

Efficient Random Vibration Analysis Using Markov Chain Monte Carlo Simulation

Mahdi Norouzi and Efstratios Nikolaidis
 Univ. of Toledo

ABSTRACT

Reliability assessment of dynamic systems with low failure probability can be very expensive. This paper presents and demonstrates a method that uses the Metropolis-Hastings algorithm to sample from an optimal probability density function (PDF) of the random variables. This function is the true PDF truncated over the failure region. For a system subjected to time varying excitation, Shinozuka's method is employed to generate time histories of the excitation. Random values of the frequencies and the phase angles of the excitation are drawn from the optimal PDF. It is shown that running the subset simulation by the proposed approach, which uses Shinozuka's method, is more efficient than the original subset simulation. The main reasons are that the approach involves only 10 to 20 random variables, and it takes advantage of the symmetry of the expression of the displacement as a function of the inputs. The paper demonstrates the method on two examples.

CITATION: Norouzi, M. and Nikolaidis, E., "Efficient Random Vibration Analysis Using Markov Chain Monte Carlo Simulation," *SAE Int. J. Mater. Manf.* 5(1):2012, doi:10.4271/2012-01-0067.

1. INTRODUCTION

Standard Monte Carlo Simulation (MCS) is a robust tool for reliability assessment. However, it can be impractical for most real-life problems because of its high computational cost. This is particularly true for dynamic systems in which response is driven by a time-varying excitation with low probability of failure (e.g. less than 10^{-4}). Reliability can be measured by the probability of the response crossing a given threshold during a time period. For real life complex systems, the failure region is small and is hard to locate. Markov Chain Monte Carlo (MCMC) is an efficient method for generating samples from a probability distribution for which sampling is difficult. MCMC was first presented by Metropolis et al. [1], and was later generalized by Hastings [2].

Au and Beck [3, 4] introduced *subset* simulation, which is based on the idea that a small failure probability can be calculated as a product of larger conditional probabilities of intermediate events. This method is more efficient than standard Monte Carlo simulation because it is much faster to calculate several large probabilities than a single low probability. However, the method is often impractical for random vibration problems because it requires generating the sample values of the excitation at hundreds or thousands points in time. Au and Beck [4] considered 301 random variables when they demonstrated their method on a single degree of freedom system. It is difficult to explore the space of the random variables because of its large dimension (301).

This paper proposes a method that combines subset simulation with Shinozuka's method to circumvent this difficulty. This method is practical because it draws samples of only about 10 random variables, which are the frequencies of the harmonic of the random excitation.

The next section explains Markov Chain Monte Carlo simulation and the Metropolis-Hastings algorithm [1, 2], which is used in subset simulation. Then section 3 explains the subset simulation method. In section 4, we present Shinozuka's method [5] to generate time histories of a random process, and in section 5, we explain our proposed approach to improve the efficiency of subset simulation. Section 6 demonstrates method on two examples.

2. METROPOLIS - HASTINGS ALGORITHM

Metropolis-Hastings [1, 2] algorithm is a sequential approach that generates a Markov Chain whose states follow a target PDF, $f_{\mathbf{X}}^S(\mathbf{x})$. A state of the chain is a vector containing the values of the random variables. The algorithm starts from an initial state $\mathbf{X}_0$, and generates a sequence of states $\mathbf{X}_1, \mathbf{X}_2, \dots$ as follows.

Repeat k times:

1. Draw candidate state $\mathbf{X}_i$ from transition PDF $h_{\mathbf{X}_i/\mathbf{X}_{i-1}}(\mathbf{x}_i/\mathbf{x}_{i-1})$. This is the PDF of state $\mathbf{X}_i$ conditioned upon the previous state $\mathbf{X}_{i-1}$.

2. Make $\mathbf{X}_i$ the next state of the chain with probability of $\min\{f_{\mathbf{X}}^S(\mathbf{x}_i)/f_{\mathbf{X}}^S(\mathbf{x}_{i-1}), 1\}$. Set $\mathbf{X}_i = \mathbf{X}_{i-1}$ with the remaining probability, $\max\{1 - f_{\mathbf{X}}^S(\mathbf{x}_i)/f_{\mathbf{X}}^S(\mathbf{x}_{i-1}), 0\}$. The algorithm returns to the previous state in this case.

The transition PDF is symmetric, $h_{\mathbf{X}_i/\mathbf{X}_{i-1}}(\mathbf{x}_i/\mathbf{x}_{i-1}) = h_{\mathbf{X}_{i-1}/\mathbf{X}_i}(\mathbf{x}_{i-1}/\mathbf{x}_i)$. This algorithm produces a sequence of states with stationary PDF equal to the target one. This means that if a particular state follows the target PDF, so does the next state.

MCMC enables the user to explore the failure region more efficiently than standard MCS because it uses information from the current state to move to the next state. This is especially important when the failure region is small. On the other hand, it generates dependent states, which increases the variability of the estimates of the probability of failure. In addition, the analyst must understand well the problem and the shape of the failure region. If the failure region consists of disjoint or loosely connected regions, the user must generate multiple sequences of points to cover each region with enough points.

The user has to select the transition PDF. In this study, the transition PDF is the probability density of random vector $\mathbf{X}_i - \mathbf{X}_{i-1} \cdot (1 + \Delta)$, where Δ is a zero-mean Gaussian variable. The components of the vector are statistically independent. The choice of the standard deviation of Δ is challenging. A small value produces a sequence of highly depended states, with the attendant reduction in the accuracy of the estimates of the probability of failure. On the other hand, a large value of the standard deviation of Δ increases the frequency that the algorithm returns to the previous state. A standard deviation in the range from 0.05 to 0.3 produced reasonable results in this study.

It is challenging to apply MCMC to a problem with more than 10 random variables. In this case, the Metropolis algorithm tends to return to same state very frequently, which increases the dependence of the states with the attendant reduction in the accuracy of the estimate of the probability of failure. Au and Beck [4] suggested a modified algorithm that changes one random variable at a time in step 1 of MCMC. We adopted this algorithm in this study because it needed fewer steps than the original to estimate the probability of failure.

3. SUBSET SIMULATION TO ESTIMATE LOW FAILURE PROBABILITIES

Reliability of a system is often assessed by MCS. When the probability of failure is low, performing MCS becomes

impractical. Au and Beck [4] introduced subset simulation to address this drawback. In the subset simulation, the problem of estimating a low probability is broken into several sub-problems with higher probabilities of failure, which are much less expensive to solve. For example, the estimation of a probability of 0.1 requires approximately 1,000 replications, whereas estimation of a probability of 10^{-4} requires 10^6 replications. In both cases, the maximum acceptable value of the standard deviation is equal to 10% of that probability. Determination of these sub-problems is important in subset simulation, and requires setting the intermediate probabilities to calculate the probability of failure of the system.

Consider a series of thresholds as g_1, g_2 to g_n , where n is an unknown number of subset subdivisions. Then failure regions F_1 to F_n are defined as follows:

$$F_i = \{x: G(x) \leq g_i\} \quad i = 1, 2 \dots n. \quad (1)$$

In the above equation x is a vector of random variables and $G(x)$ is a function to evaluate the response. The total probability of failure becomes the product of intermediate conditional probabilities, $P(F_{i+1}|F_i)$, and PF_1 as follows:

$$PF = P(F_1) \prod_{i=1}^{n-1} P(F_{i+1}|F_i). \quad (2)$$

The accuracy of subset simulation depends on the choice of the intermediate failure regions. The analyst needs to perform standard MCS to initiate the subset simulation and to calculate PF_1 . After getting the result of the initial MCS, it becomes more convenient to set the subsets F_2 to F_n . One approach is to consider subsets with almost equal failure probabilities. Selecting failure regions with conditional probabilities approximately greater or equal to 0.1 often yields good results. Conditional probabilities that are smaller than 0.1 can result in excessive coefficient of variation (COV) thereby losing accuracy.

Subset simulation consists of the following steps:

a). Perform standard MCS to determine PF_1 and failure region F_1 to initiate subset simulation. This probability is calculated as follows:

$$PF_1 = \frac{1}{K_1} \sum_{k=1}^{K_1} I_{F_1}(x_k^{(1)}) \quad (3)$$

where in the above equation $I_{F_1}(\cdot)$ is the indicator function with first level threshold and $x_k^{(1)}$ is a sample drawn from the input PDF using regular sampling methods like Acceptance-Rejection or Inverse-Transform methods [6].

b). Obtain the probability functions for each subset:

$$f(x|F_i) = \frac{I_{F_i}(x) f(x)}{P(F_i)} \quad (4)$$

c). Generate samples using the Metropolis-Hastings [1,2] algorithm (see Sec. 2) to estimate conditional $P(F_{i+1}|F_i)$ as follows:

$$P(F_{i+1}|F_i) = \frac{1}{K_{i+1}} \sum_{k=1}^{K_{i+1}} I_{F_{i+1}}(x_k^{(i+1)}). \quad (5)$$

d). Calculate the total probability of failure by Eq. (2)

Au and Beck [4] modified the original Metropolis-Hastings algorithm [1, 2] so that only one random variable changes in each step of the algorithm. They did so in order to increase the frequency that the MCMC process moves to a new state. This is important when there are many random variables (for example, more than 10).

The accuracy of the results is usually measured by their COV,

$$\delta_1 = \sqrt{\frac{1 - PF_1}{PF_1 K_1}}. \quad (6)$$

The COV for the intermediate conditional probabilities can be estimated as follows [4]:

$$\delta_{i+1} = \sqrt{\frac{1 - P(F_{i+1}|F_i)}{P(F_{i+1}|F_i) K_{i+1}}} (1 + \gamma_{i+1}) \quad i = 1, 2, \dots, n-1. \quad (7)$$

For a single Markov chain, the correlation factor γ_{i+1} is calculated as follows,

$$\gamma_{i+1} = 2 \sum_{j=1}^N \left(1 - \frac{j}{N}\right) \rho_{i+1}(j) \quad (8)$$

where $\rho_{i+1}(j)$ is the correlation coefficient of the Markov chain samples, and j is the separation between two values of the displacement in the chain. In practice, the correlation coefficient between two points in the chain vanishes beyond an upper limit that is less than N . This limit is used in Eq. (8) to reduce the computational effort of the method.

Assuming independence of the intermediate conditional probabilities in Eq. (2), the COV of the total probability of failure from subset simulation is approximated by the following equation, as shown in the appendix:

$$\delta = \sqrt{\sum_{i=1}^n \delta_i^2} \quad i = 1, 2, \dots, n. \quad (9)$$

4. SHINOZUKA'S METHOD TO SIMULATE TIME HISTORIES OF A RANDOM PROCESS

Several methods exist to generate input time series. These are classified in three groups:

I. White noise filtration method: In this method, Gaussian white noise passes through a filter representing the input spectrum.

II. The random phase angle method: In this method, the time series are produced by the sum of cosine series, where each cosine function has a random frequency and a fixed amplitude dependent on the input spectrum and a random phase angle

III. The random coefficient method: This approach generates realizations by adding sine and cosine functions, where each sine and each cosine function has random, normally distributed amplitude dependent on the input spectrum.

The first method is widely used by researchers. First a realization of a white noise process $W(\omega)$ is generated. Then the time series are calculated by the following equation:

$$X(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} W(\omega) \sqrt{2\pi S(\omega)} e^{-j\omega t} d\omega. \quad (10)$$

The realizations generated by the above method are Gaussian.

The 2nd and 3rd methods resemble Fourier series and are also valid for Gaussian processes.

Shinozuka [5] proposed a method that generates a time history by superimposing a number of cosines. The frequencies of these cosines are drawn from a PDF function that is equal to the PSD of the random process normalized by the area under it,

$$f_{\Omega}(\omega) = \frac{S(\omega)}{\sigma^2} \quad (11)$$

where $S(\omega)$ is the PSD of the random process, and σ is the square root of the area under one-sided PSD function, $S(\omega)$,

$$\sigma = \sqrt{\int_0^{\infty} S(\omega) d\omega}. \quad (12)$$

If a process $X(t)$ is stationary and Gaussian with zero mean, then its sample time history can be approximated as follows,