Assignment 2

Anika Tabassum
Student ID: 1505034
Course ID : CSE 411

<u>Monte Carlo Integration:</u> In mathematics, **Monte Carlo integration** is a technique for numerical integration using **random numbers**.

It is a particular Monte Carlo method that numerically computes a definite integral. While other algorithms usually evaluate the integrand at a regular grid, Monte Carlo randomly chooses points at which the integrand is evaluated. This method is particularly useful for higher-dimensional integrals.

We define Monte Carlo simulation to be a scheme employing random numbers, that is, U(O, 1) random variates, which is used for solving certain stochastic or deterministic problems where the passage of time plays no substantive role. Thus, Monte Carlo simulations are generally static rather than dynamic. The- name "Monte Carlo" simulation Or method originated during World war II , when this approach was applied to problems related to the development of the atomic bomb.

There are different methods to perform a Monte Carlo integration, such as uniform sampling, stratified sampling, importance sampling, sequential Monte Carlo (also known as a particle filter), and mean field particle methods.

$$I = \int_{limit} f(x)dx$$

Suppose that we want to evaluate the integral:

$$I = \int_a^b f(x)dx$$

where g(x) is a real-valued function that is not analytically integrable. (In practice, Monte Carlo simulation would probably not be used to evaluate a single integral, since there are more efficient numerical-analysis techniques for this purpose. It is more likely to be used on a multiple-integral problem with an ill-behaved integrand) To see how this deterministic problem can be approached by Monte Carlo simulation, let Y be the random variable (b - a)g(X), where X is a continuous random variable distributed uniformly on [a, b] [denoted by U(a, b)]. Then the expected value of Y is

$$\begin{aligned} E(Y) &= E[(b - a)g(X)] \\ &= (b - a)E[g(X)] \\ &= \int_a^b g(x)f_x(x)\, dx \end{aligned}$$

$$=(b-a)\ \{ \int_a^b g(x)\,dx\} / (b-a)$$

$$= I$$

Where $f_x(x) = 1/(b - a)$ is the probability density function of a U(a, b) random variable.Thus, the problem of evaluating the integral has been reduced to One of estimating the expected value E(Y). In particular, we shall estimate E(Y) = I by the sample mean

$$Y'(n) = (\sum_{i=1 \text{ to } n} Y_i) / n = (b-a) (\sum_{i=1 \text{ to } n} g(x_i)) / n$$

where $X_1, X_2 \ldots, X_n$ are IID U(a, b) random variables.

Part 1:

$Y'(n)$ for various values of n resulting from applying Monte Carlo simulation to the estimation of the integral $\int_0^{\pi/4} \sin x\, dx$

C code for Analytical Result :

```
float Analytical()
{
   float a;
   a = cos(0/4) - cos(M_PI/4);
   return a;
}
```

C code for Monte Carlo Result :

```
float Monte_Carlo()
{
   float sum = 0;
   int i;
   int j;
   float x;
   int Values_n[] = {10,20,30,40,50,100};
   int n;
   n = sizeof Values_n / sizeof Values_n[0]; //length of Values_n array
   float Y_n[n];  // Array for saving Monte Carlo result
   float div[n];  // Array for saving Diversion from analytical result
   for(i=0;i<n;i++)
   {
      sum = 0;
      for(j=1;j<(Values_n[i]+1);j++)
      {
         // Uniform random number within range 0 to pi/4
         x=((long double)rand()/RAND_MAX)*(M_PI/4-0) + 0;
         sum += sin(x);
      }
      // calculation Monte Carlo result
```

```
        float y = (M_PI/4 - 0)*sum/Values_n[i];
        Y_n[i]=y;
        printf("\n %f ",y);
        float analytical_result = Analytical();
        div[i]= analytical_result - y ;
        printf("Diversion from analytical result %f ",div[i]);

    }
}
```

Overall C code :

```
#include<stdio.h>
#include<math.h>
#include <stdlib.h>

float Analytical()
{
    float a;
    a = cos(0/4) - cos(M_PI/4);
```

```
    return a;

}
float Monte_Carlo()
{
    float sum = 0;
    int i;
    int j;
    float x;
    int Values_n[] = {10,20,30,40,50,100};
    int n;
    n = sizeof Values_n / sizeof Values_n[0]; //length of Values_n array
    float Y_n[n];  // Array for saving Monte Carlo result
    float div[n];  // Array for saving Diversion from analytical result
    for(i=0;i<n;i++)
    {
        sum = 0;
        for(j=1;j<(Values_n[i]+1);j++)
        {
            // Uniform random number within range 0 to pi/4
            x=((long double)rand()/RAND_MAX)*(M_PI/4-0) + 0;
            sum += sin(x);
        }
        // calculation Monte Carlo result
```

```
        float y = (M_PI/4 - 0)*sum/Values_n[i];
        Y_n[i]=y;
        printf("\n %f ",y);
        float analytical_result = Analytical();
        div[i]= analytical_result - y ;
        printf("Diversion from analytical result %f ",div[i]);

    }
}

int main()
{
    float a;
    a=Analytical();
    printf("Analytical Result  %f",a);
    Monte_Carlo();
}
```

Result Obtained :

Analytical Result : 0.292893

Table: Monte Carlo result with diversion from analytical result:

Value of n	Monte Carlo Result	Diversion from analytical result
10	0.318842	-0.025949
20	0.224790	0.068103
30	0.323645	-0.030752
40	0.290609	0.002284
50	0.302656	-0.009763
100	0.288978	0.003915

Part 2:

For Part 2 , the parameter average rate λ=1

I have to generate the probabilities for different ranges in a probability table.

C code for Monte Carlo Result :

```
void Monte_Carlo()
{
    float sum = 0;
    int i;
    int j;
    float x;
    int n = 10000;
    //float Y_n[100][10];
    float div[n];
    float X = 10;
    float a = 0.00;
    float p = 0.0;

    // For file output
    FILE *fptr;
    fptr = fopen("program.txt", "w");

    if (fptr == NULL)
    {
        printf("Error!");
        exit(1);
```

```
    }
    fprintf(fptr, "      ");
    for(i=0; i<10; i++)
    {

        printf(" %f  ",p);
        fprintf(fptr, " %.2lf ", p);
        p += .01;
    }
    fprintf(fptr, "\n");
    while(a <= X)   // For 0.0 to 9.9
    {
        p = 0.0;
        printf(" %.2lf  ",a);
        fprintf(fptr, " %.2lf ", a);
        for(i=0; i<10; i++)  // For 0.00 to 0.09
        {
            sum = 0;

            for(j=1; j<(n+1); j++)
            {
                // Uniform random number within range 0 to higher limit each
time
```

```
            x=((long double)rand()/RAND_MAX)*(a+p-0) + 0;  // Higher
limit a+p Lower limit 0
            sum += (exp(-x));
         }
         // calculation Monte Carlo result
         float y = (a+p - 0)*sum/n;
         //Y_n[i]=y;
         //printf(" %f  ",y);
         fprintf(fptr, " %.2lf ", y);
         p += .01;  // Increasing by .01
      }
      printf("\n");
      fprintf(fptr, "\n");
      a += .1; //Increasing by .1


   }
   fclose(fptr);
}
```

Overall C code :

```
#include<stdio.h>
#include<math.h>
#include <stdlib.h>

float Analytical()
{
    float a;
    a = -exp(-9.99) - (-exp(-0));
    return a;

}
void Monte_Carlo()
{
    float sum = 0;
    int i;
    int j;
    float x;
    int n = 10000;
    //float Y_n[100][10];
    float div[n];
    float X = 10;
```

```
float a = 0.00;
float p = 0.0;

// For file output
FILE *fptr;
fptr = fopen("program.txt", "w");

if (fptr == NULL)
{
   printf("Error!");
   exit(1);
}
fprintf(fptr, "      ");
for(i=0; i<10; i++)
{

   printf(" %f  ",p);
   fprintf(fptr, " %.2lf ", p);
   p += .01;
}
fprintf(fptr, "\n");
while(a <= X)   // For 0.0 to 9.9
{
   p = 0.0;
```

```
        printf(" %.2lf  ",a);
        fprintf(fptr, " %.2lf ", a);
        for(i=0; i<10; i++)  // For 0.00 to 0.09
        {
            sum = 0;

            for(j=1; j<(n+1); j++)
            {
                // Uniform random number within range 0 to higher limit each
time
                x=((long double)rand()/RAND_MAX)*(a+p-0) + 0;  // Higher
limit a+p Lower limit 0
                sum += (exp(-x));
            }
            // calculation Monte Carlo result
            float y = (a+p - 0)*sum/n;
            //Y_n[i]=y;
            //printf(" %f  ",y);
            fprintf(fptr, " %.2lf ", y);
            p += .01;  // Increasing by .01
        }
        printf("\n");
        fprintf(fptr, "\n");
        a += .1; //Increasing by .1
```

```
    }
    fclose(fptr);
}
int main()
{
    float a;
    a= Analytical();
    printf("Analytical Result %f",a);
    Monte_Carlo();
}
```

Result Obtained :

Analytical Result : 0.999954

Monte Carlo Result :

	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.00	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.10	0.10	0.10	0.11	0.12	0.13	0.14	0.15	0.16	0.16	0.17
0.20	0.18	0.19	0.20	0.21	0.21	0.22	0.23	0.24	0.24	0.25
0.30	0.26	0.27	0.27	0.28	0.29	0.30	0.30	0.31	0.32	0.32
0.40	0.33	0.34	0.34	0.35	0.36	0.36	0.37	0.37	0.38	0.39
0.50	0.39	0.40	0.41	0.41	0.42	0.42	0.43	0.43	0.44	0.45
0.60	0.45	0.46	0.46	0.47	0.47	0.48	0.49	0.49	0.49	0.50
0.70	0.50	0.51	0.51	0.52	0.52	0.53	0.53	0.54	0.54	0.55
0.80	0.55	0.56	0.56	0.56	0.57	0.57	0.58	0.58	0.58	0.59
0.90	0.59	0.60	0.60	0.61	0.61	0.61	0.62	0.62	0.62	0.63
1.00	0.64	0.63	0.64	0.65	0.65	0.65	0.65	0.66	0.66	0.66
1.10	0.66	0.67	0.68	0.68	0.68	0.68	0.69	0.69	0.69	0.69
1.20	0.70	0.71	0.71	0.71	0.71	0.71	0.72	0.72	0.72	0.73
1.30	0.73	0.73	0.73	0.74	0.73	0.73	0.74	0.75	0.75	0.75
1.40	0.75	0.76	0.75	0.75	0.76	0.77	0.77	0.77	0.77	0.77
1.50	0.78	0.78	0.78	0.79	0.78	0.79	0.79	0.79	0.79	0.79
1.60	0.80	0.80	0.80	0.80	0.81	0.81	0.81	0.82	0.82	0.81
1.70	0.81	0.82	0.82	0.82	0.83	0.83	0.83	0.83	0.83	0.83
1.80	0.84	0.84	0.83	0.83	0.84	0.84	0.84	0.85	0.85	0.85
1.90	0.86	0.85	0.85	0.85	0.85	0.86	0.85	0.86	0.86	0.86
2.00	0.86	0.86	0.88	0.87	0.87	0.88	0.87	0.88	0.88	0.88
2.10	0.87	0.87	0.88	0.88	0.88	0.88	0.88	0.89	0.89	0.89
2.20	0.90	0.89	0.89	0.90	0.90	0.89	0.90	0.91	0.90	0.90
2.30	0.90	0.90	0.90	0.90	0.90	0.91	0.91	0.91	0.91	0.90
2.40	0.91	0.91	0.90	0.91	0.92	0.91	0.91	0.91	0.91	0.92
2.50	0.92	0.92	0.91	0.92	0.92	0.92	0.92	0.93	0.93	0.92
2.60	0.93	0.93	0.93	0.92	0.93	0.93	0.93	0.94	0.94	0.94

	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
2.70	0.93	0.94	0.92	0.93	0.93	0.94	0.93	0.94	0.94	0.94
2.80	0.94	0.94	0.94	0.94	0.92	0.95	0.95	0.95	0.94	0.94
2.90	0.94	0.94	0.95	0.94	0.95	0.94	0.95	0.92	0.95	0.94
3.00	0.96	0.96	0.94	0.95	0.94	0.95	0.95	0.95	0.95	0.96
3.10	0.96	0.95	0.95	0.96	0.95	0.95	0.96	0.96	0.97	0.96
3.20	0.96	0.95	0.95	0.95	0.96	0.96	0.96	0.97	0.98	0.98
3.30	0.96	0.96	0.98	0.97	0.96	0.96	0.96	0.96	0.97	0.96
3.40	0.98	0.97	0.96	0.98	0.98	0.98	0.97	0.98	0.98	0.97
3.50	0.97	0.98	0.97	0.97	0.97	0.97	0.96	0.97	0.97	0.97
3.60	0.98	0.97	0.99	0.98	0.98	0.96	0.97	0.97	0.97	0.98
3.70	0.97	0.97	0.96	0.96	0.97	0.98	0.98	0.98	0.98	0.99
3.80	0.97	0.97	0.97	0.99	0.97	0.98	0.97	0.98	0.99	0.98
3.90	0.97	0.97	0.98	0.99	0.99	0.97	0.96	0.99	0.98	0.99
4.00	1.00	0.98	0.99	0.98	0.99	1.00	0.96	0.99	0.99	0.98
4.10	0.98	1.00	0.98	0.97	0.99	1.00	1.00	0.99	0.99	0.98
4.20	0.98	0.98	0.98	1.00	1.00	0.99	1.00	0.99	1.00	0.98
4.30	0.99	0.99	0.99	1.00	0.99	1.01	0.99	1.01	1.00	0.98
4.40	0.98	0.99	1.00	0.99	1.01	0.97	0.97	0.99	1.00	0.99
4.50	0.98	0.98	0.97	0.98	0.98	1.00	0.99	0.97	0.95	0.99
4.60	0.98	1.01	0.99	0.99	1.00	0.99	0.98	0.97	1.00	0.99
4.70	1.01	1.00	0.98	0.99	1.01	1.00	0.99	0.97	0.99	1.01
4.80	0.98	0.99	1.02	1.00	0.98	0.97	0.98	1.01	1.00	0.98
4.90	0.98	1.00	1.01	0.97	0.97	1.01	0.98	1.00	1.01	0.99
5.00	1.02	0.97	1.00	0.99	1.00	1.00	1.00	0.99	0.98	1.00
5.10	0.99	0.98	0.99	1.00	1.02	1.00	1.02	0.99	0.98	1.03
5.20	0.99	1.00	1.00	1.00	1.02	1.00	1.00	0.99	1.00	1.02
5.30	0.99	1.01	1.01	1.00	1.00	1.00	1.00	1.00	0.98	0.98
5.40	0.99	0.98	1.01	0.97	0.99	0.97	0.99	0.99	0.99	0.99

	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
5.50	1.01	1.01	0.99	0.99	0.99	0.99	0.99	1.01	0.98	1.00
5.60	1.01	1.02	0.98	1.00	1.00	0.98	0.99	1.00	0.99	1.00
5.70	1.00	0.99	0.99	1.00	0.99	0.99	0.99	1.01	1.01	0.99
5.80	1.02	1.01	1.02	1.00	0.99	0.97	1.00	1.01	0.99	0.99
5.90	0.99	0.98	1.00	0.99	0.99	1.00	1.01	1.00	1.01	1.00
6.00	1.01	1.00	0.98	1.01	0.96	1.01	0.99	1.01	1.00	1.00
6.10	1.01	0.96	0.98	0.99	1.00	1.01	0.99	1.01	1.04	0.98
6.20	0.98	1.01	1.02	0.99	1.01	1.01	1.01	1.03	1.00	0.98
6.30	0.98	0.98	0.99	0.98	0.99	1.03	1.00	1.00	0.99	1.02
6.40	0.99	1.03	0.99	1.01	0.98	1.01	0.99	1.01	0.96	1.00
6.50	0.99	1.02	0.98	1.00	1.02	1.00	0.99	1.00	0.98	0.99
6.60	0.98	1.00	1.01	0.99	0.99	1.01	0.99	1.00	0.97	1.00
6.70	1.01	1.02	1.01	1.00	1.00	1.01	1.00	0.99	1.00	1.00
6.80	1.01	0.98	1.00	0.99	1.01	1.03	0.99	1.00	1.01	0.98
6.90	0.98	0.98	0.99	0.98	1.01	1.01	1.00	1.02	1.00	0.99
7.00	0.99	1.03	1.00	1.01	1.02	1.01	1.00	1.01	1.00	1.00
7.10	0.99	1.00	1.00	0.99	1.03	0.99	1.00	0.97	0.99	1.02
7.20	1.04	1.00	1.00	1.00	1.02	0.98	0.99	1.00	0.96	0.99
7.30	1.04	1.01	0.97	1.02	1.01	1.00	1.00	0.98	1.00	0.99
7.40	0.99	1.02	1.02	1.00	1.01	0.98	0.97	1.01	1.02	0.97
7.50	1.00	1.01	1.04	1.00	1.01	1.02	1.03	1.00	0.99	1.01
7.60	0.98	1.00	0.99	1.02	0.96	1.00	0.99	1.01	1.00	1.02
7.70	1.00	0.97	0.97	0.96	0.99	1.02	0.98	1.01	1.05	1.00
7.80	0.98	1.03	0.99	0.99	1.00	1.01	1.02	1.00	1.01	0.98
7.90	1.02	0.97	0.99	1.02	0.99	1.02	1.00	1.01	1.00	1.03
8.00	1.03	0.98	0.97	1.02	1.03	1.02	1.03	1.01	0.99	1.00
8.10	1.02	0.99	1.02	1.00	1.01	0.99	1.01	1.00	1.00	1.00
8.20	0.99	1.00	1.00	0.99	0.99	1.00	0.98	1.01	1.02	1.02

	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
8.30	0.99	0.98	1.00	1.01	1.00	1.00	1.04	1.00	1.01	0.99
8.40	0.98	0.98	0.99	1.01	1.00	1.03	0.98	1.00	1.01	1.01
8.50	0.98	0.99	1.01	1.00	1.01	1.02	0.98	1.02	0.99	1.00
8.60	1.01	0.99	1.03	1.01	0.98	0.99	1.05	0.99	1.00	1.00
8.70	1.01	1.01	0.99	1.01	0.96	1.01	1.00	1.04	1.00	1.01
8.80	0.99	1.01	1.00	1.00	0.98	1.01	1.03	1.00	1.00	1.01
8.90	1.00	0.99	0.99	0.99	0.99	0.99	0.98	1.01	1.02	0.99
9.00	1.01	0.98	1.02	1.00	1.02	1.02	1.00	0.99	0.98	1.00
9.10	0.99	1.01	1.03	1.01	1.01	0.99	1.00	0.99	0.98	1.03
9.20	0.99	1.01	1.02	0.98	1.00	1.01	0.99	1.01	0.99	0.99
9.30	1.05	1.02	1.02	0.99	0.97	1.02	0.99	1.00	1.01	1.01
9.40	1.01	1.01	0.98	0.99	1.02	1.01	1.00	1.01	0.99	1.00
9.50	1.00	0.99	1.02	0.98	0.98	0.99	1.00	1.00	1.01	0.99
9.60	0.97	0.99	0.98	0.99	1.00	1.02	0.98	1.02	0.99	1.00
9.70	1.00	0.97	1.02	1.01	1.01	1.02	1.02	0.98	1.00	0.98
9.80	1.00	1.02	1.01	1.02	1.00	1.04	0.96	1.00	0.99	0.99
9.90	1.02	0.97	0.99	1.05	1.00	1.01	1.00	1.03	0.99	1.01