Experiment 22 – Monte Carlo Simulation

Department of Electrical Engineering and Electronics

November 2022

Abstract

Monte Carlo method was adopted to get the probability of score and the simulation was carried out via MATLAB. The probability of scoring is obtained by making N attempts and repeating them R times, and compare the experimental probabilities with the theoretical probabilities. Experiment has shown that the optimal solution for the goalkeeper is to choose position one for defence and the optimal solution for the penalty taker is to choose to kick the ball towards the top left and top right corners

Declaration

I confirm that I have read and understood the University's definitions of plagiarism and collusion from the Code of Practice on Assessment. I confirm that I have neither committed plagiarism in the completion of this work nor have I colluded with any other party in the preparation and production of this work. The work presented here is my own and in my own words except where I have clearly indicated and acknowledged that I have quoted or used figures from published or unpublished sources (including the web). I understand the consequences

Content

1 Introduction

This experiment focuses on simulating a Monte Carlo method using MATLAB to get the probability of a certain random event. It is required to simulate the distribution of penalty kicks and goalkeepers through MATLAB programming. The probability of scoring is obtained by making N attempts and repeating them R times, and compare the experimental probabilities with the theoretical probabilities. Furthermore, the advantages and disadvantages of the Monte Carlo method are analysed. Also, the optimisation of the model and the best strategy for the goalkeeper or penalty taker are obtained.

1.1 Objectives

1. To develop, explore and test Monte Carlo techniques in simulating and finding solutions to real-life random process [1].
2. To review MATLAB skills.

1.2 Theoretical Background

Monte Carlo methods are a large class of stochastic algorithms that estimate true values from random samples. It approximates the final result by constant sampling and thus constant calculation. In statistics, the quantities of interest are the distributions of estimators and test statistics, the size of a test statistic under the null hypothesis, or the power of a test statistic under some specified alternative hypothesis [1].

2 Materials and Methods

2.1 Apparatus list

MATLAB (R2022B)

2.2 Procedure

The steps of the Monte Carlo method can be briefly summarised as follows: determine the random variable, construct a probability distribution model of the random variable, draw random numbers in the probability distribution model, transform the random numbers into sample values of the random variable, calculate the probability based on this set of data, repeat the experiment R times and calculate the empirical distribution.

1.2.1 Part I: No Goalkeeper Tests

1. Calculate the probability that a ball uniformly distributed within a rectangle will land in a circle.
2. Subject to the balls satisfying the uniform distribution, design the program so that N random penalty shots can be entered and the experiment repeated R times.
3. Fix R=5 and take N equals to 100, 1000, 10000, 100000 respectively to get the probability.
4. Fix N=1000 and take R equals to 5, 10, 15, 20 respectively to get the probability.
5. Repeat the procedure with the ball satisfying the normal distribution.

2.2.2 Part II: With Goalkeeper Tests

1. The five posture choices of the goalkeeper and the position of the ball all fit a uniform distribution to obtain the probability when N equals 100 and 1000.
2. Repeat step 1 of part2, at which point the ball conforms to a normal distribution.
3. When statistically 90% of the goalkeepers tend to jump to the lower two corners of the goal, and balls satisfy uniform distribution, then get the probabilities when N equals to 100 and 1000 respectively. When balls satisfy Gaussian distribution, repeat this step and obtain the probabilities.

3 Results and Analysis

3.1 The results of part I

Task 1

If a large number of shots are attempted (uniformly distributed), a numerical value for the fraction of balls entering the goal to the total number of balls in the circle area can be derived as below:

$$theoretical\,probability = \frac{WL}{\pi R^2} = \frac{2 \times 4}{5\pi} = 0.5093$$

Task 2

The code of the program is shown in Appendix-Part I-Task 2.

Task 3 (uniform random number generator)

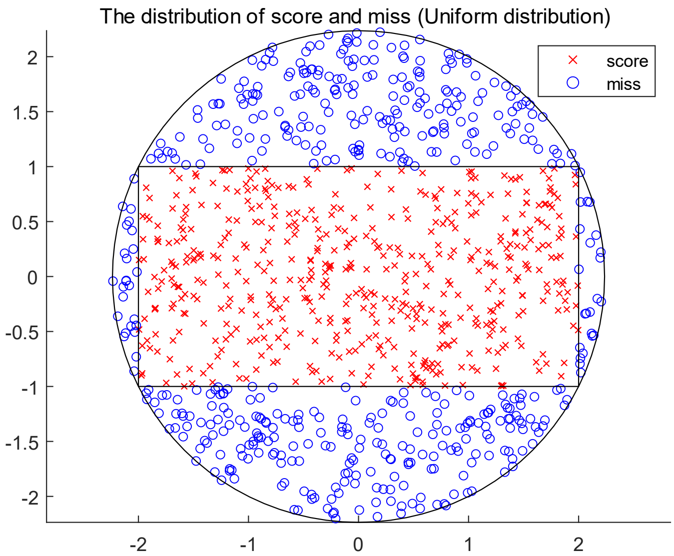


Figure 1. Scatter plot for N=1000, R=1

The probability in this situation is 0.5010.

Task 4

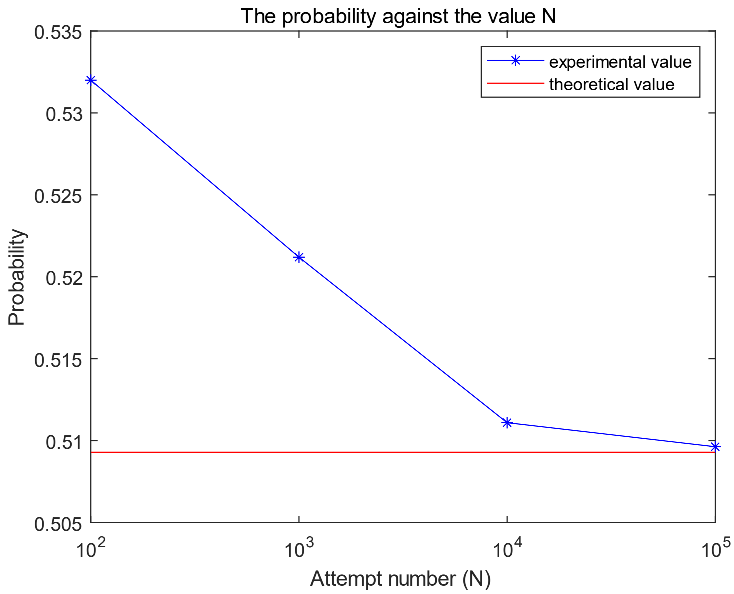


Figure 2. Probability against N and comparison

The shape of the line graph shown in task3 is decreasing and gradually converge to the theoretical probability. As the number of attempts increases (N), and with R fixed, the probability gradually approaches the theoretical probability value of 0.5093.

Task 5

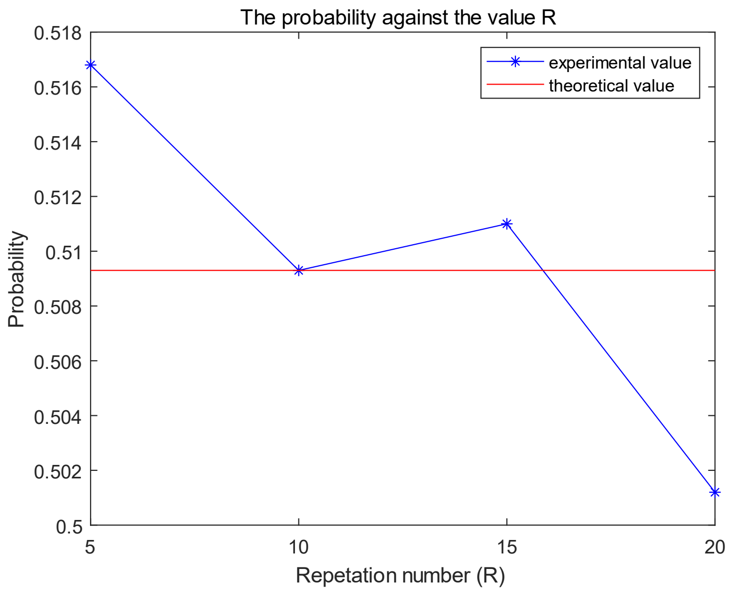


Figure 3. Probability against R and comparison

There is no obvious pattern to the shape of the line chart in Figure 3, other than to say that all four values float around the theoretical values. The reason for this is that the number of repetitions does not vary much and is relatively small, so no clear pattern can be seen. However, if the value of R is large enough, an experimental value that approximates the theoretical value can be obtained. Here, the value of N is kept constant and the value of R is changed, mainly to investigate the effect of the mean operation on reducing the error.

Task 6

In the case of task4, R is fixed and the size of N grows rapidly. In the case of task5, N is fixed at 1000 and R grows in size at intervals of 5. Compared to Figure 3, the experimental probability values in Figure 2 show greater variability and a more pronounced tendency to 'converge'. The explanation is as follows: when using the Monte Carlo method to obtain experimental probability values, the experimental probability values will gradually approach the theoretical values as N increases; the smaller N is, the more pronounced the randomness and therefore the greater the possible error. If N is fixed and the number of repetitions R is gradually increased, reducing the random error by the operation of averaging, the variation and regularity of the experimental values cannot be observed with small variations in R. However, when R is sufficiently large, the experimental probability values obtained will also approximate the theoretical probability values.

Task 7 (Gaussian random number generator)

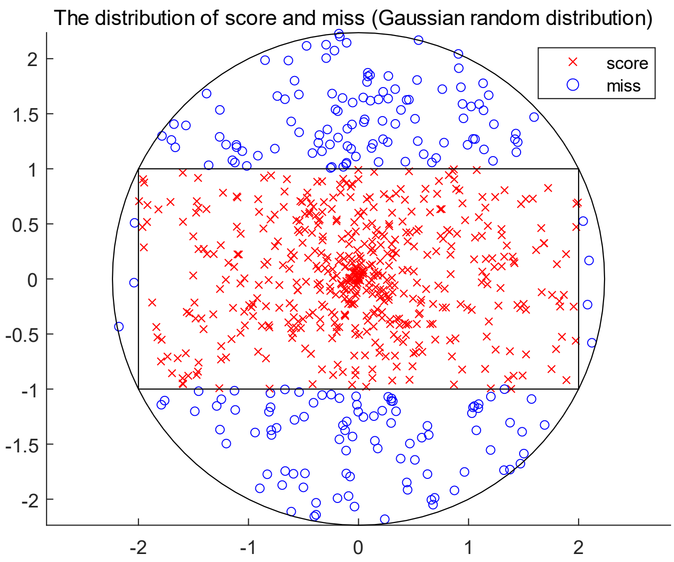


Figure 4. Scatter plot for N=1000, R=1 (balls out of the circle is cancelled)

When N=1000, the probability is 0.7346.

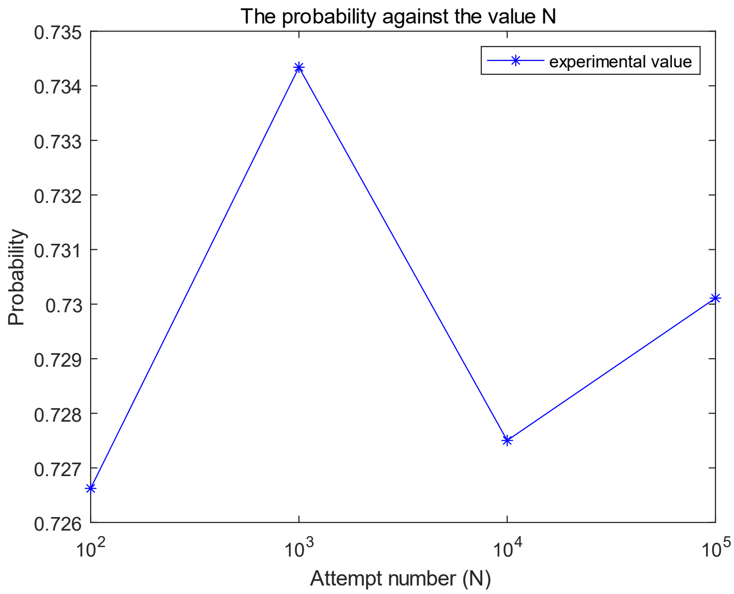


Figure 5. Probability against N and comparison

When N=10^6, the probability is 0.7298.

As shown in Figure 5, in the case of a normal distribution and with fixed R, although there is no numerical decreasing trend in the probability values obtained experimentally, the difference between the experimental and the theoretical probability values gradually becomes smaller as the number of sample size (N) increases. That is, as the sample size N increases, the experimental probability value will gradually approach the theoretical value.

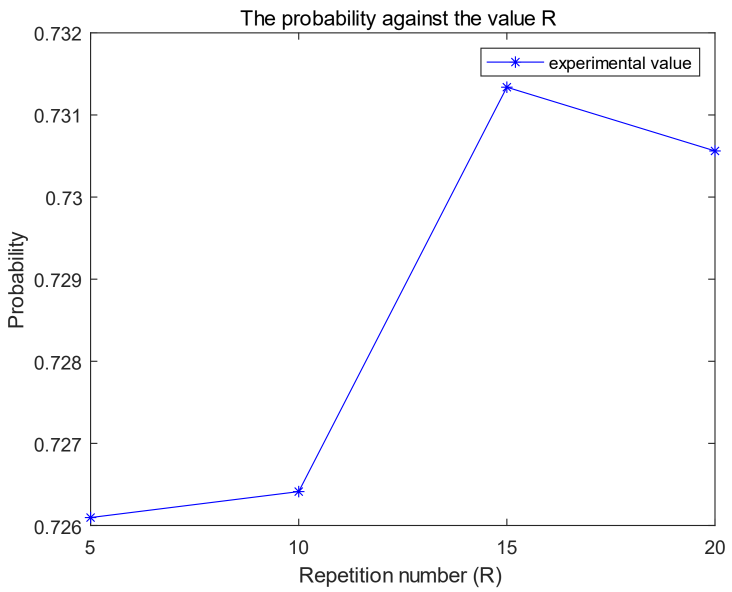


Figure 6. Probability against R and comparison

As shown in Figure 6, in the case of a normal distribution and with fixed N, no clear trend can be generalised from the chart, which can only be described as all four values oscillating around their theoretical values.

In the case of Gaussian distribution, the explanation in Task 6 can also be used to illustrate the difference between the probability plot with fixed R but changing N and the probability graph with fixed N but changing R.

Comment on the difference between Gaussian distribution case and uniform distribution case:

When N is large enough, that is, N=10^6, the probability of score under Gaussian distribution as 0.7298 and that under uniform distribution is 0.5091. The probability of a goal is approximately 22 % higher in the case of a Gaussian distribution than in the case of a uniform distribution. The reasons are as follows: when calculating the probability of goal in the case Gaussian distribution, the balls outside the circle are cancelled and not included in the calculation. In this experiment, the Gaussian distribution has $\mu = 0$ and σ equals to the radius. The centre of the rectangle is also at the (0, 0). In the case of Gaussian distribution, most balls will converge to (0, 0) and have a tendency to aggregate. In the Gaussian distribution, μ represents the location of aggregation and σ represents the degree of dispersion. So, the probability of score (points within the rectangle) is greater in case of Gaussian distribution compared to uniform distribution.

3.2 The results of part II

Task 8

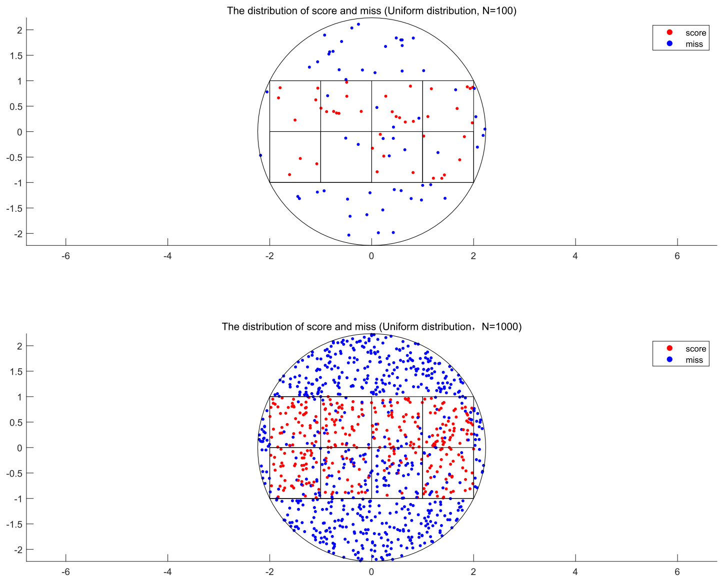


Figure 7. Goalkeeper (uniform) & balls (uniform)

Table 1. The probability with goalkeeper (uniform process) and balls with uniform distribution

N	Probability
100	0.3500
1000	0.3550

And when N=10^6, the probability is 0.3566.

Table 2. Comparison

Situation	Probability
With goalkeeper	0.3500 (N=100)
	0.3550 (N=1000)
Without goalkeeper (N=100)	0.5010
Calculation (without goalkeeper)	0.5093

As shown in Table.2, the probability of score in ‘with goalkeeper’ situation is smaller than the probability of score in ‘without goalkeeper’ situation, and the difference is about 15%. Compare Figure 1 and Figure 7, it can be observed that in Task 3, all the balls in the range of rectangle are regarded as 'score', and in Task 4, there is some 'miss' points appear in the lower half of the rectangle.

Task 9

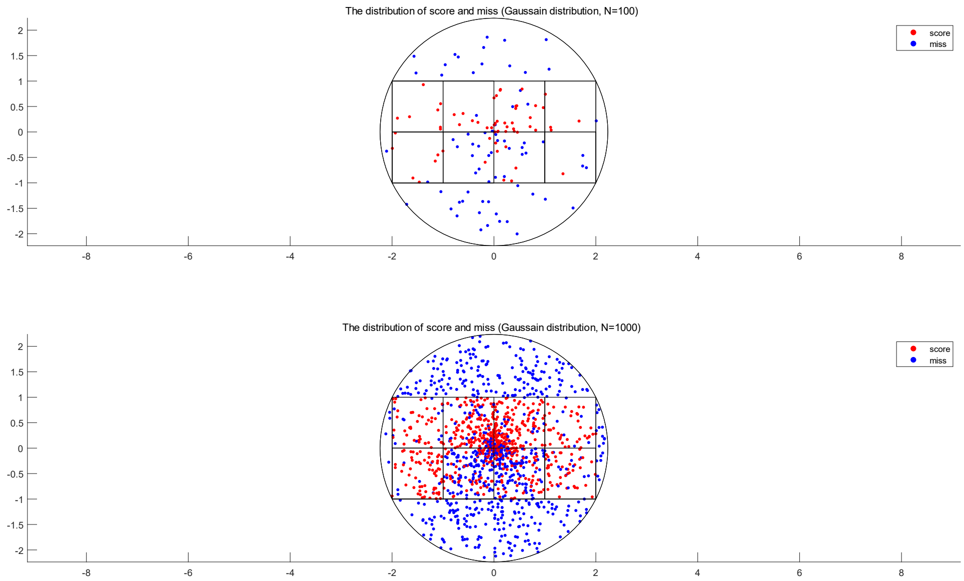


Figure 8. Goalkeeper (uniform) & balls (gaussian)

Table 3. The probability with goalkeeper (uniform process) and balls with gaussian distribution

N	Probability
100	0.4462
1000	0.4747

And when N=10^6, the probability is 0.4711.

Table 4. Comparison

Situation	Probability
Without goalkeeper (Gaussian distribution)	0.7298
With goalkeeper (Uniform distribution)	0.3566
With goalkeeper (Gaussian distribution)	0.4711

According to Table 4 (all the N is big enough and very close to theoretical value), when the balls are in Gaussian distribution and without goalkeepers, the probability of 'score' is highest among these three situations. As in Gaussian distribution, all the balls have a tendency to gather at the centre, which means more balls will be scored (the balls out of the circle are cancelled). And if balls satisfy Gaussian distribution, the probability of ‘score’ without goalkeeper is about 25% higher than that with goalkeeper. If in the situation with goalkeeper, the probability of ‘score’ under Gaussian distribution is about 10% higher than that under uniform distribution.

Task 10

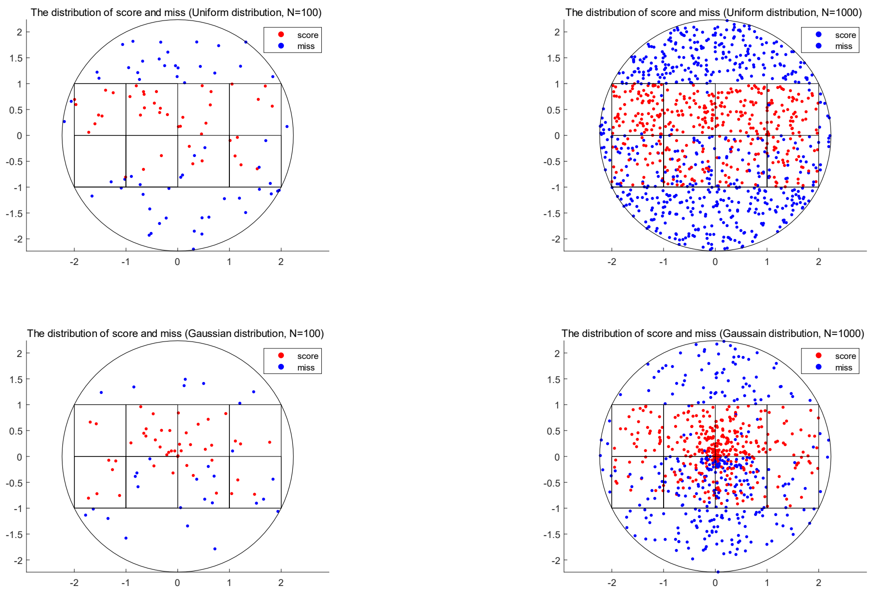


Figure 9. 90% of the time goalkeeper tends to jump to the lower two corners

Table 5. The probability with goalkeeper (90% two lower corners) and balls with uniform distribution

N	Probability
100	0.4100
1000	0.3770

Table 6. The probability with goalkeeper (90% two lower corners) and balls with gaussian distribution

N	Probability
100	0.4394
1000	0.5627

According to Table 6 and 7, the probability when balls satisfy Gaussian distribution is higher than the other. That is, when the penalty taker kicks balls under uniform distribution, it is easier to score.

4 Review Questions

Q1. In terms of what you have done in this experiment, comment on the advantage and disadvantages (or drawbacks) of the Monte Carlo experiment.

Advantages: ① Monte Carlo method can be based on the actual problem itself, rather than relying on mathematical formulas, which is more intuitive and easy to understand. In this experiment, the program can directly simulate the process of balls' kicking, instead of representing the pdf formula. ② The computation complexity of Monte Carlo method does not depend on the dimension. In this experiment, the coordinates of the ball are a two-dimensional quantity, and to simulate it at the program level, only the random number generator of two variables needs to be set. ③ Monte Carlo method is less constrained by geometry. In this experiment, the range is related to a circle and several rectangles. If calculated directly by mathematical formula, the process of integral is difficult. But in MC method, only the description of geometric characteristics of the region is required. ④ The program structure is simple and easy to implement.

Disadvantages: ① Approximate results can be calculated on random sampling. However, until the real results are obtained (by giving up random sampling and adopting deterministic methods like full sampling), it is impossible to know whether the current results are accurate or not. ② A very large value of N is required to get an accurate result. It can be known from tasks 4 and 5 that the larger the value of N, the closer the probability value is to the exact theoretical value, and at this time N has reached 10^5. ③ The errors generated by Monte Carlo method are probabilistic errors. In this penalty kick experiment, the position of the ball and the position of the goalkeeper are random, and the number of attempts is limited, so the error is also random. ④ Only the numerical solution of probability can be obtained, but the analytical solution cannot be obtained. In this experiment, all probabilities are values that can float in some numerical value, and N and R should be assigned to get the result.

Q2. Discuss the ways in which the above model could be made more accurate and realistic.

① In the situation of Gaussian distribution, the balls out of the circle should not be cancelled and it should be included in 'miss'. ② The probability when the coordinates happen to be on the edge of the rectangle should not be 0. That is, this model can be improved by taking the width of the edge into consideration. ③ In practice, the ball cannot be kicked into the ground, so the ball under the rectangle will bounce off the ground. The reflection in the MATLAB model should be that the points under the rectangle are symmetric into the rectangle through the horizontal plane (as shown in Figure 10).

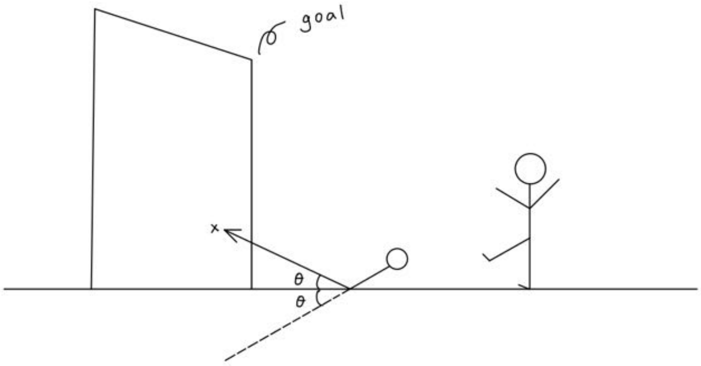
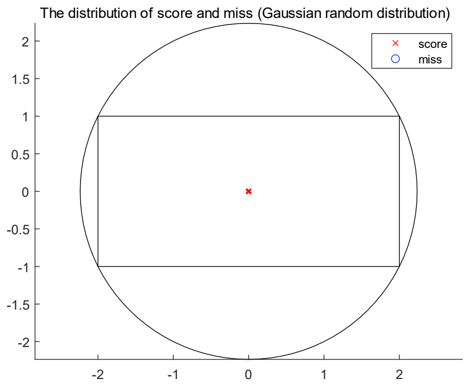


Figure 10. Schematic diagram

Q3. With reference to Task-7 and Task-9, discuss the effect of changing the standard deviation of the Gaussian distribution on both the accuracy and precision of the penalty shots.



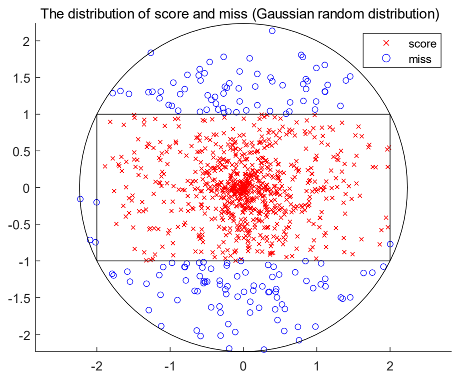


Figure 11. standard derivation is 0.001R and standard derivation is 0.5R

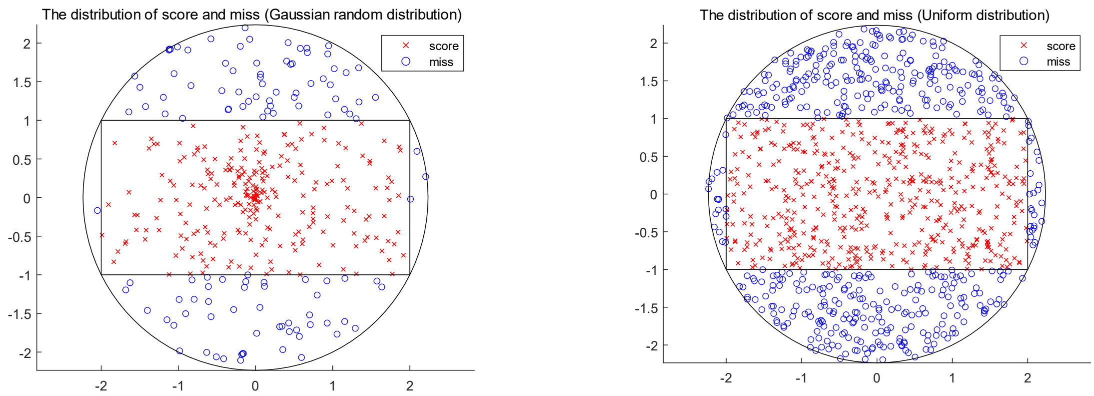


Figure 12. standard derivation is 2R and standard derivation is 100R

Table 7. The relationship between standard derivation and score probability

σ	probability
~0 (0.001R)	1
0.5R	0.8341
2R	0.6968
100R	0.6839

According to Table 5, the score probability will increase as the standard derivation getting small. Extremely, $\lim_{\sigma \to 0} probability = 1$. And according to Figure 10 and 11, when the standard deviation goes down, both the accuracy and the precision go up (the balls are more cantered and more likely to score).

Q4. If a large number of balls are kicked on the goal (i.e., if N is sufficiently large), the value of π can be estimated using (some function of) the ratio of the number of scores to the total number of the shots. Hence, find the relation that estimates the value of π. Verify this using your results for both uniform and Gaussian distribution.

When the balls satisfy uniform distribution as in Task 1, π can be estimated by:

$$\pi = \frac{8}{\sqrt{5}^2 \times p}, \qquad ①$$

$$here\ p\ represents\ score\ probability$$

When N=10^6, R=10, and apply equation① to both Gaussian distribution and uniform distribution.

Table 8. The numerical values of π in different distribution

Gaussian distribution	Uniform distribution
2.1927	3.1397

According to Table 6, the estimated value of π in Gaussian distribution is smaller than in uniform distribution. And compared to previous knowledge ($\pi \approx 3.14$), it can also be derived that uniform distribution is the appropriate model to estimate the value of π. As the Gaussian distribution is a clustered state, so it is impossible to reasonably estimate the value of pi.

Q5. From your observation and results of Part II, what is the best strategy that should be adopted by the penalty taker? What is the best strategy that should be adopted by the goalkeeper?

As shown in Figure 7 & 8 & 9, regardless of the goalkeeper's strategy of frequent puck dives or uniform posture distribution, the top right corner and the top left corner has the fewest blue dots, that is, *if the penalty taker kicks the ball to the top left or right corner*, the probability of scoring will be increased, and it is the *best* strategy for penalty takers.

And regardless of the ball's distribution, the middle two columns of the second row have the fewest red dots, that is, *if the goalkeeper adopt position 1*, the probability of intercept will be increased, and it is the *best* strategy for goalkeepers.

5 Conclusions

In conclusion, this experiment was carried out by simulating Monte Carlo methods to obtain solutions to real-life random processes. MATLAB was used to simulate the process of penalty and goalkeeper defence and the ease of graphing in MATLAB was utilised. Experiments have shown that the optimal solution for the goalkeeper is to choose position one for defence and the optimal solution for the penalty taker is to choose to kick the ball towards the top left and top right corners. This experiment took advantage of MATLAB's graphing strengths and took advantage of the matrix in the code, which help to obtain the experimental results successfully.

References

[1] Department of Electrical Engineering & Electronics, "Experiment 22 - Monte Carlo Simulation", Experiment 22: 202223-ELEC273 - Applied Design & Industrial Awareness (instructure.com), Liverpool University, Sep. 2022.

Appendices

A.1 Code and Programmes

Functions

```
function [prob,rect,between] = simulation_Uniform(N)
% N points are scattered into the circle
% returning [probability of falling in, coordinates of falling in the door,
coordinates of falling inside and outside the door of the ball]
u = rand(N,1);
theta = rand(N,1).*(2*pi);
r = sqrt(u).*sqrt(5);
circle(:,1) = r.*cos(theta);
circle(:,2) = r.*sin(theta);
% Determine whether it is in a rectangle and return a logical matrix (01 matrix)
judgerect = (abs(circle(:,1))<2) & (abs(circle(:,2))<1);
rect = circle.*[judgerect(:),judgerect(:)];
between = circle - rect;
rect(~all(rect,2),:) = [];
between(~all(between,2),:) = [];%For 1 use red, 0 use blue
prob = sum(judgerect(:))/N;%calculate the probability of dropping in
end
```

```
function [prob,rect,between,point] = simulation_Gaussian(N)
% N points are scattered into the circle
% returning [probability of falling in, coordinates of falling in the door,
coordinates of falling inside and outside the door of the ball]
theta = rand(N,1).*(2*pi);
r0 = sqrt(5);
r = normrnd(0,r0,[N,1]);
% r = randn(N,1).*sqrt(5);
point(:,1) = r.*cos(theta);
point(:,2) = r.*sin(theta);
judgecircle = (point(:,1).*point(:,1)+point(:,2).*point(:,2)) < 5;
circle = point.*[judgecircle(:),judgecircle(:)];
circle(~all(circle,2),:) = [];
%Determine whether it is in a rectangle and return a logical matrix (01 matrix)
judgerect = (abs(circle(:,1))<2) & (abs(circle(:,2))<1);
rect = circle.*[judgerect(:),judgerect(:)];
between = circle - rect;
rect(~all(rect,2),:) = [];
between(~all(between,2),:) = [];%For 1 use red, 0 use blue
prob = sum(judgerect(:))/sum(judgecircle(:));%calculate the probability of
dropping in
end
```

```
function [pos,rect,right,between] = keeper_Uniform(N)
%Gaussian distribution, penalty taker N times, return score/circle/between
    goal =
[0,1,0,0,0;1,0,0,0,0;1,0,0,0,0;0,0,1,0,0;0,0,0,1,0;1,1,0,1,0;1,0,1,0,1;0,0,0,0,1
];
    [~,rect,between]=simulation_Uniform(N);
    %move the matrix
    rect_matrix = ceil([rect(:,1)+2,1-rect(:,2)]);
```

```
    keeper = round(rand(size(rect_matrix,1),1)*5+0.5);
    right = ~goal(sub2ind([4,2],rect_matrix(:,1),rect_matrix(:,2))+(keeper(:)-
ones(size(keeper,1),1)).*8);
    pos = sum(right(:))/(length(rect)+length(between));
end
```

```
function [pos,rect,right,between] = keeper_Gaussian(N)
% Gaussian distribution, penalty taker N times, return score/circle/between
    goal =
[0,1,0,0,0;1,0,0,0,0;1,0,0,0,0;0,0,1,0,0;0,0,0,1,0;1,1,0,1,0;1,0,1,0,1;0,0,0,0,1
];
    [~,rect,between]=simulation_Gaussian(N);
    % Move the matrix
    rect_matrix = ceil([rect(:,1)+2,1-rect(:,2)]);
    keeper = round(rand(size(rect_matrix,1),1)*5+0.5);
    right = ~goal(sub2ind([4,2],rect_matrix(:,1),rect_matrix(:,2))+(keeper(:)-
ones(size(keeper,1),1)).*8);
    pos = sum(right(:))/(length(rect)+length(between));
end
```

```
function [pos,rect,right,between] = keeper_lower_Gaussian(N)
% 90% pounce
    goal =
[0,1,0,0,0;1,0,0,0,0;1,0,0,0,0;0,0,1,0,0;0,0,0,1,0;1,1,0,1,0;1,0,1,0,1;0,0,0,0,1
];
    [~,rect,between]=simulation_Gaussian(N);
    rect_matrix = ceil([rect(:,1)+2,1-rect(:,2)]);
    ran1 = rand(size(rect_matrix,1),1);
    ran2 = rand(size(rect_matrix,1),1);
    keeper = round((ran1<0.1).*(ran2*3+0.5)+(ran1>=0.1).*(ran2*2+3.5));
    right = ~goal(sub2ind([4,2],rect_matrix(:,1),rect_matrix(:,2))+(keeper(:)-
1).*8);
    pos = sum(right(:))/(length(rect)+length(between));
end
```

```
function [pos,rect,right,between] = keeper_lower_Uniform(N)
% 90% pounce
    goal =
[0,1,0,0,0;1,0,0,0,0;1,0,0,0,0;0,0,1,0,0;0,0,0,1,0;1,1,0,1,0;1,0,1,0,1;0,0,0,0,1
];
    [~,rect,between]=simulation_Uniform(N);
    rect_matrix = ceil([rect(:,1)+2,1-rect(:,2)]);
    ran1 = rand(size(rect_matrix,1),1);
    ran2 = rand(size(rect_matrix,1),1);
    keeper = round((ran1<0.1).*(ran2*3+0.5)+(ran1>=0.1).*(ran2*2+3.5));
    right = ~goal(sub2ind([4,2],rect_matrix(:,1),rect_matrix(:,2))+(keeper(:)-
1).*8);
    pos = sum(right(:))/(length(rect)+length(between));
end
```

Part I-Task 2

```
function aveprob = task2(N,R)
    totalprob = 0;
    for i = 1:R
        [prob,~,~] = simulation_Uniform(N);
        totalprob = totalprob + prob;
    end
    aveprob = totalprob/R;
end
```

Part I-Task 3

```
clear
[~,rect,between] = simulation_Uniform(1000);
scatter(rect(:,1),rect(:,2),20,'red','x')
hold on
scatter(between(:,1),between(:,2),20,'blue','o')
axis equal;%Make the coordinate system proportioned
legend('score','miss')
r = sqrt(5);
a = [-r, -r, 2*r, 2*r];
rectangle('Position',a,'Curvature',[1 1]);
rectangle('Position',[-2 -1 4 2]);
title('The distribution of score and miss (Uniform distribution)')
```

Part I-Task 4

```
clear
hun = task2(100,5);
tho = task2(1000,5);
ten_tho = task2(10000,5);
hun_tho = task2(100000,5);
plot(2:5,[hun,tho,ten_tho,hun_tho],'-*b')
hold on
x = 2:5;% theoretical value
y = 0*x + 0.5093;
plot(x,y)
legend('experimental value','theoretical value')
title('The probability against the value N')
xlabel('Attempt number (N)')
ylabel('Probability')
```

Part I-Task 5

```
clear
fiv = task2(1000,5);
ten = task2(1000,10);
fif = task2(1000,15);
twe = task2(1000,20);
plot(2:5,[fiv,ten,fif,twe],'-*b')
hold on
x = 2:5;% theoretical value
y = 0*x + 0.5093;
plot(x,y)
legend('experimental value','theoretical value')
```

```
title('The probability against the value R')
xlabel('Repetition number (R)')
ylabel('Probability')
```

Part I-Task 7

```
function aveprob = task7_2(N,R)
    totalprob = 0;
    for i = 1:R
        [prob,~,~] = simulation_Gaussian(N);
        totalprob = totalprob + prob;
    end
    aveprob = totalprob/R;
end
```

```
clear
[~,rect,between] = simulation_Gaussian(1000);
scatter(rect(:,1),rect(:,2),20,'red','x')
hold on
scatter(between(:,1),between(:,2),20,'blue','o')
axis equal;%Make the coordinate system proportioned
legend('score','miss')
r = sqrt(5);
a = [-r, -r, 2*r, 2*r];
rectangle('Position',a,'Curvature',[1 1]);
rectangle('Position',[-2 -1 4 2]);
title('The distribution of score and miss (Gaussian distribution)')
```

```
clear
hun = task7_2(100,5);
tho = task7_2(1000,1);
ten_tho = task7_2(10000,5);
hun_tho = task7_2(100000,5);
plot(2:5,[hun,tho,ten_tho,hun_tho],'-*b')
legend('experimental value')
title('The probability against the value N')
xlabel('Attempt number (N)')
ylabel('Probability')
```

```
clear
fiv = task7_2(1000,5);
ten = task7_2(1000,10);
fif = task7_2(1000,15);
twe = task7_2(1000,20);
plot(2:5,[fiv,ten,fif,twe],'-*b')
legend('experimental value')
title('The probability against the value R')
xlabel('Repetition number (R)')
ylabel('Probability')
```

Part II-Task 8

```
clear

[hun,rect_hun,right_hun,between_hun] = keeper_Uniform(100);
subplot(2,1,1);
colour = right_hun.*[1,0,0]+(1-right_hun).*[0,0,1];
scatter(rect_hun(:,1),rect_hun(:,2),10,colour,"filled")
hold on
scatter(between_hun(:,1),between_hun(:,2),10,'b','filled')
axis equal;
legend('score','miss')
r = sqrt(5);
a = [-r, -r, 2*r, 2*r];
rectangle('Position',a,'Curvature',[1 1]);
rectangle('Position',[-2 -1 2 2]);
rectangle('Position',[-2 -1 2 1]);
rectangle('Position',[-2 -1 1 1]);
rectangle('Position',[-2 -1 1 2]);
rectangle('Position',[-1 -1 2 2]);
rectangle('Position',[-1 -1 2 1]);
rectangle('Position',[1 -1 1 1]);
rectangle('Position',[1 -1 1 2]);
title('The distribution of score and miss (Uniform distribution, N=100)')
fprintf("The probability when N is 100 = %.4f\n",hun);


[tho,rect_tho,right_tho,between_tho] = keeper_Uniform(1000);
subplot(2,1,2);
colour = right_tho.*[1,0,0]+(ones(length(right_tho),1)-right_tho).*[0,0,1];
scatter(rect_tho(:,1),rect_tho(:,2),10,colour,'filled')
hold on
scatter(between_tho(:,1),between_tho(:,2),10,'b','filled')
axis equal;
legend('score','miss')
r = sqrt(5);
a = [-r, -r, 2*r, 2*r];
rectangle('Position',a,'Curvature',[1 1]);
rectangle('Position',[-2 -1 2 2]);
rectangle('Position',[-2 -1 2 1]);
rectangle('Position',[-2 -1 1 1]);
rectangle('Position',[-2 -1 1 2]);
rectangle('Position',[-1 -1 2 2]);
rectangle('Position',[-1 -1 2 1]);
rectangle('Position',[1 -1 1 1]);
rectangle('Position',[1 -1 1 2]);
title('The distribution of score and miss (Uniform distribution, N=1000)')
fprintf("The probability when N is 1000 = %.4f\n",tho);
```

Part II-Task 9

```
clear

[hun,rect_hun,right_hun,between_hun] = keeper_Gaussian(100);
subplot(2,1,1);
colour = right_hun.*[1,0,0]+(1-right_hun).*[0,0,1];
scatter(rect_hun(:,1),rect_hun(:,2),10,colour,'filled')
```

```
hold on
scatter(between_hun(:,1),between_hun(:,2),10,'b','filled')
axis equal;
legend('score','miss')
r = sqrt(5);
a = [-r, -r, 2*r, 2*r];
rectangle('Position',a,'Curvature',[1 1]);
rectangle('Position',[-2 -1 2 2]);
rectangle('Position',[-2 -1 2 1]);
rectangle('Position',[-2 -1 1 1]);
rectangle('Position',[-2 -1 1 2]);
rectangle('Position',[-1 -1 2 2]);
rectangle('Position',[-1 -1 2 1]);
rectangle('Position',[1 -1 1 1]);
rectangle('Position',[1 -1 1 2]);
title('The distribution of score and miss (Gaussain distribution, N=100)')
fprintf("The probability when N is 100 = %.4f\n",hun);


[tho,rect_tho,right_tho,between_tho] = keeper_Gaussian(1000);
subplot(2,1,2);
colour = right_tho.*[1,0,0]+(ones(length(right_tho),1)-right_tho).*[0,0,1];
scatter(rect_tho(:,1),rect_tho(:,2),10,colour,'filled')
hold on
scatter(between_tho(:,1),between_tho(:,2),10,'b','filled')
axis equal;
legend('score','miss')
r = sqrt(5);
a = [-r, -r, 2*r, 2*r];
rectangle('Position',a,'Curvature',[1 1]);
rectangle('Position',[-2 -1 2 2]);
rectangle('Position',[-2 -1 2 1]);
rectangle('Position',[-2 -1 1 1]);
rectangle('Position',[-2 -1 1 2]);
rectangle('Position',[-1 -1 2 2]);
rectangle('Position',[-1 -1 2 1]);
rectangle('Position',[1 -1 1 1]);
rectangle('Position',[1 -1 1 2]);
title('The distribution of score and miss (Gaussain distribution, N=1000)')
fprintf("The probability when N is 1000 = %.4f\n",tho);
```

Part II-Task 10

```
clear
[hun_uni,rect_hun_uni,right_hun_uni,between_hun_uni] =
keeper_lower_Uniform(100);
subplot(2,2,1);
colour = right_hun_uni.*[1,0,0]+(1-right_hun_uni).*[0,0,1];
scatter(rect_hun_uni(:,1),rect_hun_uni(:,2),10,colour,"filled")
hold on
scatter(between_hun_uni(:,1),between_hun_uni(:,2),10,'b','filled')
axis equal;
legend('score','miss')
r = sqrt(5);
a = [-r, -r, 2*r, 2*r];
```

```
rectangle('Position',a,'Curvature',[1 1]);
rectangle('Position',[-2 -1 2 2]);
rectangle('Position',[-2 -1 2 1]);
rectangle('Position',[-2 -1 1 1]);
rectangle('Position',[-2 -1 1 2]);
rectangle('Position',[-1 -1 2 2]);
rectangle('Position',[-1 -1 2 1]);
rectangle('Position',[1 -1 1 1]);
rectangle('Position',[1 -1 1 2]);
title('The distribution of score and miss (Uniform distribution, N=100)')
fprintf("The probability when N is 100 (Uniform) = %.4f\n",hun_uni);



[thou_uni,rect_thou_uni,right_thou_uni,between_thou_uni] =
keeper_lower_Uniform(1000);
subplot(2,2,2);
colour = right_thou_uni.*[1,0,0]+(1-right_thou_uni).*[0,0,1];
scatter(rect_thou_uni(:,1),rect_thou_uni(:,2),10,colour,"filled")
hold on
scatter(between_thou_uni(:,1),between_thou_uni(:,2),10,'b','filled')
axis equal;
legend('score','miss')
r = sqrt(5);
a = [-r, -r, 2*r, 2*r];
rectangle('Position',a,'Curvature',[1 1]);
rectangle('Position',[-2 -1 2 2]);
rectangle('Position',[-2 -1 2 1]);
rectangle('Position',[-2 -1 1 1]);
rectangle('Position',[-2 -1 1 2]);
rectangle('Position',[-1 -1 2 2]);
rectangle('Position',[-1 -1 2 1]);
rectangle('Position',[1 -1 1 1]);
rectangle('Position',[1 -1 1 2]);
title('The distribution of score and miss (Uniform distribution, N=1000)')
fprintf("The probability when N is 1000 (Uniform) = %.4f\n",thou_uni);



[hun_gau,rect_hun_gau,right_hun_gau,between_hun_gau] =
keeper_lower_Gaussian(100);
subplot(2,2,3);
colour = right_hun_gau.*[1,0,0]+(1-right_hun_gau).*[0,0,1];
scatter(rect_hun_gau(:,1),rect_hun_gau(:,2),10,colour,"filled")
hold on
scatter(between_hun_gau(:,1),between_hun_gau(:,2),10,'b','filled')
axis equal
legend('score','miss')
r = sqrt(5);
a = [-r, -r, 2*r, 2*r];
rectangle('Position',a,'Curvature',[1 1]);
rectangle('Position',[-2 -1 2 2]);
rectangle('Position',[-2 -1 2 1]);
rectangle('Position',[-2 -1 1 1]);
rectangle('Position',[-2 -1 1 2]);
rectangle('Position',[-1 -1 2 2]);
rectangle('Position',[-1 -1 2 1]);
rectangle('Position',[1 -1 1 1]);
rectangle('Position',[1 -1 1 2]);
```

```
title('The distribution of score and miss (Gaussian distribution, N=100)')
fprintf("The probability when N is 100 (Gaussian) = %.4f\n",hun_gau);




[thou_gau,rect_thou_gau,right_thou_gau,between_thou_gau] =
keeper_lower_Gaussian(1000);
subplot(2,2,4);
colour = right_thou_gau.*[1,0,0]+(1-right_thou_gau).*[0,0,1];
scatter(rect_thou_gau(:,1),rect_thou_gau(:,2),10,colour,"filled")
hold on
scatter(between_thou_gau(:,1),between_thou_gau(:,2),10,'b','filled')
axis equal;
legend('score','miss')
r = sqrt(5);
a = [-r, -r, 2*r, 2*r];
rectangle('Position',a,'Curvature',[1 1]);
rectangle('Position',[-2 -1 2 2]);
rectangle('Position',[-2 -1 2 1]);
rectangle('Position',[-2 -1 1 1]);
rectangle('Position',[-2 -1 1 2]);
rectangle('Position',[-1 -1 2 2]);
rectangle('Position',[-1 -1 2 1]);
rectangle('Position',[1 -1 1 1]);
rectangle('Position',[1 -1 1 2]);
title('The distribution of score and miss (Gaussain distribution, N=1000)')
fprintf("The probability when N is 1000 (Gaussian) = %.4f\n",thou_gau);
```

A.2 Screenshot

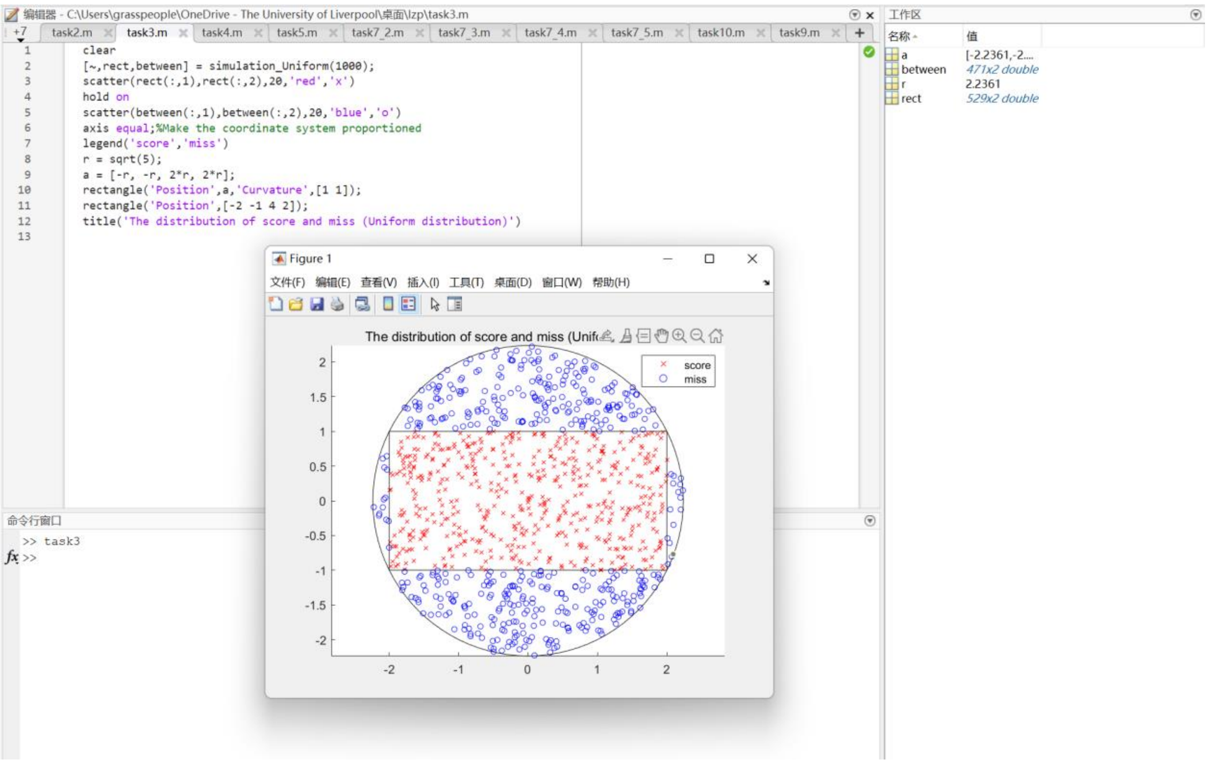


Figure 13. Screenshot of Task 3

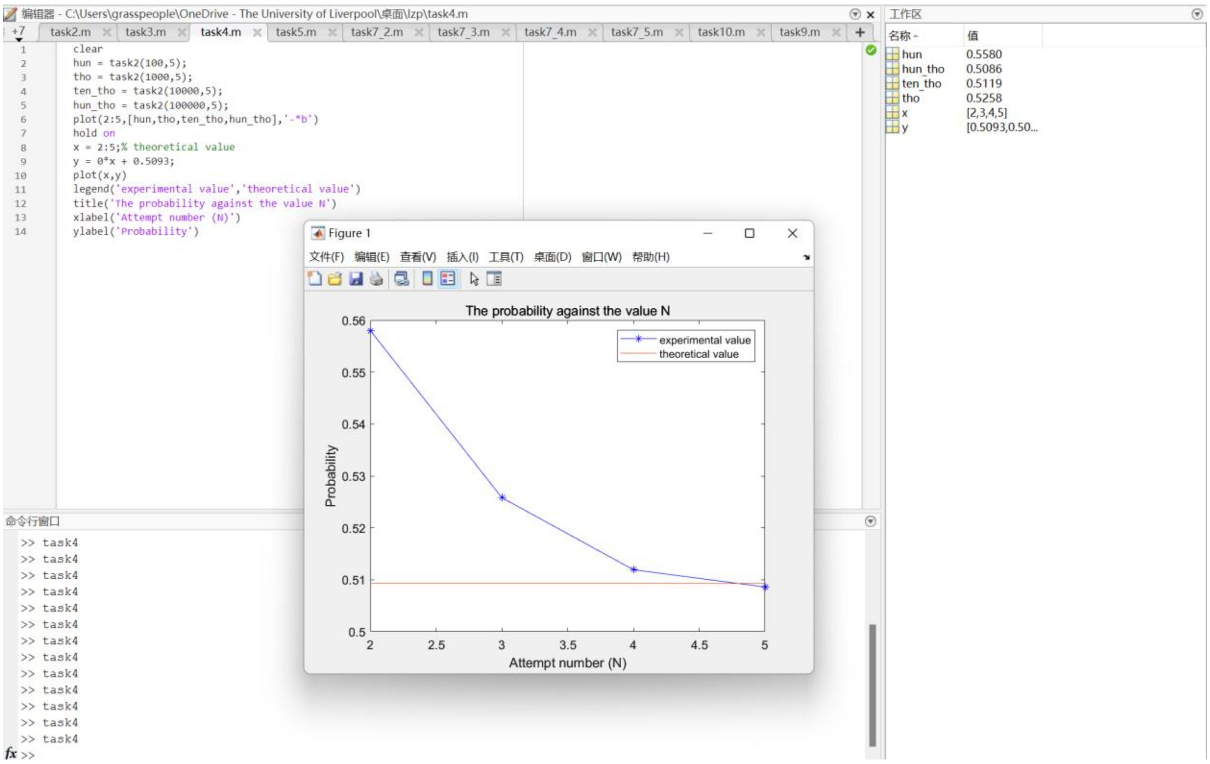


Figure 14. Screenshot of Task 4

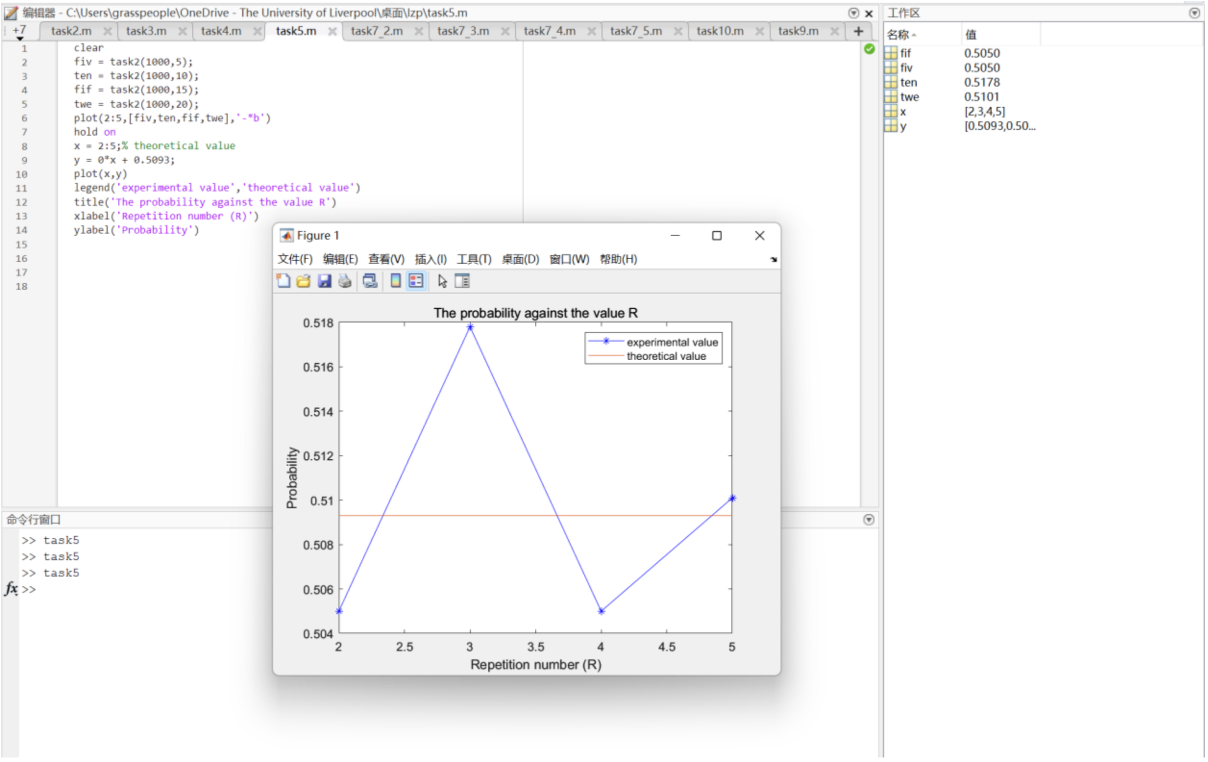


Figure 15. Screenshot of Task 5

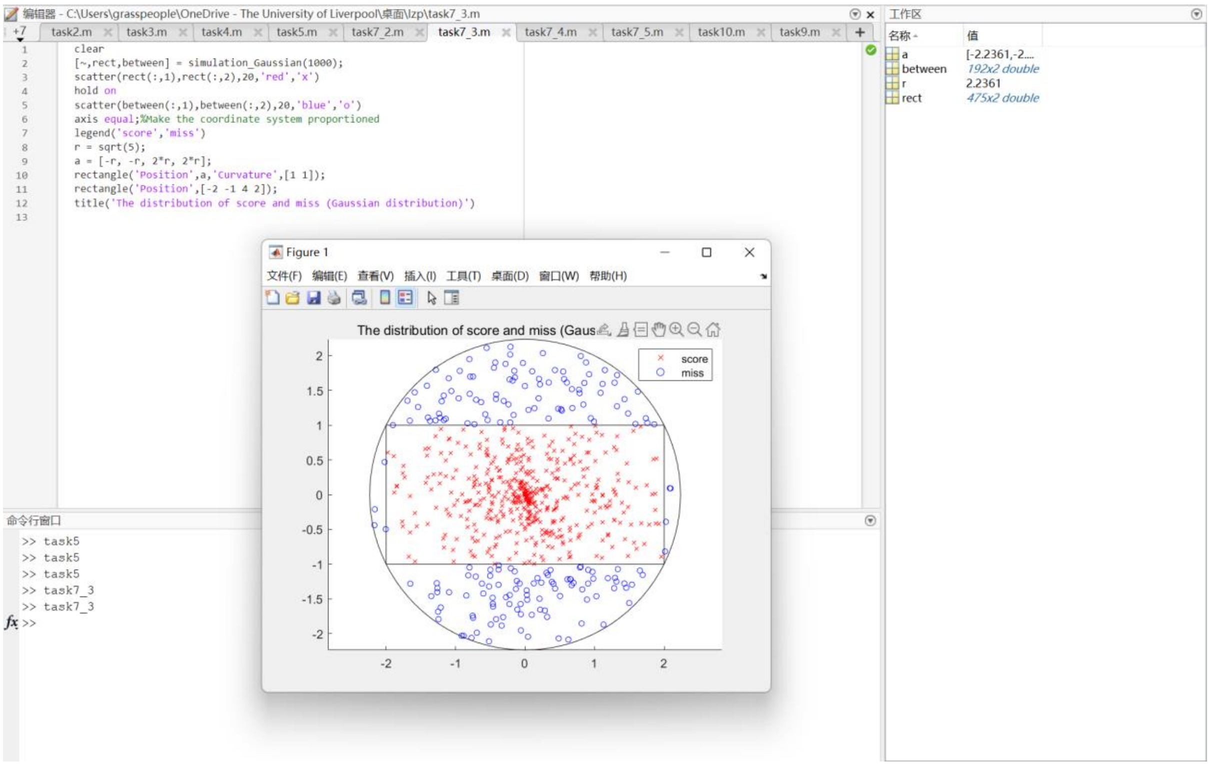


Figure 16. Screenshot of Task 7_3

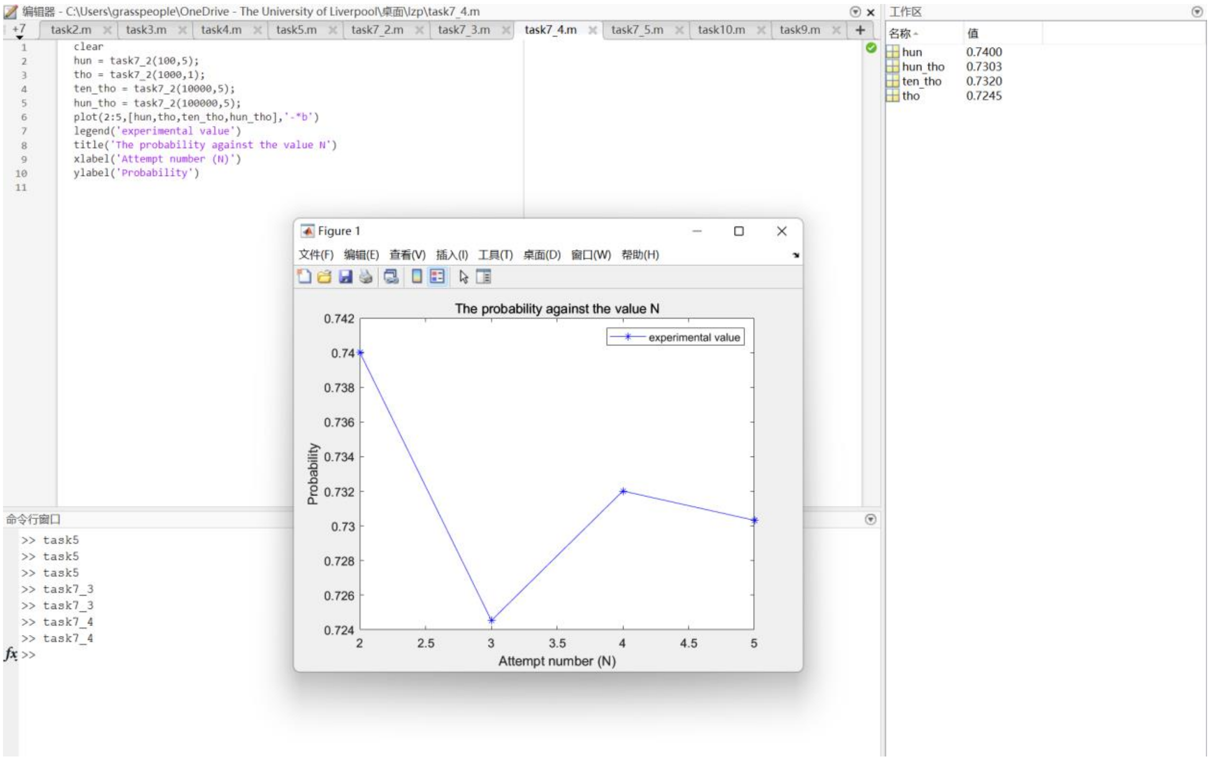


Figure 17. Screenshot of Task 7_4

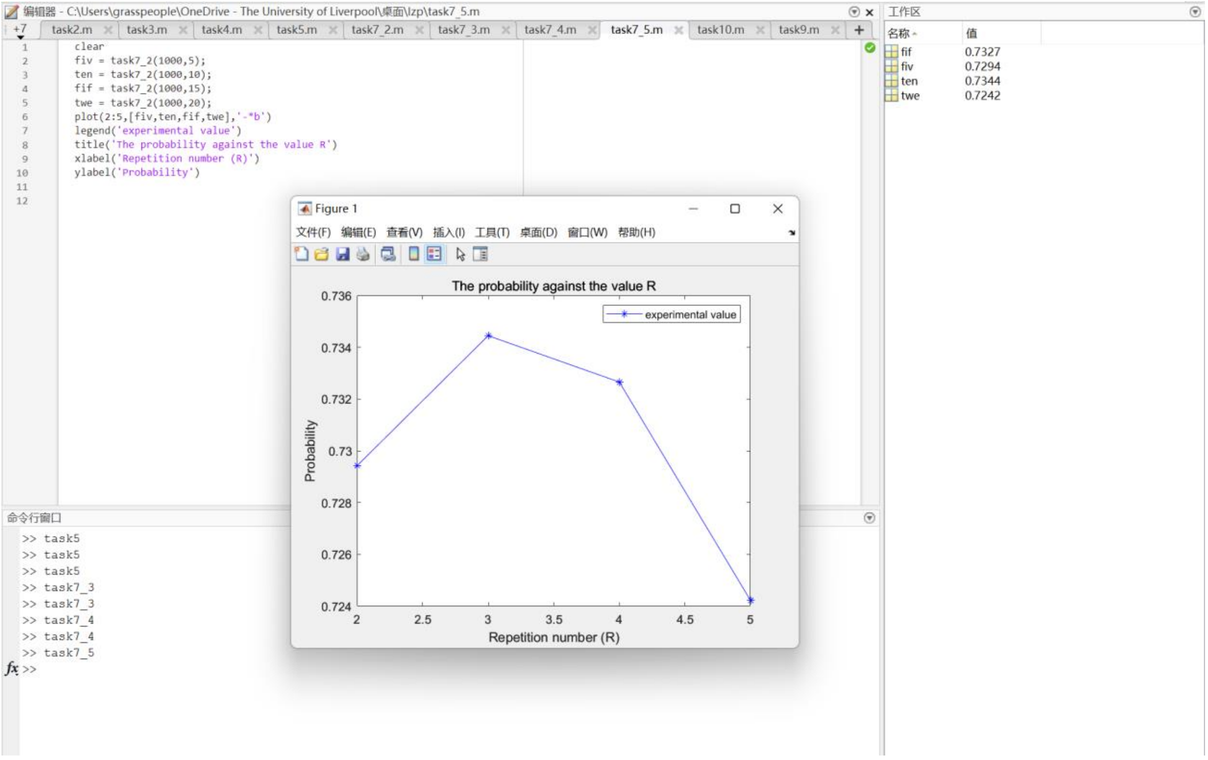


Figure 18. Screenshot of Task 7_5

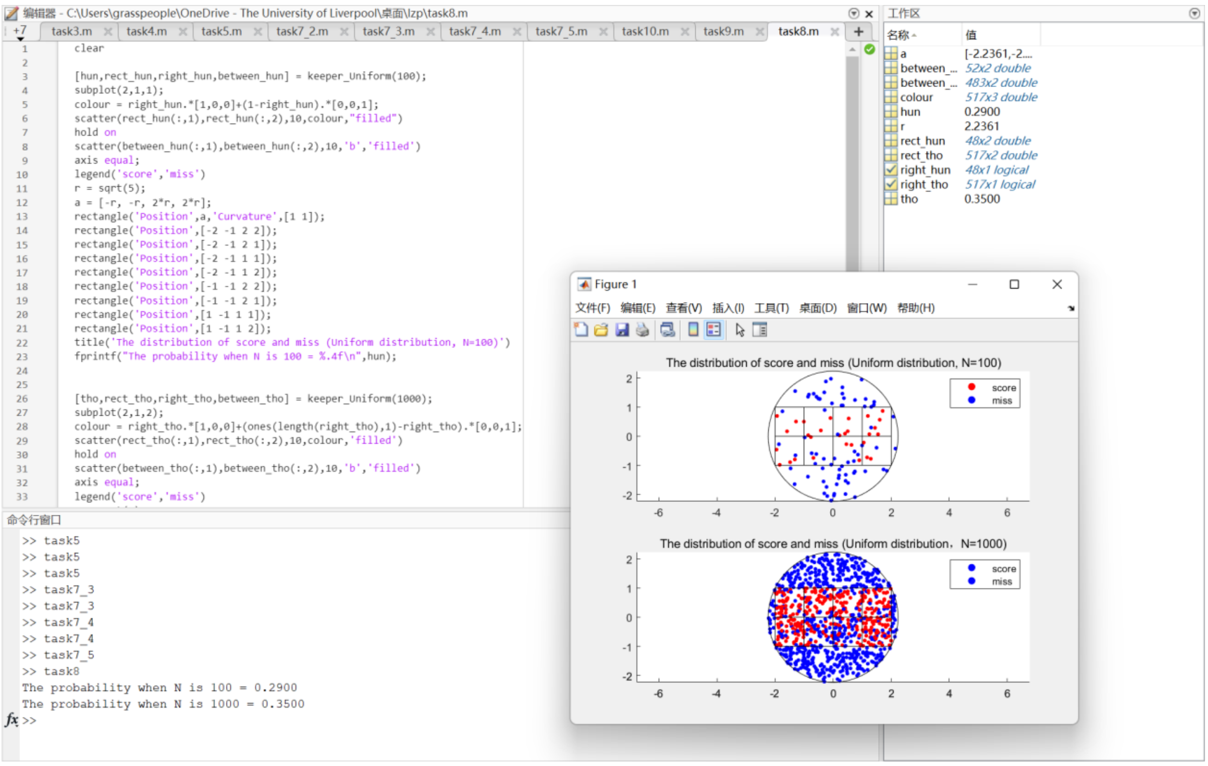


Figure 19. Screenshot of Task 8

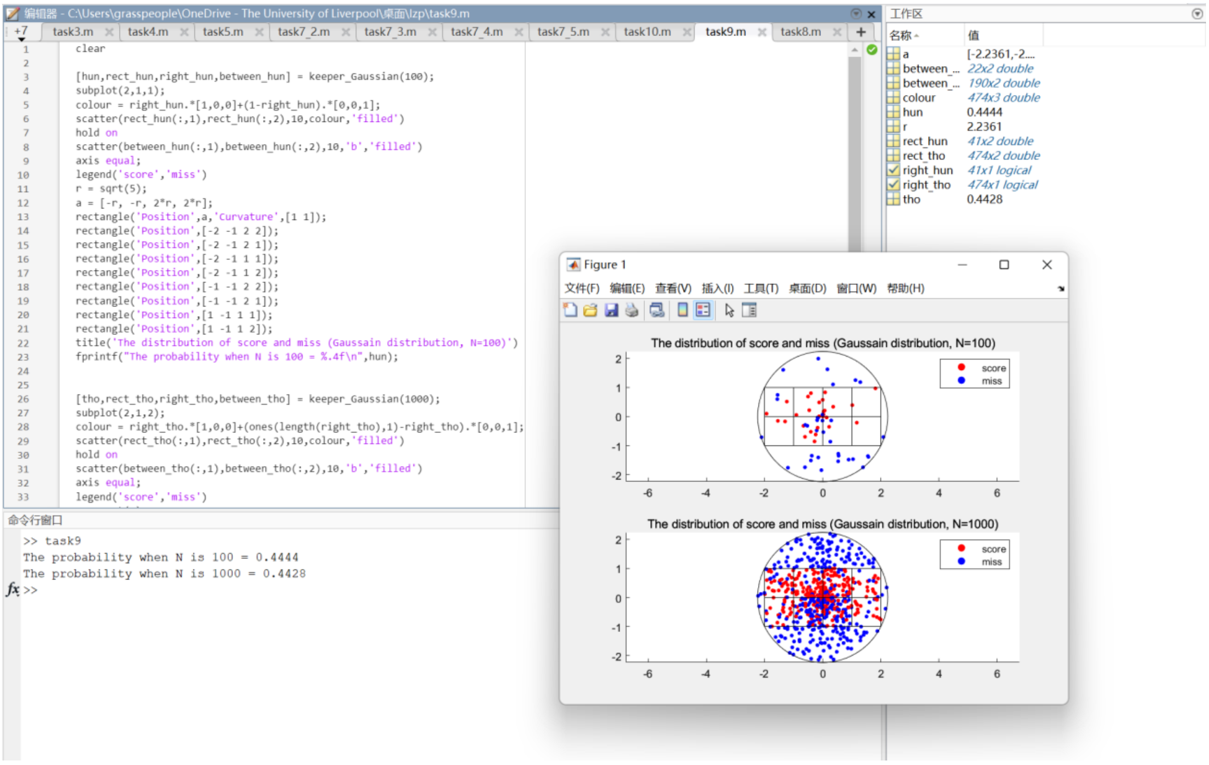


Figure 20. Screenshot of Task 9

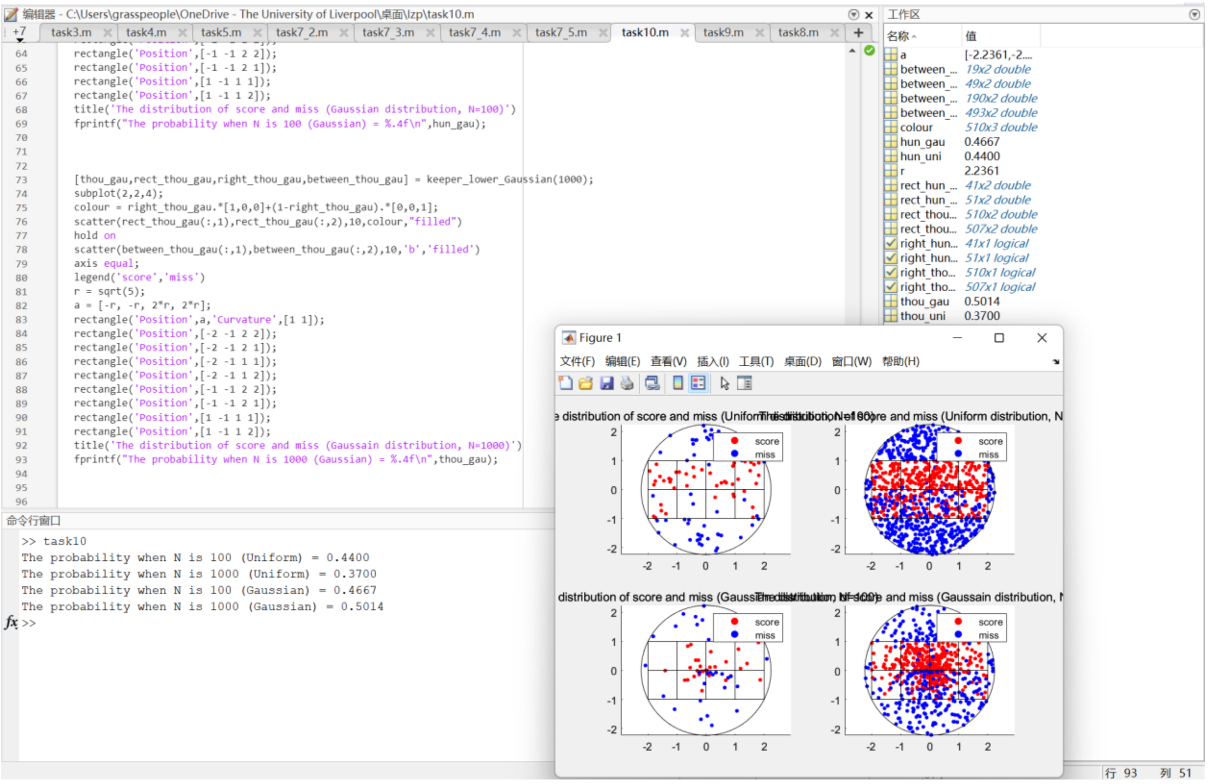


Figure 21. Screenshot of Task 10