CSE – 411
Student ID: 1505112

Monte Carlo Simulation: Monte Carlo Simulation is a mathematical technique that generates random variables, which is used for solving certain stochastic or deterministic problems. It can be used to calculate integration of a function within a range.

Let, g(x) is a function and we want to integrate within [a, b]. Integration:

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

Let, *X* is a continuous random variable distributed uniformly on [*a*, *b*] which is denoted by U(*a*, *b*) and *Y* be the random variable (*b* - *a*)**g*(*X*). Then the expected value of *Y* is

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

Where $f(x) = 1/(b-a)$ 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

$$\bar{Y} = \frac{\sum_{i=1}^{n} Y_i}{n} = (b-a)\frac{\sum_{i=1}^{n} g(X_i)}{n}$$

Where X_1, X_2 to X_n are IID U(*a*, *b*) random variables.

Here g(x) = sin(x) and a = 0, b = π/4.

$$A = \int_0^{\pi/4} sin(x)dx$$

Analytical Result: We know,

$$\int \sin(x)\,dx \;=\; -\cos(x)$$

So,

$$A = \int_0^{\pi/4} sin(x)dx$$
$$= -\cos\left(\frac{\pi}{4}\right) + \cos(0)$$
$$= -\frac{1}{\sqrt{2}} + 1$$
$$= 0.292895$$

Simulated Result: Monte Carlo Simulation result for different number of random numbers are shown below. PI = 3.1416.

n	10	100	1000	10000	100000	1000000
$\bar{Y}(n)$	0.307846	0.262520	0.289287	0.293904	0.293729	0.292861

Simulated result may be near to analytical result if we increase number of random numbers.

Probability Table:

Exponential Distribution: The parameter average rate λ=1.

Monte Carlo simulation is used to calculate probability distribution of exponential distribution. For simulation, n = 1000 random numbers are used to calculate integration.

	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.0000	0.0100	0.0198	0.0296	0.0392	0.0488	0.0582	0.0677	0.0768	0.0861
0.1	0.0951	0.1042	0.1129	0.1222	0.1307	0.1394	0.1477	0.1560	0.1648	0.1739
0.2	0.1817	0.1895	0.1971	0.2053	0.2131	0.2208	0.2296	0.2371	0.2438	0.2511
0.3	0.2591	0.2676	0.2744	0.2806	0.2890	0.2947	0.3034	0.3097	0.3167	0.3240
0.4	0.3302	0.3340	0.3439	0.3497	0.3557	0.3642	0.3697	0.3750	0.3831	0.3899
0.5	0.3942	0.3977	0.4058	0.4140	0.4176	0.4232	0.4265	0.4396	0.4410	0.4494
0.6	0.4511	0.4596	0.4601	0.4667	0.4761	0.4774	0.4828	0.4886	0.4917	0.5001
0.7	0.5020	0.5149	0.5181	0.5163	0.5260	0.5332	0.5362	0.5366	0.5395	0.5430
0.8	0.5516	0.5479	0.5599	0.5598	0.5737	0.5677	0.5724	0.5767	0.5868	0.5828
0.9	0.5959	0.5936	0.5985	0.6060	0.6037	0.6173	0.6086	0.6128	0.6361	0.6261
1.0	0.6236	0.6255	0.6437	0.6460	0.6410	0.6588	0.6483	0.6585	0.6589	0.6576
1.1	0.6612	0.6605	0.6627	0.6904	0.6799	0.6828	0.6898	0.6802	0.6976	0.6720
1.2	0.6967	0.6903	0.6906	0.7102	0.7102	0.7235	0.7212	0.7220	0.7130	0.7284
1.3	0.7315	0.7157	0.7239	0.7397	0.7442	0.7551	0.7544	0.7385	0.7554	0.7463
1.4	0.7623	0.7530	0.7630	0.7588	0.7661	0.7774	0.7750	0.7603	0.7808	0.7928
1.5	0.7802	0.7891	0.7841	0.7799	0.7930	0.7891	0.7948	0.7852	0.8057	0.7972
1.6	0.8170	0.7949	0.8023	0.7845	0.8059	0.7981	0.8018	0.7904	0.8165	0.8196
1.7	0.7920	0.8216	0.8197	0.8332	0.8421	0.8198	0.8078	0.8391	0.8244	0.8305
1.8	0.8063	0.8453	0.8225	0.8402	0.8667	0.8290	0.8285	0.8309	0.8583	0.8427
1.9	0.8410	0.8762	0.8411	0.8621	0.8736	0.8421	0.8630	0.8371	0.8537	0.8803
2.0	0.8669	0.8729	0.8699	0.8583	0.8627	0.8804	0.8608	0.8703	0.8726	0.8961
2.1	0.9263	0.8780	0.8708	0.8748	0.8835	0.8896	0.8664	0.8686	0.8646	0.8832
2.2	0.8763	0.9088	0.9112	0.8906	0.8821	0.9065	0.9270	0.9111	0.9018	0.8765

2.3	0.9226	0.9063	0.9143	0.9000	0.8747	0.9127	0.9021	0.9275	0.9361	0.9146
2.4	0.9117	0.9150	0.9217	0.8758	0.8784	0.9249	0.8992	0.8728	0.9061	0.8669
2.5	0.9141	0.8961	0.9148	0.9405	0.9470	0.9017	0.9461	0.9305	0.9332	0.9314
2.6	0.9189	0.9497	0.8960	0.9140	0.9149	0.9157	0.9464	0.9498	0.8836	0.9147
2.7	0.9495	0.9120	0.9243	0.8900	0.9284	0.9604	0.9461	0.9345	0.9394	0.9001
2.8	0.9168	0.9352	0.9755	0.9553	0.9608	0.9364	0.9629	0.9459	0.9013	0.9418
2.9	0.9957	0.9175	0.9596	1.0021	0.9086	0.9255	0.9481	0.9760	0.9027	0.9957
3.0	0.9692	0.9454	0.9643	0.9554	0.9793	0.9260	0.9489	0.9375	0.9626	0.9338
3.1	0.9305	0.9604	0.9932	0.9391	0.9556	0.9990	0.9626	0.9409	0.9745	0.9575
3.2	0.9717	0.9602	0.9493	0.9719	0.9303	0.9726	0.9942	0.9553	0.9675	1.0127
3.3	1.0078	0.9636	0.9754	0.9714	0.9967	0.9913	0.9565	0.9674	0.9626	0.9749
3.4	0.9301	0.9575	0.9489	0.9526	0.9614	0.9850	0.9043	0.9998	0.9709	0.9657
3.5	0.9755	1.0019	0.9479	0.9696	0.9957	0.9492	0.9681	0.9673	0.9344	0.9762
3.6	1.0153	0.9654	0.9581	1.0026	1.0074	0.8735	0.9391	0.9448	0.9428	0.9738
3.7	0.9783	0.9519	0.9697	0.9175	0.9426	0.9861	0.9364	0.9395	0.9749	0.9399
3.8	0.9600	0.9921	0.9859	0.9421	1.0095	0.9732	0.9752	0.9633	0.9974	0.9484
3.9	1.0212	0.9679	0.9711	0.9667	0.9657	0.9798	0.9626	0.9916	0.9962	1.0186
4.0	1.0199	1.0024	0.9813	1.0264	0.9773	1.0020	0.9322	0.9415	1.0138	1.0018
4.1	0.9711	0.9998	0.9912	1.0121	0.9470	0.9781	1.0108	1.0475	0.9397	0.9618
4.2	0.9666	0.9871	0.9772	0.9888	0.9951	0.9892	1.0240	0.9617	0.9640	0.9679
4.3	1.0293	1.0624	1.0117	0.9899	1.0379	0.9745	1.0354	0.9359	1.0005	0.9865
4.4	0.9961	0.9906	1.0510	0.9317	0.9505	0.9456	0.9516	1.0225	0.9827	0.9922
4.5	0.9176	0.9626	1.0331	1.0232	0.9120	1.0383	0.9759	1.0406	0.9600	1.0071
4.6	1.0260	1.0006	1.0222	0.9594	1.0379	1.0039	1.0223	0.9982	0.9593	0.9595
4.7	0.9565	1.0024	0.9908	1.0076	1.0311	0.9763	1.0294	1.0210	0.9921	0.9517
4.8	0.9531	0.9510	0.9325	0.9794	1.0069	0.9551	0.9608	1.0065	1.0318	0.9169
4.9	1.0268	1.0001	0.9704	1.0213	0.9869	0.9742	0.9955	1.0579	0.9520	0.9589

Code:

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

#define PI 3.1416

// monte carlo simulation for sin(x) for n random numbers
double monte_carlo_sim(float a, float b, int n){
    double result, rand_x, sum = 0.0;
    int i, rd;
    for(i = 0; i < n; i++){
        rd = rand();
        // random number X within a to b
        rand_x = a+(b - a)*rd/RAND_MAX;
        sum += sin(rand_x);
    }
    result = (b - a)*sum/n;
    return result;
}
```

```
// monte carlo simulation for exp(x) for n random numbers
double monte_carlo_sim_exp(float a, float b, float lemda, int n){
    double result, rand_x, sum = 0.0;
    int i, rd;
    for(i = 0; i < n; i++){
        rd = rand();
        // random number X within a to b
        rand_x = a+(b - a)*rd/RAND_MAX;
        // f(x) = lemda*exp(-lemda*x)
        sum += lemda*exp(-(lemda*rand_x));
    }
    result = (b - a)*sum/n;
    return result;
}

void integration(){
    // integration of six(x) from 0 to PI/4
    double a = 0;
    double b = PI/4;
    // integral(sin(x)) = -cos(x);
    double analytic_result = cos(a) - cos(b);
    printf("Analytic result: %f\n", analytic_result);

    // monte carlo simulation
    int N = 6; // number of simulation run
    int MAX_RUN = 10;
    int n = 10; // initial number of IID random number
    double simulated_result[MAX_RUN];
    int num_random[MAX_RUN], i;

    for(i = 0; i < N; i++){
        simulated_result[i] = monte_carlo_sim(a, b, n);
        //printf("%f\n", simulated_result[i] - analytic_result);
        num_random[i] = n;
        n = n*10;
    }
    // print result
    printf("\nSimulated result:\n");
    printf("Random Number(N) : Estimated Integral\n");
    for(i = 0; i < N; i++){
        printf("%14d   :  %f\n", num_random[i], simulated_result[i]);
    }
}
```

```c
void probability_table(){
    // open file to write probability table
    FILE *file = fopen("exp_prob_table.txt", "w");
    if(file == NULL){
        printf("File can not be opened");
        return;
    }
    // probability table from 0.00 to N-0.01
    int N = 5;
    double table[N*10][10];
    // F(x) = 1 - exp(-lemda*x)
    float lemda = 1;
    int i, j, k;
    double x, frac1, frac2;
    for(i = 0; i<N; i++){
        frac1 = 0.0;
        for(j = 0; j<10; j++){
            frac2 = 0.00;
            for(k = 0; k<10; k++){
                x = i + frac1 + frac2;
                //table[10*i+j][k] = 1 - exp(-(lemda*x)); // analytic
                table[10*i+j][k] = monte_carlo_sim_exp(0, x, lemda, 1000);
                frac2 += 0.01;
            }
            frac1 += 0.1;
        }
    }

    // output kept in file
    fprintf(file, "      ");
    frac1 = 0.00;
    for(i = 0; i<10; i++){
        fprintf(file, "%.2f     ", frac1);
        frac1 += 0.01;
    }
    fprintf(file, "\n");
    frac1 = 0.0;
    for(i = 0; i<N; i++){
        for(j = 0; j<10; j++){
            fprintf(file, "%.1f   ", frac1);
            for(k = 0; k<10; k++){
                fprintf(file, "%.4f   ", table[10*i+j][k]);
            }
            fprintf(file, "\n");
            frac1 += 0.1;
        }
    }
    fprintf(file, "\n");
}
int main(){
    srand(time(0));
    // integration of six(x) from 0 to PI/4
    integration();
    // probability table
    probability_table();

    return 0;
}
```