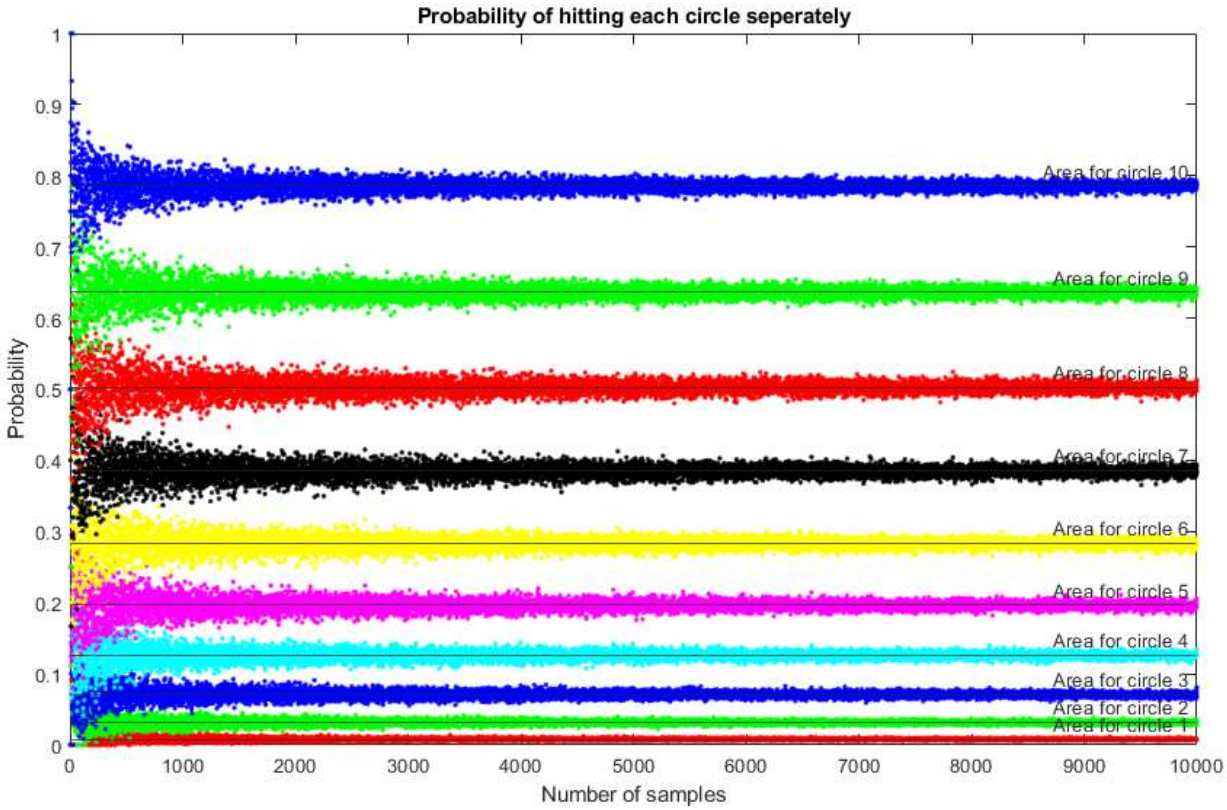
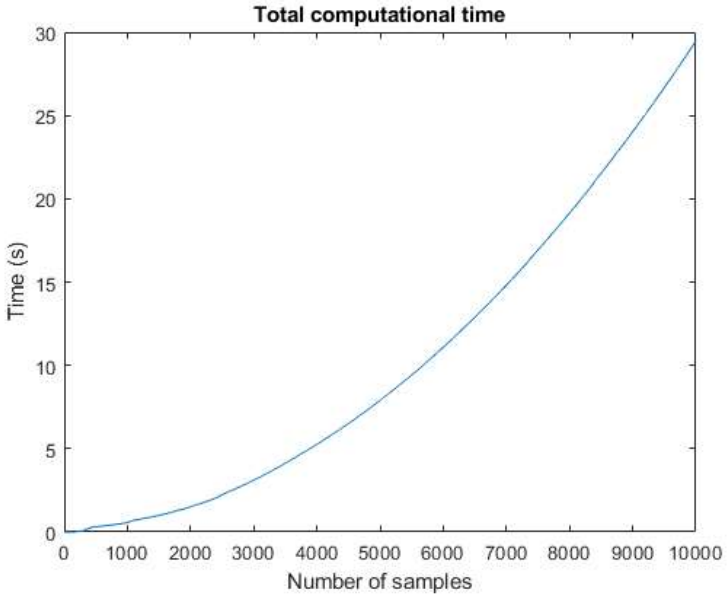
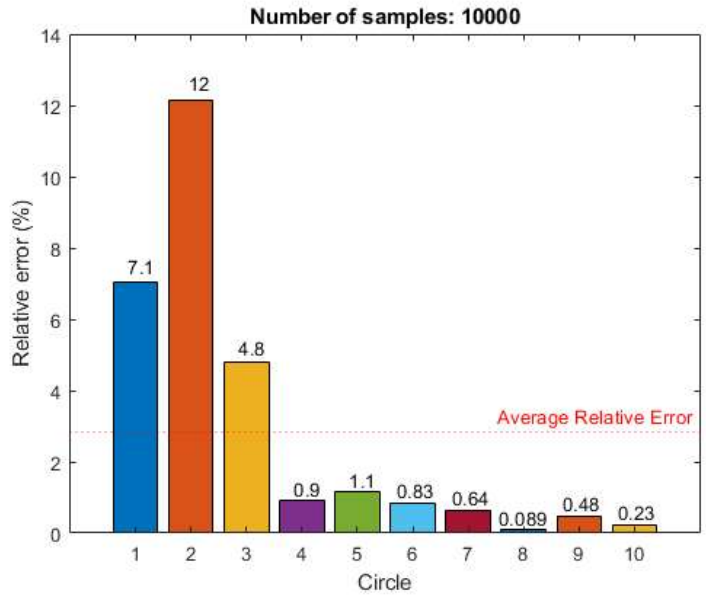
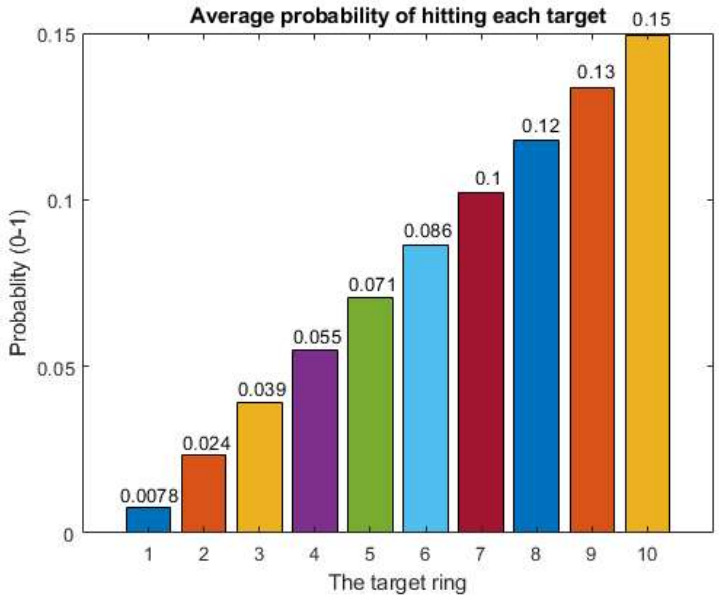
```
% Numerical Methods in Engineering Week 5
% Dominik Lasinski 791351

clc
clear
close all

% Center of the board
xc = 0.5;
yc = 0.5;

number_samples = 10000; % The number of random samples
number_circles = 10; % The number of circles

area_analytical = zeros(number_circles, 1); % Analytical results for each circle
area_numerical = zeros(number_circles, number_samples); % Numerical results for each circle
number_shots = zeros(number_circles, number_samples); % Number of shots for each circle
radius = zeros(number_circles, 1); % Size of the radius for each circle
errors = zeros(number_circles, 1); % Error for each circle
area_ring = zeros(number_circles, 1); % Area/Probability of each ring

% How to read the arrays
% area_numerical(1, :) refers to the smallest circle
% area_numerical(2, :) refers to the second smallest circle
% area_ring(2) refers to the second smallest ring, ie. area_numerical(2, :) - area_numerical(1, :)

for k = 1:number_circles
    radius(k) = k*0.05; % Radius for each circle
    area_analytical(k) = (pi*(radius(k)^2)); % Analytical area of each circle
end


for q = 1 : number_samples
    start_time(q) = cputime; % Start the time
    n_max(q) = q;
    for m = 1:q
        xn = rand(1,1); % Random x point
        yn = rand(1,1); % Random y point
        r = sqrt((xn-xc)^2+(yn-yc)^2); % The distance from the centre

        for w = 1:number_circles
            if r <= radius(w) % If the shot is within the circle
                number_shots(w, q) = number_shots(w, q) + 1;
            end
        end
    end
    % End the time
    if q == 1
        end_time(q) = cputime - start_time(q);
    else
        end_time(q) = cputime - start_time(q) + end_time(q-1);
    end
end

% Calculating the area for each sample and circle
for y = 1: number_samples
    for x = 1: number_circles
        area_numerical(x, y) = number_shots(x, y) / y;
    end
end

% Calculating the average area of each ring
average_area_numerical = mean(area_numerical, 2); % Average area of each circle
area_ring(1) = average_area_numerical(1);
for u= 2:10
    area_ring(u) = average_area_numerical(u) - sum(average_area_numerical(u-1));
end


% The errors for only the maximum number of samples
for p = 1: number_circles
    errors(p) = abs((area_numerical(p, number_samples) - area_analytical(p)) / area_analytical(p)) * 100; % Percentage
end
avg_error = sum(errors)/number_circles; % Average error



% Plotting and Visualization
figure(1)
set(gcf,'position',[100,100,1000,600])
plot(n_max, area_numerical(1, :), 'r.');
hold on
```

```
yline(area_analytical(1), '-', 'Area for circle 1');
hold on
plot(n_max, area_numerical(2, :), 'g.');
hold on
yline(area_analytical(2), '-', 'Area for circle 2');
hold on
plot(n_max, area_numerical(3, :), 'b.');
hold on
yline(area_analytical(3), '-', 'Area for circle 3');
hold on
plot(n_max, area_numerical(4, :), 'c.');
hold on
yline(area_analytical(4), '-', 'Area for circle 4');
hold on
plot(n_max, area_numerical(5, :), 'm.');
hold on
yline(area_analytical(5), '-', 'Area for circle 5');
hold on
plot(n_max, area_numerical(6, :), 'y.');
hold on
yline(area_analytical(6), '-', 'Area for circle 6');
hold on
plot(n_max, area_numerical(7, :), 'k.');
hold on
yline(area_analytical(7), '-', 'Area for circle 7');
hold on
plot(n_max, area_numerical(8, :), 'r.');
hold on
yline(area_analytical(8), '-', 'Area for circle 8');
hold on
plot(n_max, area_numerical(9, :), 'g.');
hold on
yline(area_analytical(9), '-', 'Area for circle 9');
hold on
plot(n_max, area_numerical(10, :), 'b.');
hold on
yline(area_analytical(10), '-', 'Area for circle 10');
xlabel('Number of samples');
ylabel('Probability');
title('Probability of hitting each circle seperately');
% Comment: I had no idea how to implement different colors in a loop, so I
% wrote them seperately


figure(2)
plot(n_max, end_time);
title('Total computational time');
ylabel('Time (s)');
xlabel('Number of samples');

figure(3)
b = bar(diag(errors), 'stacked');
title(['Number of samples: ' num2str(number_samples)]);
xlabel('Circle');
ylabel('Relative error (%)');
xtips = b.XEndPoints;
ytips = errors*1.01;
text(xtips, ytips, num2str(errors,2),'HorizontalAlignment','center','VerticalAlignment','bottom');
hold on
yline(avg_error,':r','Average Relative Error');

figure(4)
b = bar(diag(area_ring), 'stacked');
title('Average probability of hitting each target');
ylabel('Probablity (0-1)');
xlabel('The target ring');
xtips2 = b.XEndPoints;
ytips2 = area_ring*1.01;
text(xtips2, ytips2, num2str(area_ring,2),'HorizontalAlignment','center','VerticalAlignment','bottom');
hold on
```


Probability of hitting each circle seperately
Probability
Number of samples
Area for circle 10
Area for circle 9
Area for circle 8
Area for circle 7
Area for circle 6
Area for circle 5
Area for circle 4
Area for circle 3
Area for circle 2
Area for circle 1


Total computational time
Time (s)
Number of samples





Published with MATLAB® R2019b