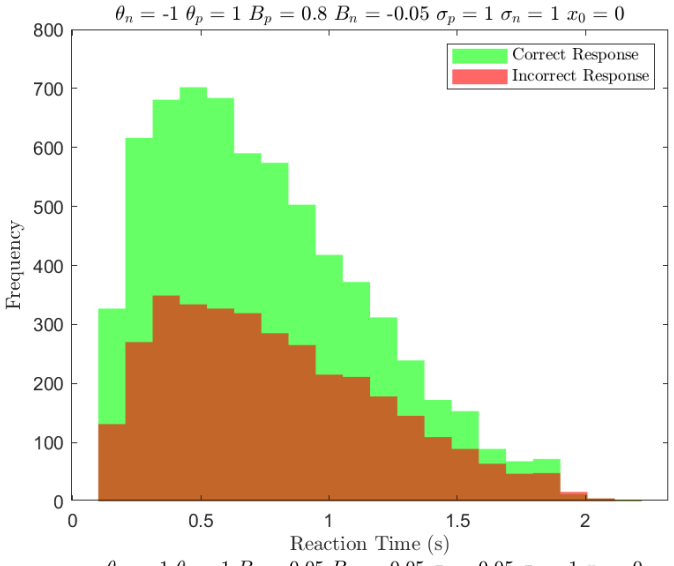
```matlab
function [rt, response] = race_trial(theta_p, theta_n, bias_p, bias_n, sigma_p, sigma_n, x0)
    % Set the time step and initialize the evidence, time, and threshold variables
    dt = 0.1;
    x_p = x0;
    x_n = x0;
    t = 0;
    % Initialize the response variable
    response = 0;
    % Loop until one of the accumulators exceeds its threshold
    while x_p < theta_p && x_n > theta_n
        % Update the evidence for each accumulator by adding a new sample from the random walk
        r = randn();
        x_p = x_p + bias_p * dt + sigma_p * sqrt(dt) * r;
        x_n = x_n + bias_n * dt + sigma_n * sqrt(dt) * r;
        % Update the time step
        t = t + dt;
    end
    % Record the response and calculate the reaction time and response direction
    if x_p >= theta_p
        response = 1;
    elseif x_n <= theta_n
        response = -1;
    end
    rt = t;
end
```

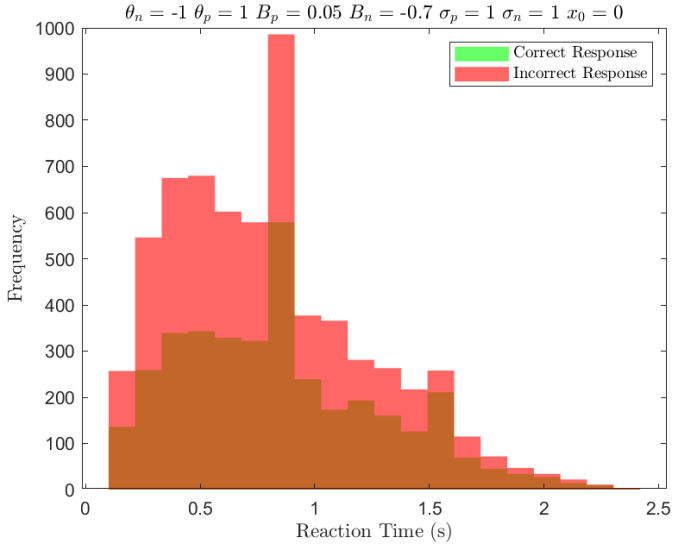
The discrete version of the diffusion drift model can accommodate a two threshold paradigm, but even with two thresholds, the single iterator allows for only a single set of bias and variance values. Under some two choice scenarios, the race model may be a better match. Under the race model, each of the two choices has an independent iterator and threshold. The first process whose accumulated evidence exceeds its threshold is the choice of the system. This choice "wins the race". We write a MATLAB code, race_trial, to simulate this race model.

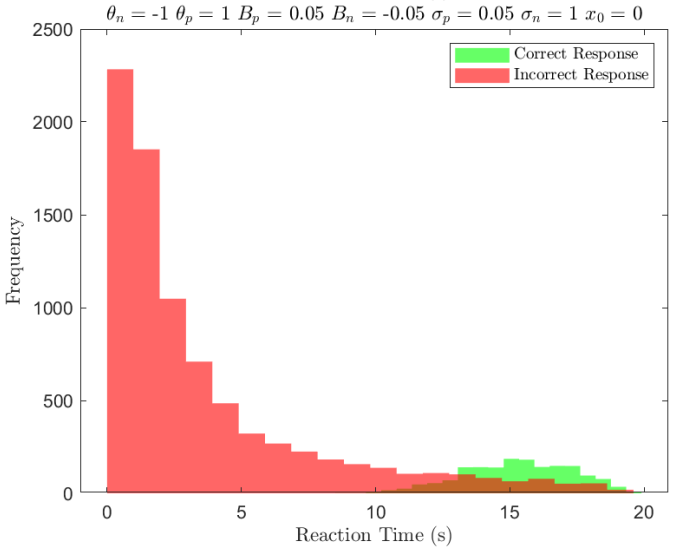

θ_n = -1 θ_p = 1 B_p = 0.05 B_n = -0.05 σ_p = 1 σ_n = 1 x_0 = 0
Correct Response
Incorrect Response
Frequency
Reaction Time (s)

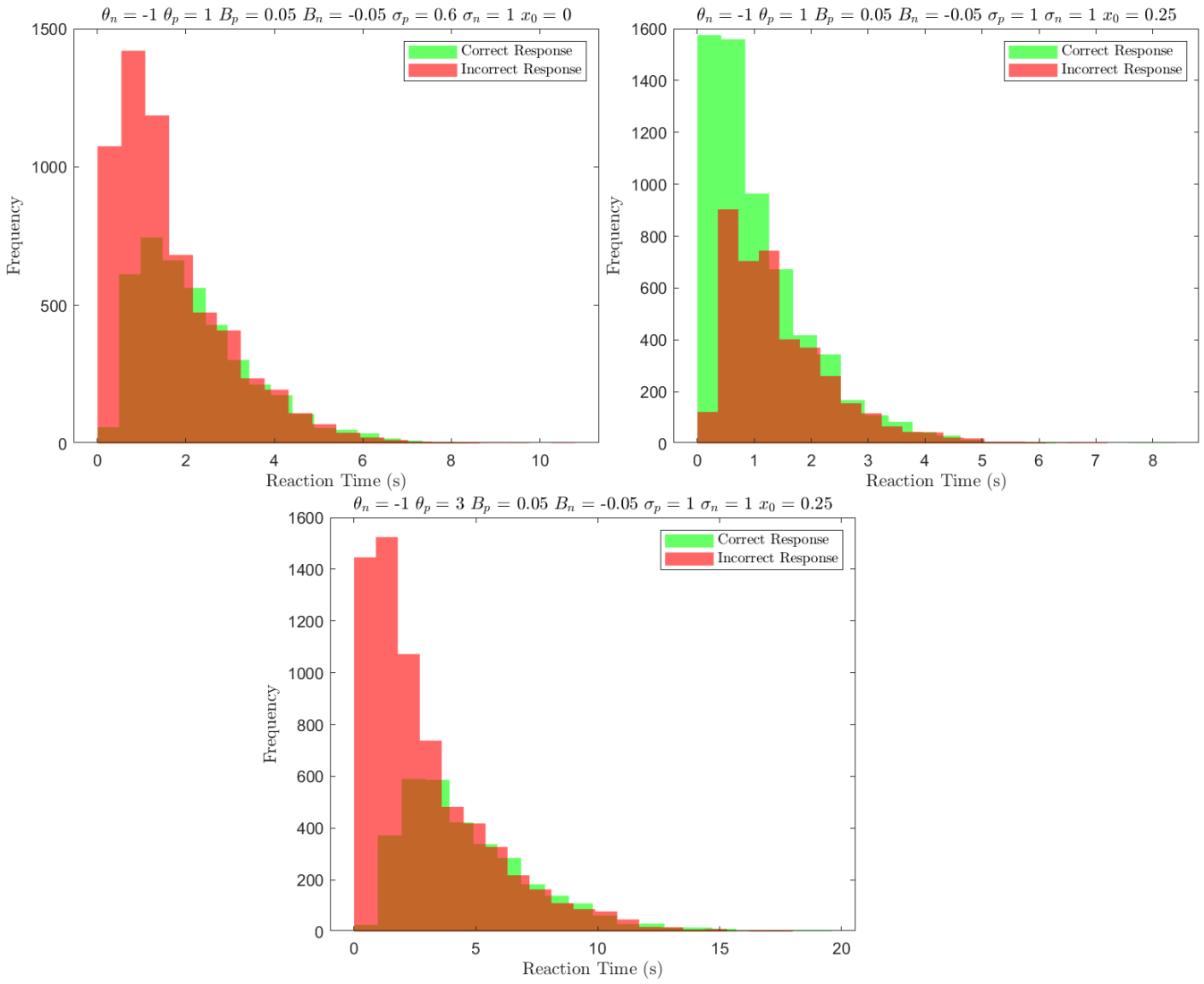

θ_n = -1 θ_p = 1 B_p = 0.2 B_n = -0.05 σ_p = 1 σ_n = 1 x_0 = 0
Correct Response
Incorrect Response
Frequency
Reaction Time (s)


θ_n = -1 θ_p = 1 B_p = 0.4 B_n = -0.05 σ_p = 1 σ_n = 1 x_0 = 0
Correct Response
Incorrect Response
Frequency
Reaction Time (s)


θ_n = -1 θ_p = 1 B_p = 0.8 B_n = -0.05 σ_p = 1 σ_n = 1 x_0 = 0
Correct Response
Incorrect Response
Frequency
Reaction Time (s)


θ_n = -1 θ_p = 1 B_p = 0.05 B_n = -0.7 σ_p = 1 σ_n = 1 x_0 = 0
Correct Response
Incorrect Response
Frequency
Reaction Time (s)


θ_n = -1 θ_p = 1 B_p = 0.05 B_n = -0.05 σ_p = 0.05 σ_n = 1 x_0 = 0
Correct Response
Incorrect Response
Frequency
Reaction Time (s)

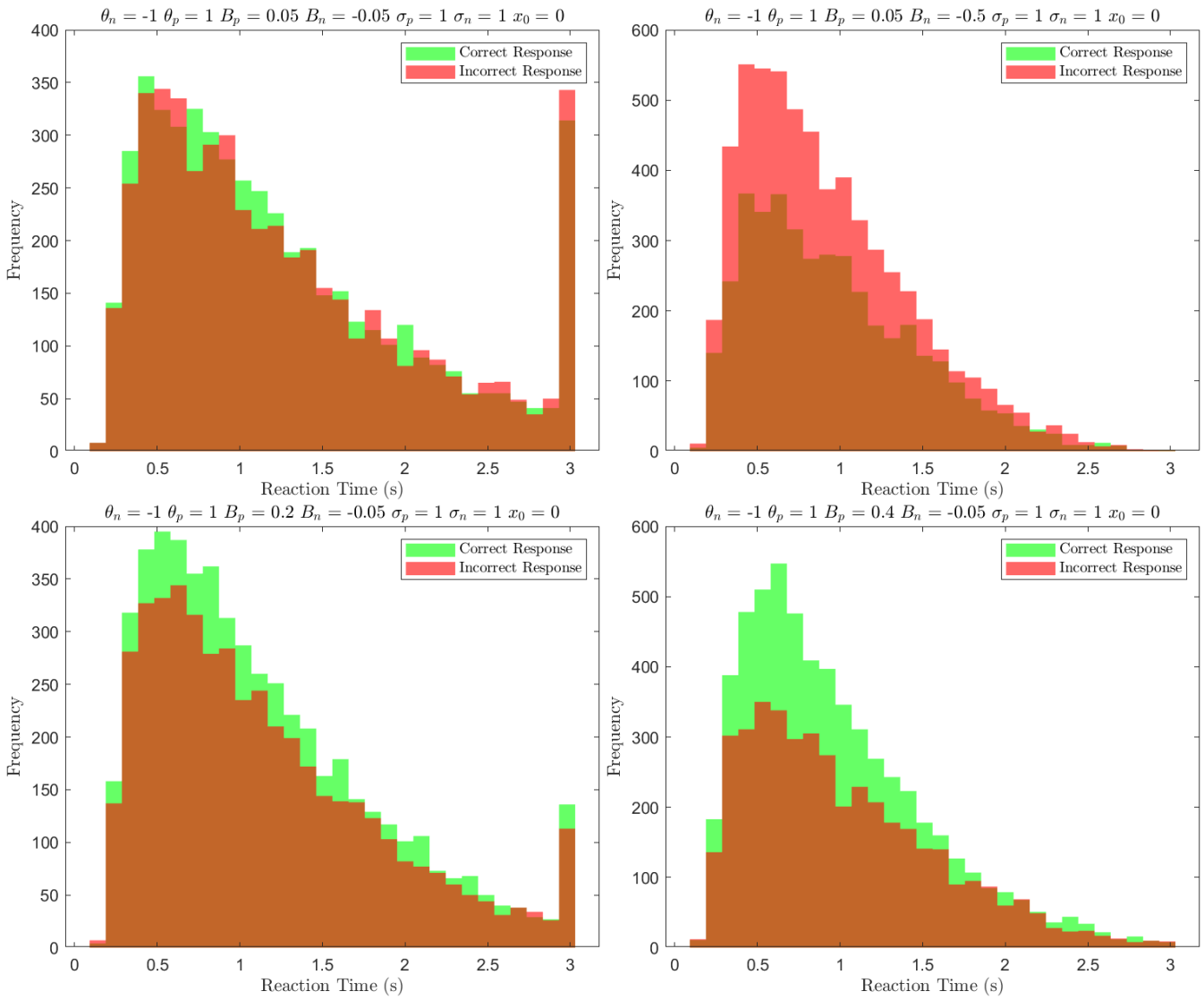


Q8:

Now we will extend race_trial to accept a fixed time interval, and to return the best choice if the total time reaches the maximum interval without a clear winner.

```
function [rt, response] = race_trial_mod(theta_p, theta_n, bias_p, bias_n, sigma_p, sigma_n,
    x0, time)
    % Set the time step and initialize the evidence, time, and threshold variables
    dt = 0.1;
    x_p = x0;
    x_n = x0;
    t = 0;
    % Initialize the response variable
    response = 0;
    % Loop until one of the accumulators exceeds its threshold
    while t < time
        % Update the evidence for each accumulator by adding a new sample from the random
    walk
        r = randn();
        x_p = x_p + bias_p * dt + sigma_p * sqrt(dt) * r;
        x_n = x_n + bias_n * dt + sigma_n * sqrt(dt) * r;
        % Update the time step
```

```
        t = t + dt;
        % Exit the loop if the thr time has been reached
        if x_p >= theta_p
        response = 1;
        break;
        elseif x_n <= theta_n
        response = -1;
        break;
        end
    end
    if(response==0)
        if(abs(x_p - theta_p)>abs(x_n - theta_n))
            response = -1;
        elseif(abs(x_p - theta_p)<=abs(x_n - theta_n))
        response = 1;
        end
    end
    % Record the response and calculate the reaction time and response direction

    rt = t;
end
```



part 2:

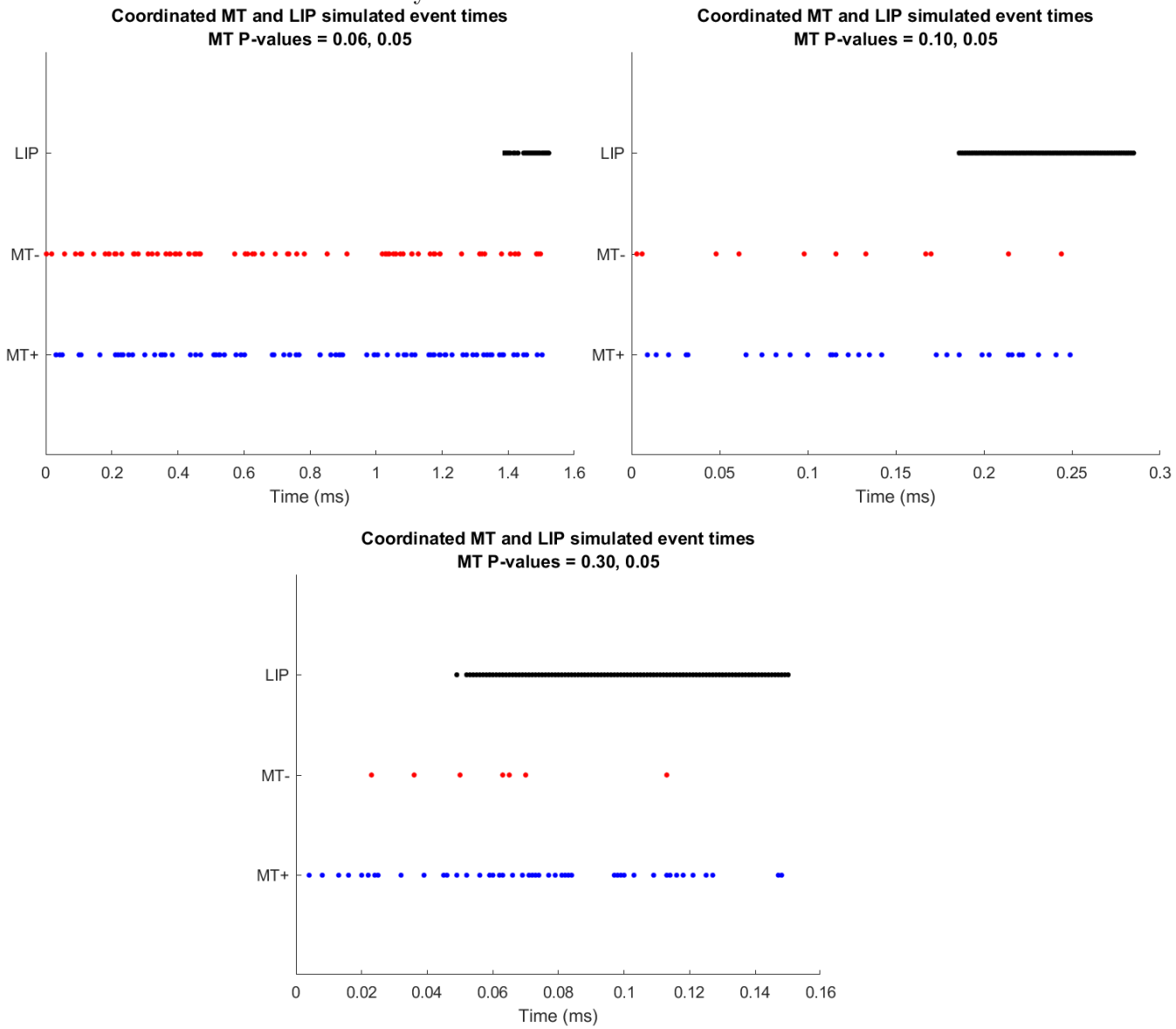
Q1:

The function LIP_activity ahead models the activity of a single LIP neuron with fixed probabilities for its corresponding excitatory and inhibitory MT neurons. Generate a sample set of coordinated MT and LIP simulated event times and examine them as a raster. How do the patterns of activity differ?

```
function [LIP_event_times, MT_event_times_plus, MT_event_times_minus] = LIP_activity(
    MT_p_values, LIP_weights, LIP_threshold, Evidence_thr)
% Parameters:
% MT_p_values - a vector with 2 elements, firing probabilities for the
% excitatory and inhibitory neurons, resp.
% LIP_weights - a length 2 vector of weighting factors for the evidence
% from the excitatory (positive) and
% inhibitory (negative) neurons
% LIP_threshold - the LIP firing rate that represents the choice threshold criterion
% Evidence_thr - threshold for evidence accumulation
% use fixed time scale of 1 ms

dt=0.001;
rate=0;
N=[0 0]; % plus is first, minus is second
t=0;
LIP_event_times=[];
MT_event_times_plus = [];
MT_event_times_minus = [];

while rate<LIP_threshold
    dN = rand(1,2) < MT_p_values;
    N = N + dN;

    % Record event times for the two MT neurons
    if dN(1) == 1
        MT_event_times_plus = [MT_event_times_plus t];
    end
    if dN(2) == 1
        MT_event_times_minus = [MT_event_times_minus t];
    end

    p_lip = sum(N.*LIP_weights);
    LIP_event = Evidence_thr < p_lip;

    if LIP_event == 1
        LIP_event_times = [LIP_event_times t];
    end

    % check LIP mean rate for last M spikes
    M = 100;
    if length(LIP_event_times)>=M
        rate = M/(t-LIP_event_times(end-M+1));
    end
    t=t+dt;
end
end
```

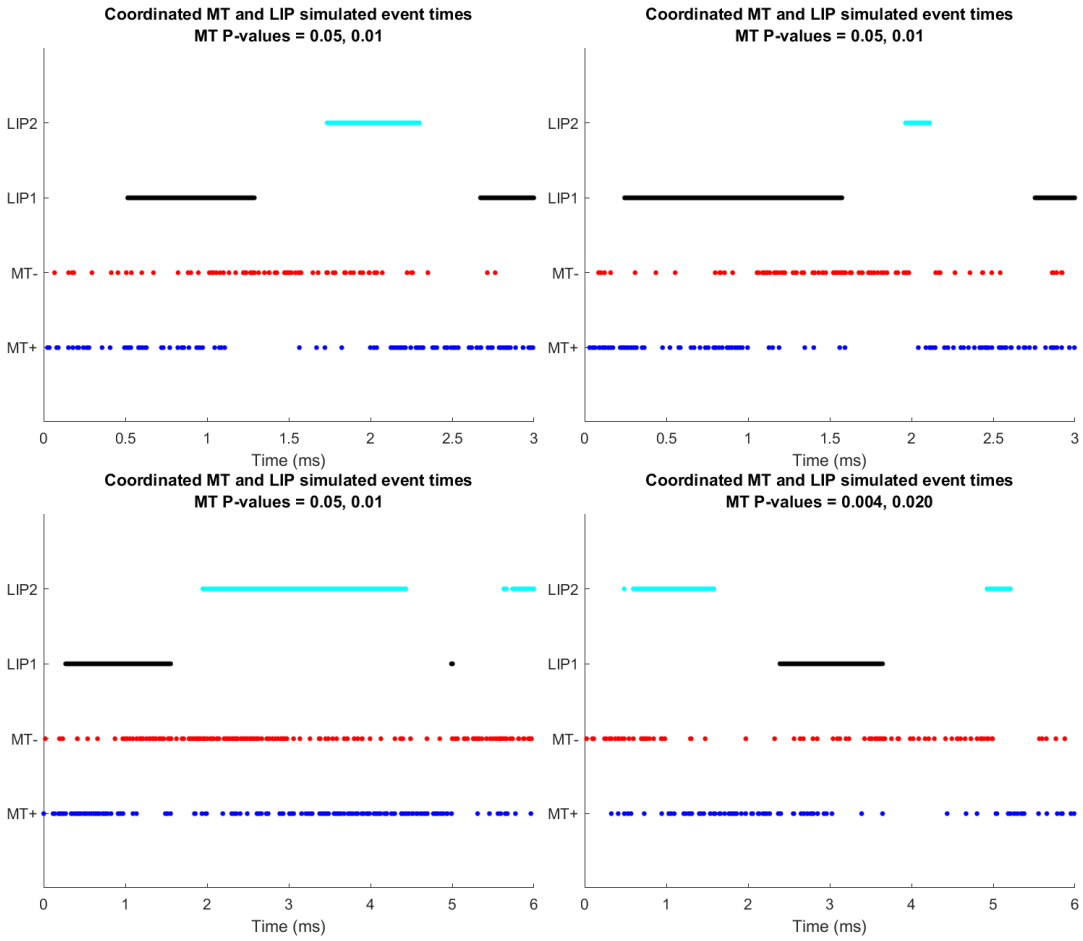
The code generates a raster plot of the event times for the two MT neurons and the LIP neuron. The blue dots represent event times for the excitatory MT neuron, the red dots represent event times for the inhibitory MT neuron, and the black dots represent event times for the LIP neuron. The plot shows the patterns of activity for the three neurons over time. We can examine the patterns to see how the activity

of the MT neurons influences the activity of the LIP neuron.



Q2:

Now we need to allow the probability of firing for the two MT neurons in the model to change with time, based on the orientation of any presented stimulus. The model should include two MT neurons and two LIP neurons. Each MT neuron should have feed forward connections to both LIP neurons, one excitatory and one inhibitory. We will generate activity patterns for both MT neurons and both LIP neurons for various stimulus presentations.


Coordinated MT and LIP simulated event times
MT P-values = 0.05, 0.01
LIP2
LIP1
MT-
MT+
0
0.5
1
1.5
2
2.5
3
Time (ms)
Coordinated MT and LIP simulated event times
MT P-values = 0.05, 0.01
LIP2
LIP1
MT-
MT+
0
0.5
1
1.5
2
2.5
3
Time (ms)
Coordinated MT and LIP simulated event times
MT P-values = 0.05, 0.01
LIP2
LIP1
MT-
MT+
0
1
2
3
4
5
6
Time (ms)
Coordinated MT and LIP simulated event times
MT P-values = 0.004, 0.020
LIP2
LIP1
MT-
MT+
0
1
2
3
4
5
6
Time (ms)