# Problem Statement

We need to do following simulations using Monte Carlo method:

1. To estimate the value of π using area method.
2. To evaluate the integral: $I(n) = \int_{(n-1)\pi}^{n\pi} \frac{\sin(x)}{x}\, dx$ for n=1,2,3,4,5.

   Also $D(n) = \int_{0}^{n\pi} \frac{\sin(x)}{x}\, dx$ for n=10, 100, 1000 and speculate the value of Dirichlet integral.
3. To find the probabilities of different poker hands

# Problem 1

## Theoretical Knowledge

- We will estimate the value of PI by taking random points inside a unit square and by counting the number of points inside the unit circle with centre at origin.

$$Estimated\ value\ of\ \pi/4 = \frac{Number\ of\ points\ in\ circle}{Number\ of\ points\ in\ square}$$

- Here, to find the confidence interval, we consider the random variable as below:

$$P_i(k) = \begin{cases} 1 & if\ point\ is\ inside\ the\ circle \\ 0 & if\ point\ is\ outside\ the\ circle \end{cases}$$

- We take random variable $\hat{p} = \frac{\sum_{i=1}^{n} P_i}{n}$. As we know that we are estimating value of π/4 so, expected value $E[\widehat{p}] = \frac{\pi}{4}$ and variance $Var(\hat{p}) = \frac{\pi}{4}\left(1 - \frac{\pi}{4}\right) = 0.1685$.
- For 95% confidence from the normal tables the value of β=1.96. So, the number of points for error in value of p is given by

$$n = \left(\frac{\beta\ std(\hat{p})}{p - \hat{p}}\right)^2$$

- For example, for 95 % confidence and 1% error (0.7854*0.01=0.007854) in value of π/4, the number of points required is given by $n = \left(\frac{1.96*\sqrt{0.1685}}{0.007854}\right)^2 = 10{,}494$.

## Simulation

We take 10000 points (x,y) using the rand function and check if the point is inside the circle of radius 1. Then we estimate the value of π/4 by counting the number of points in circle and number of points in square. We also find the number of points required for confidence of 95% and error of 1%.

**Code:**

```
% MATLAB program to estimate value of PI by Monte Carlo method
n_of_samples = 10000; % Number of points to be taken inside the unit square
points=0;
for i=1:n_of_samples
    x=rand();          % Generate random points in the unit square
    y=rand();
    if(x^2+y^2 <= 1) % Condition for points inside the circle
        points=points+1;
    end
    point(i)=(points/n_of_samples); % Estimated value of pi/4
end
plot(1:n_of_samples,point)
hold on
A(1:n_of_samples)=3.14/4;
```

```
plot(1:n_of_samples,A,'red','LineWidth',3)
legend('Estimated value of pi/4','Actual value of pi/4');

% Calculation of number of samples required for 95% confidence interval
with 1% error
pcap=pi/4;
xbar_pcap=pi/4; % mean of pcap
var_pcap=(pi/4)*(1-(pi/4)); % variance of pcap
std_pcap=sqrt(var_pcap); % standard deviation of pcap
beta=1.96; % For 95% confidence
error=0.01; % 1% error in value of pi/4
n=(beta*std_pcap/(error*pcap))^2

% Calculation of confidence interval with 95%
error=(beta*std_pcap)/sqrt(n_of_samples)
CI_lower=pcap-error % Lower bound of confidence interval
CI_upper=pcap+error % Upper bound of confidence interval
```

**Output:**

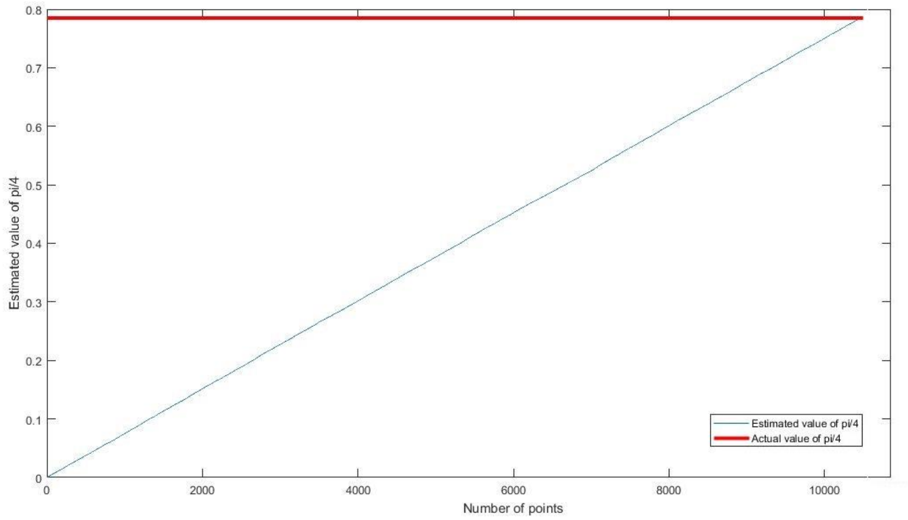


*Figure 1 Plot of Estimated value of pi/4 to the number of samples*

From figure 1 we can see that for as sample points increases, the estimated value of π/4 reaches closer to the actual value of π/4.

| | Actual value | Estimated value |
|---|---|---|
| Value of π/4 | 0.7854 | 0.7908 |

The error in estimated value is 0.7854-0.7908=-0.0054 which is within 1% error as we calculated for 10000 sample points with 95% confidence level.

The output of confidence intervals for 10000 sample points is ***0.7774 ≤ (Estimated value of π/4) ≤ 0.7934***. We can see that the estimated value obtained is well within the range.

In order to test the confidence level let's calculate the error for 1000 sample points and compare the error with the simulation results.

$$1000 = \left(\frac{1.96*\sqrt{0.1685}}{error}\right)^2 \text{ implies that } error = \pm 0.0254$$

| Theoretical error | Simulated error |
|---|---|
| ±0.0254 | (0.7854-0.7641) = +0.0213 |

We can see that the simulated error is closer to theoretical error for 1000 samples with 95% confidence level.

# Problem 2

## Theoretical Knowledge

- Suppose we want to find value of $I = \int_a^b g(x)\ dx$, then we can rewrite the integral using substitution principle by assuming $y = \frac{x-a}{a-b}$ and $dy = \frac{dx}{b-a}$. So, the final integral becomes

$$I = \int_0^1 g(a + (b-a)y)\ (b-a)\ dy$$

- Here, y varies uniformly between 0 to 1. So, we take 1000 random samples of y between 0 to 1 and evaluate $I = \sum_{i=1}^{1000} g(a + (b-a)y(i))\ (b-a)$.
- Hence, $I(n) = \int_{(n-1)\pi}^{n\pi} \frac{\sin(x)}{x}\ dx$, we can rewrite integral as $I(n) = \int_0^1 \frac{\sin((n-1)\pi + \pi y)}{(n-1+y)} dy$ for n=1,2,3,4,5. Similarly, D(n) can be written as $D(n) = \int_0^1 \frac{\sin(n\pi y)}{y} dy$ for n=10,100,1000.
- The value of Dirichlet integral $\int_0^{n\pi} \frac{\sin(x)}{x}\ dx$ converges to π/2 as n→∞

## Simulation

**Code:**

```
% MATLAB program to evalute the integral I(n) and D(n)
y=rand(1,10000); % Take 10000 uniform random samples between 0 and 1

% Calculate integral value I(n) for n=1,2,3,4,5
for n=1:5
    hx=sin((n-1)*pi + pi.*y)./(n-1+y);
    In(n)=sum(hx)/10000;
end

n=[10,100,1000];
% Calculate integral value of D(n) for n=10,100,1000
for k=1:3
    % Run for 1000 iterations to get the expected value of D(n)
    for j=1:1000
        y=rand(1,10000);% Take 10000 uniform random samples between 0 and 1
        hx=sin(n(k)*pi.*y)./y;
        P(j)=sum(hx)/10000;
    end
    Dn(k)=mean(P);  % Expected value of D(n)
end
```

**Output:**

| Integral | Theoretical value | Simulated value |
|---|---|---|
| I (1) | 1.8519 | 1.8476 |
| I (2) | -0.4337 | -0.4316 |
| I (3) | 0.2566 | 0.2553 |
| I (4) | -0.1826 | -0.1817 |
| I (5) | 0.1418 | 0.1411 |

*Table 1 Theoretical and Simulated values of I (n) for n=1,2,3,4,5*

| Integral | Theoretical value | Simulated value |
|---|---|---|
| D (10) | 1.5390 | 1.5394 |
| D (100) | 1.5676 | 1.5627 |
| D (1000) | 1.5704 | 1.5797 |

*Table 2 Theoretical and Simulated values of D (n) for n=10,100,1000*

From table 1, we can see that the simulated value obtained by the Monte Carlo approach is very close to the theoretical values of the integrals. From table 2, we see that as value of n increases from 10 to 1000, the value of integral converges to value of π/2. So, we can speculate that the value of Dirichlet integral:

$$D = \int_0^\infty \frac{\sin(x)}{x}\, dx = \frac{\pi}{2}$$

# Problem 3

## Theoretical Knowledge

A hand in poker consists of 5 cards, so there are total $\binom{52}{5}$ possibilities of the hands. The several types of hand and their number of possibilities are shown in table 3.

| Type of Hand | How can we choose from 52 cards deck | Mathematical Expression | Number of possibilities |
|---|---|---|---|
| Single pair | Type: AABCD We have 13 kinds and 4 of each kind to choose from. | $\binom{13}{1}\binom{4}{2}\binom{12}{3}\binom{4}{1}^3$ | 1098240 |
| Two pair | Type: AABBC We have 13C2 for 2 numbers showing on 2 pairs and each pair will have 2 out of 4 suits. So, 4C2*4C2. | $\binom{13}{2}\binom{4}{1}\binom{11}{1}\binom{4}{2}^2$ | 123552 |
| Triple | Type: AAABC We have 13C1 for number showing on 3 cards and 4C3 as we have 3 out of 4 suits. | $\binom{13}{1}\binom{4}{3}\binom{12}{2}\binom{4}{1}^2$ | 54912 |
| Full house | Type: AAABB We have 13C1 for number showing on 3 cards and 4C3 as we have 3 out of 4 suits. We have 12C1 for number showing on remaining two cards and 4C2 as we have 2 out of 4 suits for them. | $\binom{13}{1}\binom{4}{2}\binom{12}{1}\binom{4}{3}$ | 3744 |
| Four of a kind | Type: AAAAB We have 13C1 for number showing on 4 cards and 4C1 as we have 1 out of 4 suits. One remaining card can be 12C1. | $\binom{13}{1}\binom{12}{1}\binom{4}{1}$ | 624 |

| | | | |
|---|---|---|---|
| Straight | Type: 12345<br>We have 10C1 for 10 different ranks of straight and (4C1)^5 as we have 1 out of 4 suits for 5 cards. We are excluding royal flush and straight flush. | $\binom{10}{1}\binom{4}{1}^5 - \binom{10}{1}\binom{4}{1}$ | 10200 |
| Flush | Type: All same suit<br>We have 13C5 for five cards of same suit and 4C1 as we have 1 out of 4 suits. | $\binom{13}{5}\binom{4}{1} - \binom{10}{1}\binom{4}{1}$ | 5108 |
| Straight flush | Type: All from same suit with straight hand | $\binom{10}{1} - \binom{10}{1}\binom{4}{1}$ | 36 |
| Royal flush | We have 4C1 as we have only one royal flush in each suit. | $\binom{4}{1}$ | 4 |
| High Card | Type: ABCDE<br>We have 13C5 for 5 to choose from 13 cards and (4C1)^5 as we have 1 out of 4 suits. We subtract straights, flushes and royal flushes. | $\left[\binom{13}{5} - 10\right]\left[\binom{4}{1}^5 - 4\right]$ | 1302540 |
| **Total** | | $\binom{52}{5}$ | **2598960** |

*Table 3 Total number of possibilities for different types of hands in Poker*

## Simulation

The dealing of 5-card poker hand is simulated by the following code. The simulation of dealing the hand is done 100000 times. Before every run, deck is shuffled with the help of *randperm()* function. We draw the first 5 cards for the hand and then check for the type of hand. We will have a deck of cards from 1 to 52 having 1 to 13 are hearts, 14 to 26 are clubs, 27 to 39 are diamonds and 40 to 52 are spades.

**Code:**

```
% MATLAB program to simulate poker hands to find probability for different
% types of hands
number_of_runs=100000; % Simulate for 100000 hands
single_pair=zeros(1,number_of_runs);
two_pair=zeros(1,number_of_runs);
triple=zeros(1,number_of_runs);
full_house=zeros(1,number_of_runs);
four_of_a_kind=zeros(1,number_of_runs);
straight=zeros(1,number_of_runs);
flush=zeros(1,number_of_runs);
straight_and_royal_flush=zeros(1,number_of_runs);
royal_flush=zeros(1,number_of_runs);
high_card=zeros(1,number_of_runs);
count=0;

for n=1:number_of_runs
    deck=randperm(52);    % Function to shuffle the deck
    draw=deck(1:5);       % Draw first five cards from shuffled deck
```

```
    % For single pair, two pair, triple and high card
    for i=1:5
        for j=1:5
            for k=1:13
                if(i ~= j)
                    if((draw(i)==k && draw(j)==k+13) ||...
                            (draw(i)==k && draw(j)==k+26) ||...
                            (draw(i)==k && draw(j)==k+39) ||...
                            (draw(i)==k+13 && draw(j)==k+26) ||...
                            (draw(i)==k+13 && draw(j)==k+39) ||...
                            (draw(i)==k+26 && draw(j)==k+39))
                        count=count+1;
                    end
                end
            end
        end
    end
    if(count == 1)
        single_pair(n)=1;
    end
    if(count == 2)
        two_pair(n)=1;
    end
    if(count == 3)
        triple(n)=1;
    end
    if (count == 0)
        high_card(n)=1;
    end
    count=0;

    % For full house
    for i=1:5
        for j=1:5
            for l=1:5
                if(i~=j && j~=l && l~=i)
                    for k=1:13
                        if (draw(i)==k && draw(j)==k+13 && draw(l)==k+26
||...
                                draw(i)==k && draw(j)==k+13 &&
draw(l)==k+39 ||...
                                draw(i)==k+13 && draw(j)==k+26 &&
draw(l)==k+39)
                            sum1=15-(i+j+l);
                            for x=1:5
                                if(x~=i && x~=j && x~=k)
                                    y=sum1-x;
                                    if(y<5 ||y==5)
                                        break;
                                    end
                                end
                            end
                            for k=1:13
                                if((draw(x)==k && draw(y)==k+13) ||...
                                        (draw(x)==k && draw(y)==k+26) ||...
                                        (draw(x)==k && draw(y)==k+39) ||...
                                        (draw(x)==k+13 && draw(y)==k+26)
||...
                                        (draw(x)==k+13 && draw(y)==k+39)
||...
```

```
                                        (draw(x)==k+26 && draw(y)==k+39))
                                    full_house(n)=1;
                                end
                            end
                        end
                    end
                end
            end
        end
    end

    % Four of a kind
    for i=1:5
        for j=1:5
            for l=1:5
                for m=1:5
                    if(i~=j &&j~=l &&l~=m&& m~=i)
                        for k=1:13
                            if( draw(i)==k && draw(j) == k+13 &&
draw(l)==k+26 && draw(m)== k+39)
                                four_of_a_kind(n)=1;
                            end
                        end
                    end
                end
            end
        end
    end

    % Flush
    if((draw(1)<=13 && draw(2)<=13&& draw(3)<=13&& draw(4)<=13&&
draw(5)<=13) ||...
            (draw(1)>13 && draw(1)<=26 &&draw(2)>13 &&
draw(2)<=26&&draw(3)>13 &&...
            draw(3)<=26&&draw(4)>13 && draw(4)<=26&&draw(5)>13 &&
draw(5)<=26 )||...
            (draw(1)>26 && draw(1)<=39 &&draw(2)>26 &&
draw(2)<=39&&draw(3)>26 &&...
            draw(3)<=39&&draw(4)>26 && draw(4)<=39&&draw(5)>26 &&
draw(5)<=39) ||...
            (draw(1)>39 && draw(1)<=52 &&draw(2)>39 &&
draw(2)<=52&&draw(3)>39 &&...
            draw(3)<=52&&draw(4)>39 && draw(4)<=52&&draw(5)>39 &&
draw(5)<=52))
        flush(n)=1;
    end

    % Straight and royal flush
    for i=1:5
        for j=1:5
            for l=1:5
                for m=1:5
                    for p=1:5
                        if(i~=j &&j~=l &&l~=m&& m~=p &&p~=i)
                            for k=1:13
                                if( (draw(i)==k && draw(j) == k+1 &&
draw(l)==k+2 && draw(m)== k+3 &&draw(p)==k+4)||...
                                        (draw(i)==k+13 && draw(j) == k+14
&& draw(l)==k+15 && draw(m)== k+16 &&draw(p)==k+17)||...
                                        (draw(i)==k+26 && draw(j) == k+27
&& draw(l)==k+28 && draw(m)== k+29 &&draw(p)==k+30) ||...
```

```
                                        (draw(i)==k+39 && draw(j) == k+40
&& draw(l)==k+41 && draw(m)== k+42 &&draw(p)==k+43))
                                    straight_and_royal_flush(n)=1;
                                end
                            end
                        end
                    end
                end
            end
        end
    end
end
prob_single_pair=sum(single_pair)/number_of_runs
prob_two_pair=sum(two_pair)/number_of_runs
prob_triple=sum(triple)/number_of_runs
prob_full_house=sum(full_house)/number_of_runs
prob_four_of_a_kind=sum(four_of_a_kind)/number_of_runs
prob_high_card=sum(high_card)/number_of_runs
prob_flush=sum(flush)/number_of_runs
prob_straight_and_royal_flush=sum(straight_and_royal_flush)/number_of_runs
```

**Output:**

| Type of Hand | Theoretical Probability | Simulation Probability |
|---|---|---|
| Single pair | 0.4225 | 0.4233 |
| Two pair | 0.0475 | 0.0483 |
| Triple | 0.0211 | 0.0213 |
| Full house | 0.00144 | 0.00131 |
| Four of a kind | 0.00024 | 0.00022 |
| High card | 0.5011 | 0.5057 |
| Flush | 0.00196 | 0.0019 |
| Straight | 0.00392 | 0.0029 |
| Straight Flush and Royal flush | 0.0000153 | 0.00001 |

*Table 4 Comparison of Theoretical and Simulation values of probability of Poker hands*

From Table 4, we can see that the simulated probability of poker hands for 100000 runs is similar to the theoretical probability.

## References

[1] Svetlana Strbac-Savic; Ana Miletic; Hana Stefanovic, *“THE ESTIMATION OF PI USING MONTE CARLO TECHNIQUE WITH INTERACTIVE ANIMATIONS”*, 8th International Scientific Conference “Science and Higher Education in Function of Sustainable Development” 02-03 October 2015, Uzice, Serbia

[2] Sheldon M. Ross, *“Simulation”*