Chapter 8

Monte Carlo Simulation

8.1 Introduction

Monte Carlo simulation is named after the city of Monte Carlo in Monaco, which is famous for gambling such as roulette, dice, and slot machines. Since the simulation process involves generating chance variables and exhibits random behaviors, it has been called Monte Carlo simulation. Monte Carlo simulation is a powerful statistical analysis tool and widely used in both non-engineering fields and engineering fields. It was initially used to solve neutron diffusion problems in atomic bomb work at Alamos Scientific Laboratory in 1944. Monte Carlo simulation has been applied to diverse problems ranging from the simulation of complex physical phenomena such as atom collisions to the simulation of traffic flow and Dow Jones forecasting. Monte Carlo is also suitable for solving complex engineering problems because it can deal with a large number of random variables, various distribution types, and highly nonlinear engineering models.

Different from a physical experiment, Monte Carlo simulation performs random sampling and conducts a large number of experiments on computer. Then the statistical characteristics of the experiments (model outputs) are observed, and conclusions on the model outputs are drawn based on the statistical experiments. In each experiment, the possible values of the input random variables $\mathbf{X} = (X_1, X_2, \cdots, X_n)$ are sampled (generated) according to their distributions. Then the values of the output variable Y are calculated through the performance function $Y = g(\mathbf{X})$ at the samples of input random variables. With a number of experiments carried out in this manner, a set of samples of output variable Y are available for the statistical analysis, which estimates the characteristics of the output variable Y.

The outline of Monte Carlo simulation is depicted in Fig. 9.1. Three steps are required in the simulation process: Step 1 – sampling on random input variables $\mathbf{X}$, Step 2 – evaluating model output Y, and Step 3 – statistical analysis on model output.

We will focus our discussions on independent random variables. However, Monte Carlo simulation is applicable for dependent variables. The three steps of Monte Carlo simulation are discussed in the following sections.

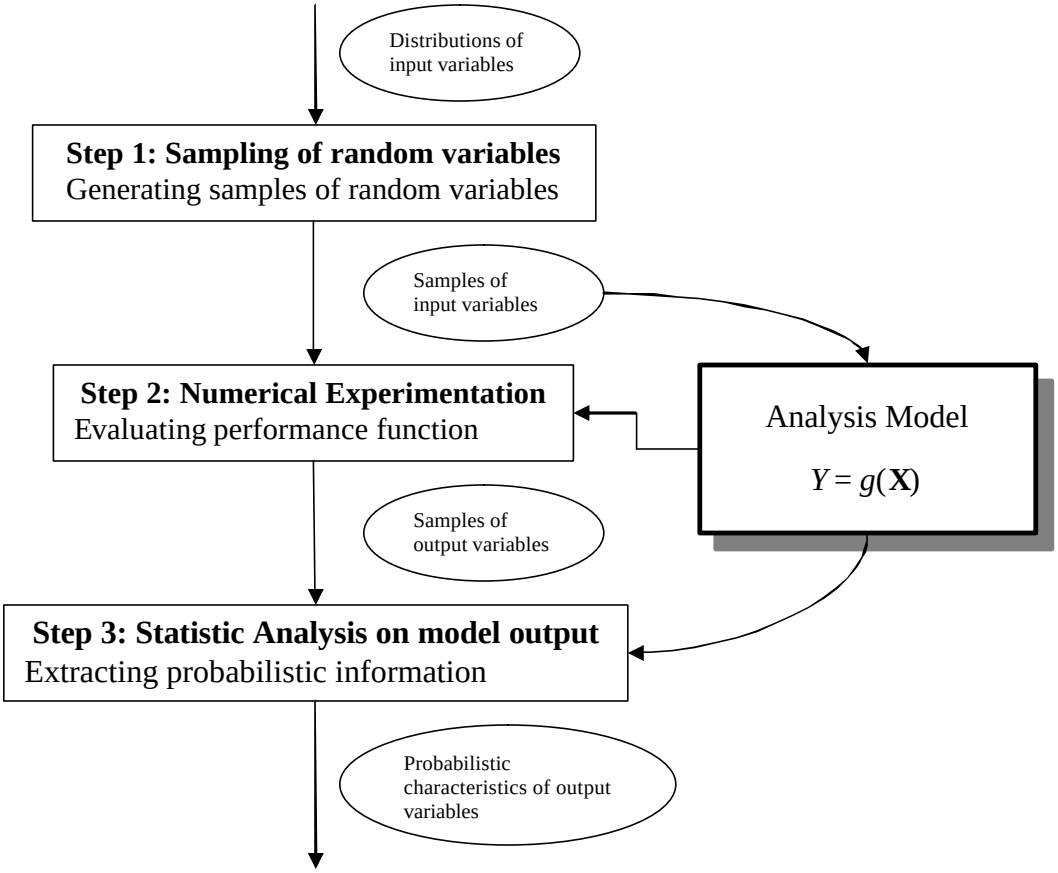


Figure 9.1 Monte Carlo Simulation

8.2 Sampling on input random variables

The purpose of sampling on the input random variables $\mathbf{X} = (X_1, X_2, \cdots, X_n)$ is to generate samples that represent distributions of the input variable from their *cdf*s $F_{X_i}(x_i)$ $(i = 1,2,\cdots,n)$. The samples of the random variables will then be used as inputs to the simulation experiments. Two steps are involved for this purpose: Step 1 – generating random variables that are uniformly distributed between 0 and 1, and Step 2 – transforming the values of the uniform variable obtained from Step 1 to the values of random variables that follow the given distributions $F_{X_i}(x_i)$ $(i = 1,2,\cdots,n)$.

Step 1 – Generating random variables that are uniformly distributed between 0 and 1

The importance of uniform numbers over the continuous range [0, 1] is that they can be transformed into real values that follow any distributions of interest. In the early times of simulation, random numbers were generated by mechanical ways, such as drawing balls, throwing dice, as the same way as many of today's lottery drawings. Now any modern computers have the capability to generate uniformly distributed random variables

between 0 and 1. There are a number of arithmetic random-generators developed for the computer-based random generation. Random variables generated this way are called *pseudo random numbers*.

A random-generator produces a sequence of uniform numbers between 0 and 1. The length of the sequence before repeating itself is machine and algorithm dependent. The following 20 uniform random variables in the interval of [0, 1] are generated with the MATLAB random variable generator *rand*.

Table 9.1 [0, 1] Uniform Random Variables

0.8381	0.6813	0.8318	0.7095	0.3046	0.1934	0.3028	0.1509	0.3784	0.8537
0.0196	0.3795	0.5028	0.4289	0.1897	0.6822	0.5417	0.6979	0.8600	0.5936

Step 2 – Transforming [0, 1] uniform variables into random variables that follow the given distributions

The task is to transform the samples of [0, 1] uniform variable, $\mathbf{z} = (z_1, z_2, \cdots, z_N)$, where N is the number of samples, generated from Step 1, into values of random variable X_i that follows a given distribution $F_{X_i}(x_i)$. There are several methods for such a transformation. The simple and direct transformation is the inverse transformation method. By this method, the random variable is given by

$$x_i = F_{X_i}^{-1}(z_i),\ i = 1, 2, \cdots, N \tag{9.1}$$

where $F_{X_i}^{-1}$ is the inverse of the *cdf* of the random variable X_i.

The transformation is demonstrated in Fig. 9.2.

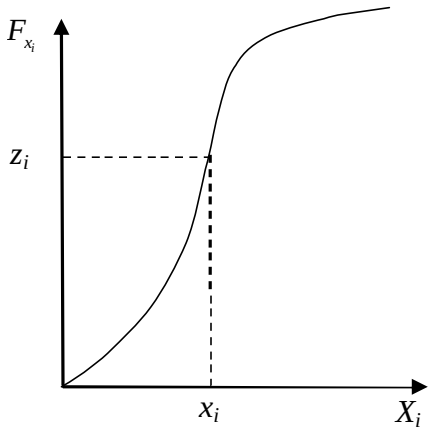


Figure 9.2 The Inverse Transformation Method

For example, if X is normally distributed with $N(m_X,\ s_X)$, since

$$z = F_X(x) = \Phi\left(\frac{x - m_X}{s_X}\right) \tag{9.2}$$

then

$$x = m_X + s_X \Phi^{-1}(z) \tag{9.3}$$

8.3 Numerical Experimentation

Suppose that N samples of each random variable are generated, then all the samples of random variables constitute N sets of inputs, $\mathbf{x}_i = (x_{i1}, x_{i2}, \cdots, x_{in})$, $i = 1,2,\cdots,N$, to the model $Y = g(\mathbf{X})$. Solving the problem N times deterministically yields N sample points of the output Y.

$$y_i = g(\mathbf{x}_i),\ i = 1,2,\cdots N \tag{9.4}$$

8.4 Extraction of probabilistic information of output variables

After N samples of output Y have been obtained, statistical analysis can be carried out to estimate the characteristics of the output Y, such as the mean, variance, reliability, the probability of failure, *pdf* and *cdf*. The associated equations are given below:

The mean

$$\overline{Y} = \frac{1}{N}\sum_{i=1}^{N} y_i \tag{9.5}$$

The variance

$$s_Y^2 = \frac{1}{N-1}\sum_{i=1}^{N} (y_i - \overline{Y})^2 \tag{9.6}$$

If the failure is defined by the event $g \le 0$, the probability of failure is then calculated by

(see Eq. 6.9)

$$p_f = \mathrm{P}\{g \le 0\} = \int\cdots\int_{g(\mathbf{x})\le 0} f_{X_1,X_2,\cdots,X_n}(x_1, x_2, \cdots, x_n)\,dx_1 dx_2 \cdots dx_n = \int_{g(\mathbf{x})\le 0} f_{\mathbf{X}}(\mathbf{x})\,d\mathbf{x} \tag{9.7}$$

where $\mathbf{X} = (X_1, X_2, \cdots, X_n)$ and $\mathbf{x}=(x_1, x_2, \cdots, x_n)$.

The equation can be rewritten as

$$p_f = \int_{-\infty}^{+\infty} I(\mathbf{x}) f_X(\mathbf{x}) d\mathbf{x} \tag{9.8}$$

where $I(\cdot)$ is an indicator function, which is defined by

$$I(\mathbf{x}) = \begin{cases} 1 & \text{if } g(\mathbf{x}) \le 0 \\ 0 & \text{otherwise} \end{cases} \tag{9.9}$$

According to Eq. 3.9, the integral on the right-hand side of Eq. 9.8 is simply the expected value (or average) of $I(\mathbf{x})$. Therefore, p_f can be estimated by the average value of $I(\mathbf{x})$ as

$$p_f = \bar{I}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} I(\mathbf{x}_i) = \frac{N_f}{N} \tag{9.10}$$

where N_f is the number of samples that have the performance function less than or equal to zero, i.e. $g \le 0$.

The reliability is then estimated by

$$R = \mathrm{P}\{g > 0\} = 1 - p_f = \frac{N - N_f}{N} \tag{9.11}$$

Similar to the calculation of the probability of failure, the *cdf* is given by

$$F_Y(y) = \mathrm{P}(g \le y) = \frac{1}{N}\sum_{i=1}^{N} I'(y_i) \tag{9.12}$$

where the indicator function is defined by

$$I'(\mathbf{x}) = \begin{cases} 1 & \text{if } g(\mathbf{x}) \le y \\ 0 & \text{otherwise} \end{cases} \tag{9.13}$$

The *pdf* $f_Y(y)$ can be obtained by the numerical differentiation of *cdf* $F_Y(y)$.

8.5 Examples

Two examples are used to demonstrate Monte Carlo simulation.

Example 9.1

The allowable stress X_1 of a mechanical component is normally distributed, $X_1 \sim N(120,20)$ MPa , and the maximum stress X_2 is normally distributed, $X_2 \sim N(100,10)$ MPa . What is the probability of failure p_f ?

Analytical solution

For this simple problem, an analytical solution exists. Since both X_1 and X_2 are normally distributed, $Y = g(\mathbf{X}) = X_1 - X_2$ is also normally distributed. Then the probability of failure is given by

$$p_f = P(Y < 0) = F_Y(0) = \Phi\left(\frac{0 - m_Y}{s_Y}\right) = \Phi\left[\frac{-(120-100)}{\sqrt{20^2+10^2}}\right] = 0.1855$$

Monte Carlo simulation solution

Table 9.2 shows the results from Monte Carlo simulation. 20 samples for X_1 and X_2 are drawn from their respective distributions. Then the output Y is calculated with the function $Y = g(\mathbf{X}) = X_1 - X_2$, and the indication function I is also calculated.

Table 9.2 Samples from Monte Carlo Simulation

Simulation	X_1	X_2	Y	I
1	158.9081	96.3591	62.549	0
2	127.9944	97.7657	30.2287	0
3	89.3021	89.0936	0.2085	0
4	114.9522	94.0466	20.9056	0
5	129.7092	74.1456	55.5636	0
6	122.9478	112.6098	10.338	0
7	163.0985	95.8219	67.2765	0
8	115.815	109.4279	6.3871	0
9	121.8649	102.7287	19.1362	0
10	103.0178	109.9262	-6.9085	1
11	126.8695	86.0036	40.8658	0
12	138.3068	114.8298	23.4771	0
13	118.5968	86.3359	32.2609	0
14	149.3739	94.2811	55.0927	0
15	145.7642	95.8336	49.9307	0

Simulation	X_1	X_2	Y	I
16	87.46	105.8226	-18.3626	1
17	89.7571	85.6344	4.1227	0
18	111.4169	92.4829	18.934	0
19	120.5707	98.9355	21.6353	0
20	119.3122	113.1373	6.1749	0

From Eq. 9.10, the probability of failure is computed by

$$p_f = \bar{I}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} I(y_i) = \frac{2}{20} = 0.10$$

Other characteristics of Y can also be calculated as follows.

The mean

$$\bar{Y} = \frac{1}{N}\sum_{i=1}^{N} y_i = \frac{1}{20}\sum_{i=1}^{20} y_i = 20.9823 \text{ MPa}$$

The standard deviation

$$S = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(y_i - \bar{Y})^2} = \sqrt{\frac{1}{20-1}\sum_{i=1}^{20}(y_i - 20.9823)^2} = 19.1673 \text{ MPa}$$

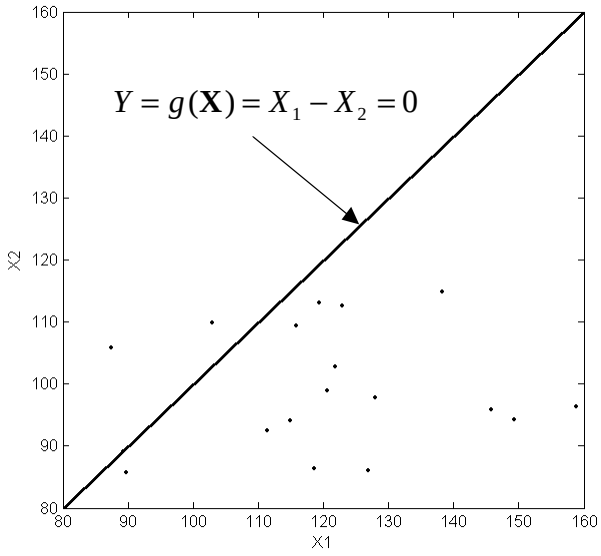


Figure 9.3 Samples of X_1 and X_2

The samples of X_1 and X_2 are plotted in Fig. 9.3. As shown in the figure and Table 9.2, two samples are in the failure region; namely, the number of failures is two.

The result of the probability of failure p_f has a large error compared to the analytical solution because the number of simulations is small. If the number of simulations *N* increases, the solution of p_f will be more accurate. The solutions from different number of simulations are displayed in Table 9.3. The numbers of failures and the errors of the solutions of p_f are also shown in the table. As the number of simulations increases, the error decreases. Theoretically, if the number of simulations approaches infinity, the solution of p_f converges to the true solution. The relationship between the error and the number of simulations will be discussed in next section.

Table 9.3 Results from Different Numbers of Simulations

N	20	10^3	10^4	10^5	10^6
N_f	2	156	1.763	18606	185446
p_f	0.10	0.1560	0.1763	0.1861	0.1854
Error	-46.09%	-15.90%	-4.96%	0.32%	-0.05%

Example 9.2

A cantilever beam is illustrated in Fig. 9.4. This is the same problem we have presented in Example 7.2.

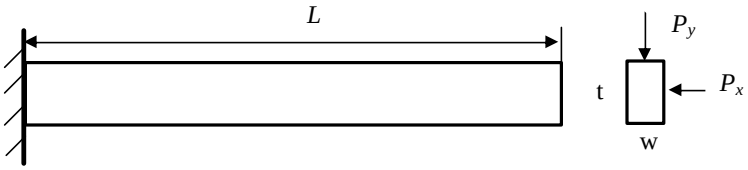


Figure 9.4 A Cantilever Beam

One of the failure modes is that the tip displacement exceeds the allowable value, D_0. The performance function is the difference between D_0 and the tip displacement and is given by

$$g = D_0 - \frac{4L^3}{Ewt}\sqrt{\left(\frac{P_y}{t^2}\right)^2 + \left(\frac{P_x}{w^2}\right)^2}$$

where $D_0 = 3^{"}$, $E = 30\times10^6$psi is the modulus of elasticity, $L = 100^{"}$ is the length, *w* and *t* are width and height of the cross section, respectively, and $w = 2^{"}$ and $t = 4^{"}$. P_x and P_y are external forces which follow normal distributions, and $P_x \sim N(500,100) lb$ and $P_y \sim N(1000,100) lb$.

The probability of failure is defined as the probability of the allowable displacement less than the tip displacement, i.e.

$$p_f = P\left\{ g = D_0 - \frac{4L^3}{Ewt}\sqrt{\left(\frac{P_y}{t^2}\right)^2 + \left(\frac{P_x}{w^2}\right)^2} \le 0 \right\}$$

This problem involves only two random variables and can be easily visualized within a 2-D random space as shown in Fig. 9.5. It is seen that the space is divided by the curve of limit state $g = 0$ into the safe region and failure region. At first, 10 samples of P_x and P_y are drawn and are shown in Table 9.4. The samples are also depicted in the Fig. 9.5 (a).

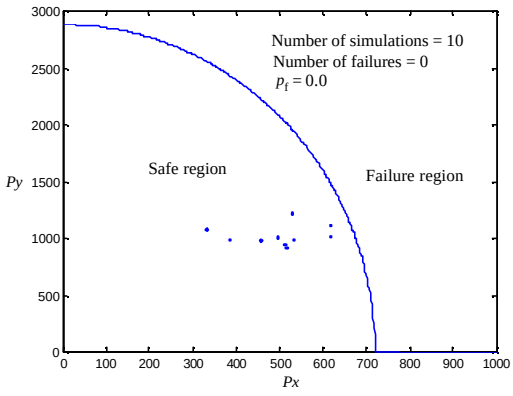
Table 9.4 Random Samples for the Example

Simulations	P_x	P_y	Y
1	456.7435	981.3291	0.8397
2	333.4416	1072.5791	1.2171
3	512.5332	941.1683	0.6502
4	528.7676	1218.3186	0.4574
5	385.3529	986.3604	1.0938
6	619.0915	1011.3931	0.2136
7	618.9164	1106.6768	0.1752
8	496.2367	1005.9281	0.6820
9	532.7292	990.4352	0.5522
10	517.4639	916.7651	0.6419

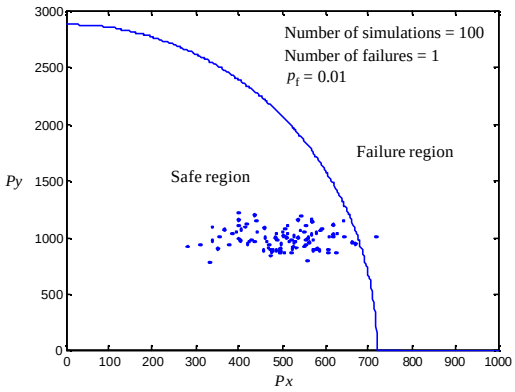
At all the sample points, the performance function *g*>0 (see Table 9.4 and Fig. 9.5(a)). In other words, all the samples fall into the safe region. Therefore the probability of failure is equal to zero. This means that the sample size is not large enough. If 100 samples are used, there is one sample falling into the failure region (see Fig. 9.5 (b)). Therefore the probability is 1/100=0.01. If we increase the sample size to 1,000, there are 27 samples where the beam fails. This gives an estimate of the probability of failure of 27/1000 = 0.027 (see Fig. 9.5 (c)). As we have seen in the last example, the larger the sample size is, the more accurate the estimate of the probability of failure will be. If 100,000 simulations are performed, the estimate of the probability of failure is 0.04143 as shown in Fig. 9.5 (d). Recall that the probability of failure computed by FORM in Example 7.2 is 0.04054 and by SORM in Example 7.5 is 0.04098. If the probability of failure obtained from

Monte Carlo simulation with 100,000 simulations is considered as an accurate solution, and the example confirms that SORM is more accurate than FORM for this problem.

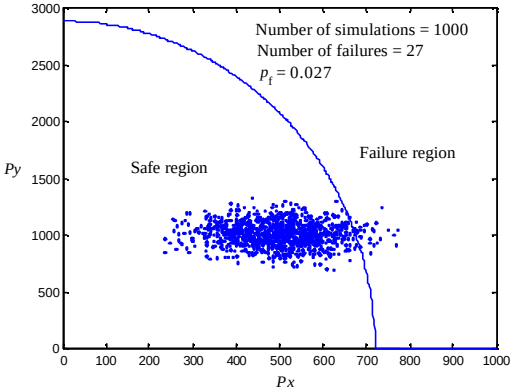
Figs. 9.6 and 9.7 show the *cdf* and *pdf* of the performance function *g* with 100,000 simulations.



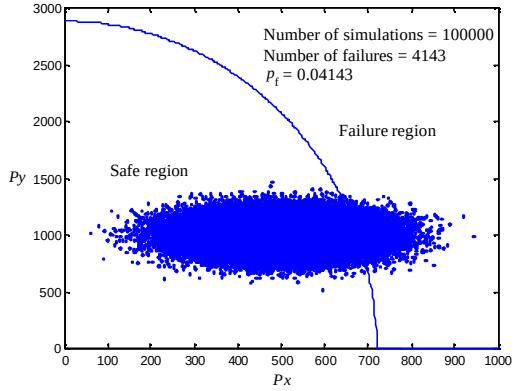
(a) N=10



(b) N=100



(c) N=1000



(d) N=100000

Figure 9.5 Monte Carlo Simulation of the Cantilever Beam

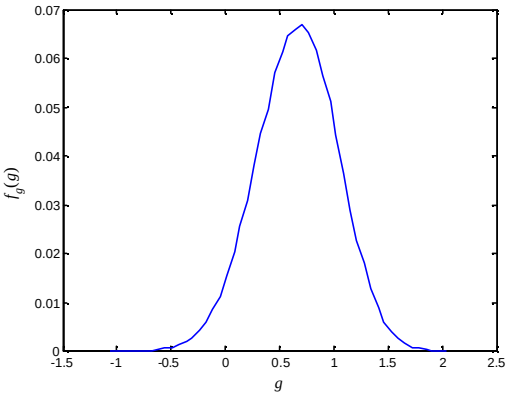


Figure 9.6 cdf

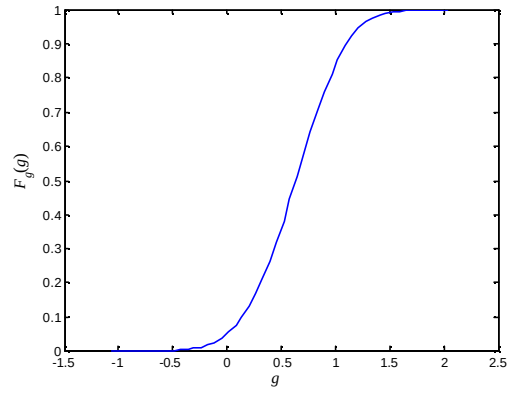


Figure 9.7 pdf

8.6 Error Analysis

As demonstrated in preceding examples, the accuracy of Monte Carlo Simulation depends on the number of simulations *N*. The higher the number of simulations is, the more accurate the estimate will be. As the number of simulations *N* approaches infinity, the solution of Monte Carlo simulation will converge to the true probability that is under estimation. Since reliability assessment normally needs high accuracy, it is important to know the error involved in the estimated probability of failure. On the other hand, it is also important to know how many simulations are required to achieve the desired accuracy. The percentage error of the estimate of the probability failure is found to be

$$e\,\% = 100 u_{1-a/2}\sqrt{\frac{(1-p_f^T)}{Np_f^T}} \tag{9.14}$$

where $u_{1-\alpha/2}$ is the (1-*a*/2) quantile (percentile value) of the standard normal distribution, and p_f^T is the true value of the probability of failure. The above equation gives the percentage error under the 100(1-*a*)% confidence. The error is not in an absolute (deterministic) sense. On the contrary, it indicates that there is a 100(1-*a*)% chance that the probability of failure will be in the range of $p_f \pm p_f \frac{\varepsilon}{100}$ with *N* simulations. The commonly used confidence level is 95% under which the error is approximately given by

$$e\,\% \approx 200\sqrt{\frac{(1-p_f^T)}{Np_f^T}} \tag{9.15}$$

Since the theoretic value of the probability of failure p_f^T is unknown, the estimated probability of failure p_f is used in Eqs. 9.14 and 9.15 to replace p_f^T.

For example, if $p_f = 0.001$ and the number of simulations $N = 10^5$, with 95% confidence, from Eq. 9.15 the error is calculated to be $\varepsilon = 20\%$. With 95% likelihood, therefore, the probability of failure will be $p_f = 0.001 \pm 0.0002$. If we desire the error to be within 10%, from the same equation, the number of simulation calculated should be 399,600.

For the beam problem in Example 9.2, the error of the probability of failure with 100,000 samples is

$$e\,\% \approx 200\sqrt{\frac{(1-0.04143)}{100000\times 0.04143}} = 3.0422\%$$

Since the point estimate of the probability of failure is 0.04143, the 95% confident interval is $0.04143 \pm 0.04143 \times 3.0422\%$, i.e. [0.0401, 0.0427].

According to Eq. 9.14, higher reliability (or lower probability of failure) requires a higher number of simulations. For example, a probability of failure of $p_f = 10^{-6}$ indicates that only one item will fail out of 1 million items. The required number of simulations is about $N = 4 \times 10^8$ with 10% error under 95% confidence. Therefore higher reliability requires a higher computational effort. For the same reason, if higher accuracy for estimating *cdf* in distribution tail areas, a large number of simulations must be used.

From the above discussion, the features of Monte Carlo simulation are summarized as follows.

1) Monte Carlo simulation is easy to use for engineers who have only limited working knowledge of probability and statistics.

2) Monte Carlo simulation is feasible to use for virtually any performance functions and distributions.

3) Monte Carlo simulation is computationally robust; with sufficient number of simulations, it can always converge.

4) The problem dimension (the number of random variables) does not affect the accuracy of Monte Carlo simulation as indicated in Eq. 9.14. This feature is beneficial to large scale engineering problems.

5) For reliability analysis, Monte Carlo simulation is generally computationally expensive. The higher the reliability is, the larger the simulation size is needed.

Because of the accuracy, Monte Carlo simulation is widely used in 1) engineering applications where the model evaluations (deterministic analyses) are not computationally expensive and 2) validating other methods. However, due to its computational inefficiency, Monte Carlo simulation is not commonly used for problems where deterministic analyses are expensive.

8.7 MPP Based Importance Sampling

From the preceding discussions, it is noted that the computational cost is very high when the probability of failure is small. The reason is that only the samples that fall into the failure region can contribute to the probability estimation. Importance sampling technique was developed with the motivation to improve the computational efficiency. The central idea of importance sampling is to sample the random variables according to an alternative set of distributions such that more samples will be in the failure region.

More samples will therefore contribute to the probability estimation. The idea is illustrated in Fig. 9.8.

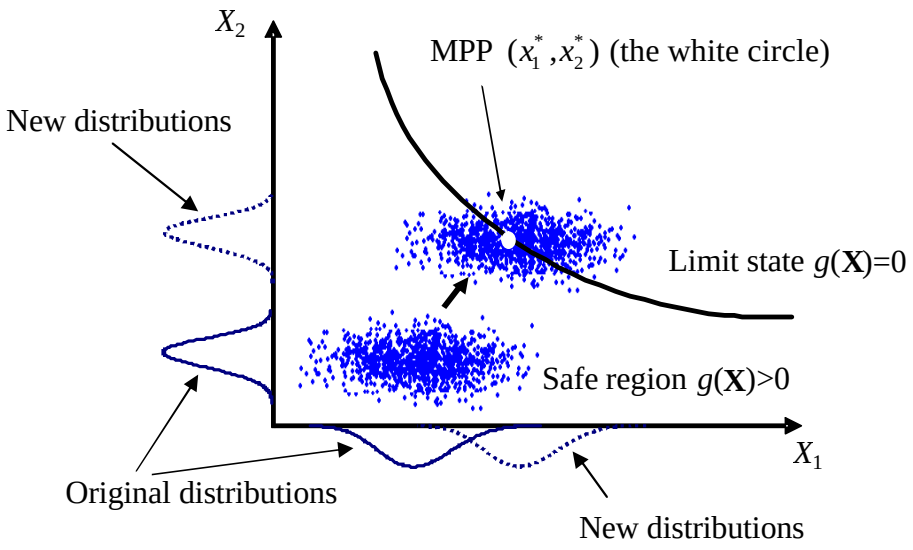


Figure 9.8 Importance Sampling

As shown in Fig. 9.8, all the samples (the lower cloud) generated from the original distributions of X_1 and X_2 are in the safe region. They do not have any contribution to the probability estimation. If a new set of distributions of X_1 and X_2 is selected such that many samples will fall into the failure region, then the samples (the upper cloud in the figure) will contribute to the probability estimation significantly. There are several importance sampling schemes, and we will discuss the MPP based importance sampling herein.

After the MPP is obtained, samples are picked around the MPP to evaluate the probability of failure through importance sampling (see Fig. 9.8). To do so, an importance-sampling density, $h_{\mathbf{X}}(\mathbf{x})$, is introduced into the Monte Carlo estimation, Eq. 9.7, to obtain the probability of failure

$$p_f = \int I[g(\mathbf{X})] f_{\mathbf{X}}(\mathbf{x}) d\mathbf{x} = \int \left\{ I[g(\mathbf{X})] \frac{f_{\mathbf{X}}(\mathbf{x})}{h_{\mathbf{X}}(\mathbf{x})} \right\} h_{\mathbf{X}}(\mathbf{x}) d\mathbf{x} \tag{9.16}$$

where the importance-sampling density $h_{\mathbf{X}}(\mathbf{x})$ is the same as $f_{\mathbf{X}}(\mathbf{x})$ except that the means values of $\mathbf{X}$ are replace by the MPP $\mathbf{x}^* = (x_1^*, x_2^*, \cdots, x_n^*)$. For example, if the distribution of X_1 is normally distributed with $N(m_1, s_1)$ and $f_{X_1}(x) = \frac{1}{\sqrt{2p}\, s_1} \exp\left[-\frac{1}{2} \left(\frac{x - m_1}{s_1} \right)^2 \right]$, the corresponding importance sampling distribution

will be $N(x_1^*, s_1)$ and $h_{X_1}(x) = \frac{1}{\sqrt{2p}\, s_1} \exp\left[-\frac{1}{2}\left(\frac{x - x_1^*}{s_1}\right)^2\right]$. It is noted that the importance sampling density $h_{X_1}(x)$ is centered at x_1^*.

Eq. 9.16 indicates that the probability of failure is the mean of the integrand $\left\{I[g(\mathbf{X})]\frac{f_{\mathbf{X}}(\mathbf{x})}{h_{\mathbf{X}}(\mathbf{x})}\right\}$ that is evaluated at the samples of $\mathbf{X}$ drawn from the importance sampling density $h_{\mathbf{X}}(\mathbf{x})$. Therefore,

$$p_f = \frac{1}{N}\sum_{i=1}^{N} I[g(\mathbf{x}_i)]\frac{f_{\mathbf{X}}(\mathbf{x}_i)}{h_{\mathbf{X}}(\mathbf{x}_i)} \tag{9.17}$$

As shown in Fig. 9.8, with the sample size, a significant number of failures occur when the samples are drawn from the importance sampling density $h_{\mathbf{X}}(\mathbf{x})$. It can be approved that with the same sample size the error bound of importance sampling is much smaller than the general Monte Carlo simulation.

Example 9.3

Use MPP based importance sampling to solve Example 9.2.

From Example 7.2, the MPP in U-space from FORM is $\mathbf{u}^* = (1.7367, 0.16376)$. Their transformation to X-space is $\mathbf{x}^* = (673.67, 1016.38)$ The original distributions of two random variables are $P_x \sim N(500,100)$ and $P_y \sim N(1000,100)$, and the distributions for importance sampling are then given by $N(673.67,100)$ for P_x and $N(1016.38,100)$ for P_y.

Fig. 9.9 shows the results of the general Monte Carlo simulation where there are only 27 failures out of 1000 samples. The probability of failure is 0.027 which has a large error as compared to the accurate solution $p_f = 0.04143$ with 100,000 simulations. Fig. 10 shows the results from importance sample with only 200 samples, out of which 96 samples are in the failure region. The calculated probability of failure is 0.040976 and is very close to the accurate solution from 100,000 general Monte Carlo samples.

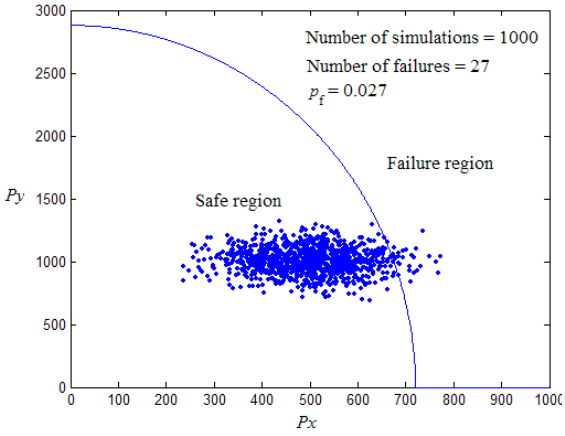


Figure 9.9 General Monte Carlo simulation

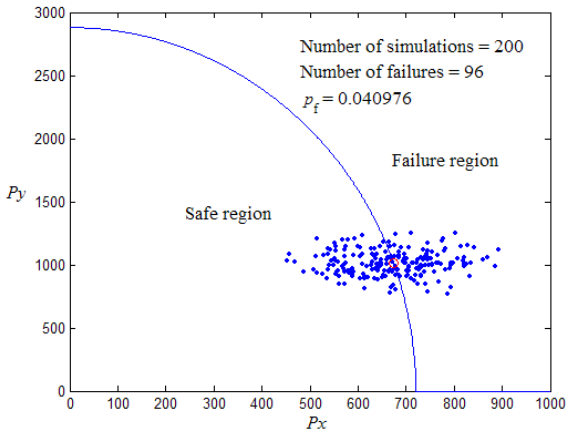


Figure 9.10 Importance Sampling

8.8 MATLAB Random Number Generators

MATLAB provides random number generators for commonly used distributions.

1) normrnd – random matrices from normal distribution

R = normrnd(MU,SIGMA) returns a random number chosen from the normal distribution with parameters MU (mean) and SIGMA (standard deviation).

R = normrnd (MU,SIGMA,M,N) returns an M by N matrix.

2) lognrnd – random matrices from the lognormal distribution

R = lognrnd (MU,SIGMA) returns a random number chosen from the lognormal distribution with parameters MU (mean) and SIGMA (standard deviation).

R = LOGNRND(MU,SIGMA,M,N) returns an M by N matrix.

3) exprnd – random matrices from exponential distribution

R = exprnd (MU) returns a random number chosen from the exponential distribution with parameter MU (mean).

R = exprnd (MU,M,N) returns an M by N matrix.

4) unifrnd – random matrices from continuous uniform distribution

R = unifrnd (A,B) returns a random number chosen from the continuous uniform distribution on the interval from A to B.

R = UNIFRND(A,B,M,N) returns an M by N matrix.

In addition to the above random variable generators, MATLAB can also generate random variables for more distributions.