UNIVERSIDAD TÉCNICA FEDERICO SANTA MARIA

DEPARTAMENTO DE OBRAS CIVILES

Reliability Estimation Methods and their Efficient Implementation

Tesis de Grado presentada por

Martin Ralf Oswald

como requisito parcial para optar al grado de

Magíster en Ingeniería Civil

Profesor Guía
Dr. Héctor Jensen V.

September, 2008

TITULO DE LA TESIS:

Reliability Estimation Methods and their Efficient Implementation

AUTOR:

Martin Ralf Oswald

TRABAJO DE TESIS, presentado en cumplimiento parcial de los requisitos para el Grado de Magíster en Ingeniería Civil de la Universidad Técnica Federico Santa María.

Dr. Héctor Jensen V. ______________________

Dr. Rodrigo Delgadillo S. ______________________

Dr. Fernando Labbe Z. ______________________

Valparaíso, Chile. September, 2008

Abstract

This thesis discusses the methods importance sampling and subset simulation, which are used to efficiently estimate small failure probabilities in order to investigate the reliability of structures under uncertain input. The work focuses on the first excursion problem for linear and non-linear systems with a high dimensional uncertain input. The methods and their variants are reviewed and studied in order to implement them efficiently and in a parallel manner. The different generality of the sampling methods is discussed and incorporated in the proposed simulation framework. This provides high source code re-usability due to the problem independent encapsulation of each method. The sampling methods are compared in several experiments, which show their advantages, limits and their applicability to parallel computation.

keywords: Monte Carlo simulation, importance sampling, subset simulation, structural analysis, reliability, parallel computing

Resumen

Esta tesis presenta los métodos “importance sampling” y “subset simulation”. Estos métodos se usan para estimar probabilidades de falla pequeñas de modo eficiente, con el objetivo de investigar la confiabilidad de estructuras bajo la influencia de parámetros inciertos. Este trabajo se focaliza en el problema de la primera incursión para sistemas lineales y no-lineales de gran dimensión probabilística. Los métodos y sus variantes son analizados y estudiados para ser implementados de una manera paralela y eficiente. Los diferentes grados de generalidad de los métodos considerados son discutidos e incluidos en la estructura del software de simulación propuesta, que ofrece un alto grado de reutilización del código de programación mediante el blindaje de cada método independientemente del problema tratado. Los métodos de sampling se comparan en varios experimentos numéricos mostrando así sus ventajas, sus límites y sus aplicaciones en la computación paralela.

Palabras claves: simulación de Monte Carlo, importance sampling, susbet simulation, problema de la primera incursión, confiabilidad, computación paralela

Acknowlegdements

I would like to thank Prof. Héctor Jensen who made my stay at the Universidad Técnica Federico Santa Maria eventually possible. I am grateful for the constructive discussions and his guidance throughout this thesis. For his help and many clarifying discussions I would like to thank Danilo Kusanovic.

The financial support of the German Academic Exchange Service is gratefully acknowledged.

Many thanks also go to my flatmates Ignacio Salazar, Dan Soto and Camila Montero for their enthusiasm and their patience to teach me 'chilean' spanish. They really made my stay in Chile to a great experience in my life.

Also, I would like to thank Steffen Jaensch, not only for proof reading, but also for his support and friendship. My parents and grandparents deserve my deepest appreciation for their constant love and support. They always helped me in any possible way and encouraged me to pursue my goals and dreams.

Lastly, I thank my girlfriend Georgiana for her accompany, patience and motivation during my work on this thesis.

Contents

List of Figures

1. Introduction

1.1. Motivation

In order to analyze real world structures adequately it is necessary to take all parameters into account. Mostly, structural parameters are subject to uncertainties, which may occur in loading conditions or in structural behavior. Of special interest is the behavior of structures subjected to dynamic loads, which may, for example, be caused by earthquakes, wind, ocean waves or traffic. In practice, it is important that the response of the structure to uncertain loadings does not exceed given tolerance thresholds in order to prevent lasting material changes or even a collapse of the structure. This problem is generally known as the first excursion problem and is of particular interest in structural safety and reliability based structural engineering [2, 3].

The idea to use uncertainty models in engineering not only stems from the pure ambition to investigate the reliability of structures. With knowledge about the structure reliability for uncertain short-term dynamic loadings one can also draw conclusion about the lifetime reliability of a structure [4]. As another example, the reliability of a structure is also a desirable property in structural optimization. In economical practice, cost minimization plays an important role. Therefore the minimization of structural costs under safety restrictions is of common interest in current research, e.g. [5, 6].

To perform structural analysis close to reality, it is natural to describe incomplete information in a stochastic way since uncertainties can usually be well described by means of a probability distribution which may be obtained from analytical or empirical studies. The mathematical model of the structure is thus extended by the probabilistic model for the uncertain parameters. Due to the higher complexity, especially for industry size structures analytical methods for probabilistic analysis often become infeasible and numerical methods need to be applied. As a very general approach, Monte Carlo-based methods have proved to be a robust approach for any level of complexity of the problem [7, 8].

In order to examine the reliability of a structure the probability for a system failure is investigated. Since real world structures are designed to withstand common types of loadings, the probability of failure is generally small. This in turn makes the pure Monte Carlo method inefficient, because rare occurrences of failure events need a large amount of samples in order to obtain a good estimation. Since the generation of each sample is coupled with a system analysis, this property is a severe drawback in view of computational efficiency. Therefore, several approaches have been developed to improve the efficiency of the estimation of small failure probabilities, e.g. importance sampling

[9, 10, 11], line sampling [12] or subset simulation [13, 1, 14]. These methods are the first steps to make reliability analysis of industry size structures practically feasible. Consequently, the main goal of research in this area is to further improve the computational efficiency of the failure probability estimation. This can be achieved not only by the development of better estimators, but also by means of approximation methods [5, 15] or by run time improvements due to an efficient implementation of the methods.

The scope of this work is to study recent failure probability methods for linear and non-linear systems with respect to their efficient implementation and in a parallel manner. Therefore, the sampling methods importance sampling and subset simulation are reviewed and studied to find out which influences the efficiency of each method and investigate possible improvements. Furthermore, differences and similarities of the methods are investigated and incorporated in order to develop a flexible and efficient software package providing these methods.

1.2. Thesis Outline

The thesis starts with the mathematical formulation of the reliability problem at the beginning of chapter 2. In the following, five methods for numerically calculating the probability of failure - each with different generality and efficiency - are described.

As the most reliable and robust basis, Monte Carlo simulation used for failure probability estimation is reviewed briefly. Then the principle of importance sampling is applied to the special case of structures with linear behavior. Then, the very general method subset simulation is explained in detail on the example of three variants of this method. Chapter 3 focuses on the efficient implementation and parallelization of the methods described in chapter 2 and explains the structure of the software package which has been developed in this work.

In the following, chapter 4 illustrates the applicability and properties of the methods by several examples. Finally, the thesis concludes with a summary in chapter 5.

2. Failure Probability Sampling

2.1. Problem Definition

2.1.1. General Definition and Notation

To examine the reliability of a system being described by its system state $x \in \mathcal{X}$ and influenced by the system input $u \in \mathcal{U}$, one first needs to define the set $F \subseteq \mathcal{X}$ of unwanted systems states (failures) and secondly calculate the probability that any system failure event occurs. In physics and engineering tasks, system input and state are generally continuous and described by vectors and the sets of states $\mathcal{X}$ and the set of inputs $\mathcal{U}$ are described by $\mathcal{X} = \mathbb{R}^n$ and $\mathcal{U} = \mathbb{R}^p$. Let $r : \mathcal{U} \to \mathcal{X}$ be the response function which describes the system behavior (shown in figure 2.1).

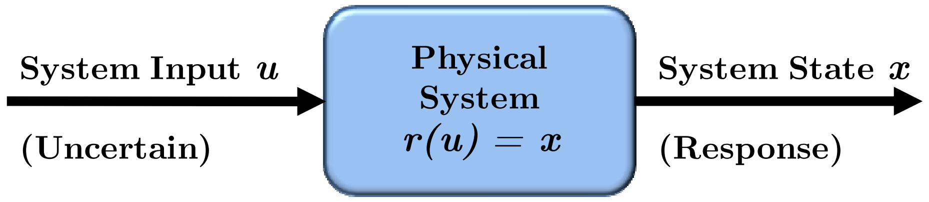


Figure 2.1.: Processing scheme of a system being subject to reliability analysis.

The uncertain system input will be modeled with random vector U which takes values $u \in \mathcal{U}$. This elementary event will be denoted as $U = u$ and $P(U = u)$, abbreviated $P(u)$, will denote the probability that random vector U takes the value u. The probability distribution $p(u)$ of the uncertain input is assumed to be given and together with the deterministic response function r the probability distribution $p(u)$ can also be described over the domain $\mathcal{X}$ of system responses.

The probability for a system failure P_F can now be described as

$$P_F = P(U = u | r(u) \in F) = \int\limits_{u|r(u) \in F} p(u)\, du = \int \mathbb{I}(r(u) \in F)\, p(u)\, du \tag{2.1}$$

where $\mathbb{I} : \{\text{false}, \text{true}\} \to \{0, 1\}$ is an indicator function defined as

$$\mathbb{I}\left(boolean\ expression\right) = \begin{cases} 1 & \text{if } boolean\ expression = \text{true} \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

With the indicator function the failure probability can also be viewed as the expected value of failure indicator function, i.e. $P_F = E\left[\mathbb{I}\left(r\left(U\right) \in F\right)\right]$

The main problem in reliability analysis is, however, that the integral in Equation (2.1) can usually not be efficiently calculated and therefore estimation techniques are often applied, especially for continuous state space variables. In engineering, the state space variables are generally continuous and therefore this thesis focuses its discussions and examples on continuous state space variables, but the following algorithms are also applicable to discrete input or state spaces. A robust and easy way to estimate the failure probability is to generate samples from the probability distribution. Since failure probabilities of interest are generally very small an efficiency problem arises, because common sampling methods are mostly inefficient to estimate small failure probabilities. Even for coarse approximations of the failure probability many samples are required due to the rare occurrences of failure events. Therefore, special algorithms have been developed to decrease the number of samples and therefore the number of system analysis, which are necessary for each sample. Especially, for dynamic systems the system response calculation may require a lot of computational effort and makes an efficient failure probability estimation valuable. In engineering, reliability analysis is generally applied to dynamic systems and more specialized and efficient methods have been found to estimate the failure probability. Some of these methods will be explained in the following sections.

Throughout this thesis the mathematical notation of sets is used frequently. For brevity in notation and to improve readability sets of elements like $\left\{x^1, x^2, \ldots, x^N\right\}$ are sometimes abbreviated by $\left\{x^k\right\}_{k=1}^{N}$. Furthermore, for sets of elements with several changing indexes, the start value will be 1 for all indexes and only the index and the end value will be denoted and aligned in the same way, that is, for example the following set of integer pairs will be abbreviated as $\{(1,1),\ldots,(1,8),\ldots,(6,1),\ldots,(6,8)\} = \{(i,j)\}_{i,j}^{6,8}$.

2.1.2. Dynamic Systems

While some of the methods being described in this thesis, i.e., Monte Carlo simulation, importance sampling, subset simulation, are more general applicable to sampling problems in different domains, subset simulation with splitting and hybrid subset simulation is specialized for reliability analysis of dynamic systems, that is, the first excursion problem.

Assuming a discrete-time state-space model of a causal dynamical system, the reliability of the system is usually investigated for a given time interval $t \in [0, T]$ at discrete time steps $t_l = (l-1) \cdot \Delta t,\ l = 1, \ldots, n_t$, so that there is a total number of $n_t = T/\Delta t + 1$ time steps. Therefore the set of system inputs will be defined as $\mathcal{U} = \mathbb{R}^{p \times n_t}$ and the set of system states $\mathcal{X} = \mathbb{R}^{n \times n_t}$ correspondingly. The uncertain system input U is then defined as a field of random variables

$$U = \{U(t) \ : \ t = t_l, l = 1, \dots, n_t\} \tag{2.3}$$

A particular system input, that is, a realization of $U(t)$ will also be called *excitation* and denoted as $u = \{u(t) : t = t_l, l = 1, \dots, n_t\}$. Its corresponding system state, also called *trajectory*, will be $x = \{x(t) : t = t_l, l = 1, \dots, n_t\}$.

The state of a dynamic causal system can generally be described by

$$x(t_{l+1}) = h\big(x(t_l), u(t_l), t_l\big) \tag{2.4}$$

where $h : \mathbb{R}^{n \times p \times 1} \to \mathbb{R}^n$ is a function which describes the system dynamics.

Using this iterative description of the system dynamics, the response function $r : \mathcal{U} \to \mathcal{X}$ for such a system can be described by

$$\begin{aligned} x &= r\,(u) \qquad (2.5)\\ &= \big\{x(t_0) := x_0,\ x(t) : x(t) = h\big(x(t_{l-1}), u(t_{l-1}), t_{l-1}\big),\ \forall l = 1, 2, \dots, n_t\big\} \end{aligned}$$

The problem to detect a failure of causal dynamic system subjected to uncertain input is also widely known as first passage point problem or first excursion problem, since it is the task to find the first point, where the dynamic system response exceeds a given limit. In reliability analysis, the only question of interest is, whether such a point exists for a given excitation. Since the structural system is now studied over time, the definition of a system failure may also vary over time. In general, it may be necessary to model a time-variant shape of the failure region, that is $F = F\,(t)$. However, the methods explained in this thesis will work in the same way and for the sake of simplicity this possible dependency is dropped in the notation.

2.2. Monte Carlo Simulation

The Monte Carlo simulation (MCS) method provides an easy approach to the given reliability problem and tends to be used often when deterministic methods are impossible or infeasible.

$$P_F = \int\limits_u \mathbb{I}\,(r\,(u) \in F)\, p\,(u)\, du = E\,[\mathbb{I}\,(r\,(U) \in F)] \tag{2.6}$$

$$= \int\limits_x \mathbb{I}\,(x \in F)\, p\,(x)\, dx = E\,[\mathbb{I}\,(X \in F)] \tag{2.7}$$

Assuming to have a set of samples $\left\{u^k\right\}_{k=1}^N$, drawn from $p\,(u)$, the set of samples $\left\{x^k\right\}_{k=1}^N$

which is distributed according to $p(x)$ can be calculated with response function $r(\cdot)$. With the strong law of large numbers the failure probability can also be calculated as

$$P_F = E\left[\mathbb{I}(X \in F)\right] = \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \mathbb{I}\left(r\left(u^k\right) \in F\right) \tag{2.8}$$

Hence, for large numbers N the failure probability can be approximated with arbitrary accuracy by using the estimator

$$\hat{P}_F = \hat{E}\left[\mathbb{I}(X \in F)\right] = \frac{1}{N} \sum_{k=1}^{N} \mathbb{I}\left(x^k \in F\right) \tag{2.9}$$

As commonly known, the MCS estimator $\hat{P}_F$ for independent identically distributed (i.i.d.) samples from $p(u)$ converges to P_F (Law of Large Numbers), is unbiased, consistent, and asymptotically Gaussian (Central Limit Theorem). Its first moments are given as

$$E\left[\hat{P}_F\right] = P_F \tag{2.10}$$

$$Var\left[\hat{P}_F\right] = \frac{(1-P_F)P_F}{N} \tag{2.11}$$

and the coefficient of variation, denoted by δ_{P_F}, is determined by

$$\delta_{P_F} = \frac{\sqrt{Var\left[\hat{P}_F\right]}}{E\left[\hat{P}_F\right]} = \sqrt{\frac{1-P_F}{P_F \cdot N}} \tag{2.12}$$

Equation (2.12) may also be used to find out how many samples are necessary to achieve a given c.o.v. value and it can be seen that the number of required samples is inversely proportional to the probability of failure P_F for small values of P_F.

$$N_\delta = \frac{1-P_F}{P_F \cdot \delta_{P_F}^2} \sim \frac{1}{P_F \cdot \delta_{P_F}^2} \quad \text{for small } P_F \tag{2.13}$$

Common disadvantages of the MCS method are on the one hand its reliance on good random number generators and on the other hand its slow convergence to better approximations when more data points are sampled. However, the main drawback of MCS - used for reliability analysis - is its inefficiency if small probability values need to be calculated, because on average it needs $1/P_F$ samples to obtain one sample which lies in the failure region. Therefore, special methods for failure probability calculation have been developed which tackle this problem by moving more samples toward the failure region.

2.3. Importance Sampling

Independent of the shape of the probability distribution $p\left(x\right)$ regions with low probability density may need many samples before they are being explored with direct Monte Carlo simulation. Conversely, most samples will be generated in regions with high probability density and a single samples does not gain much more information about this region. In view of probability density estimation in low and high probability regions, a less probable sample is much more 'important' because its contribution to the relative frequencies in this region is much higher. Hence, for the problem of estimating small probabilities, it is beneficial for computational efficiency, if samples are generated more frequently in the region of interest.

2.3.1. General Idea

The essential idea of importance sampling [7] is to simulate samples from an importance sampling density (ISD) $f\left(x\right)$ instead of simulating samples directly from the PDF $p\left(x\right)$ of interest. With direct MCS the failure probability is calculated as the expected value of the failure indicator function

$$P_F = E_p\left[\mathbb{I}\left(x \in F\right)\right] = \int_x \mathbb{I}\left(x \in F\right) p\left(x\right) dx \tag{2.14}$$

The failure probability can also be expressed as an expected value with respect to the ISD, if failure probability equation (2.14) is expanded with $f\left(x\right)$.

$$P_F = \int_x \frac{\mathbb{I}\left(x \in F\right) p\left(x\right)}{f\left(x\right)} f\left(x\right) dx = E_f\left[\mathbb{I}\left(x \in F\right) R\left(x\right)\right] \tag{2.15}$$

where

$$R\left(x\right) = \frac{p\left(x\right)}{f\left(x\right)} \tag{2.16}$$

is called the importance sampling quotient. If samples $\left\{x^k\right\}_{k=1}^N$ are drawn from $f\left(x\right)$ instead of $p\left(x\right)$, the failure probability can be estimated by

$$P_F \approx \hat{P}_F = \frac{1}{N}\sum_{k=1}^{N} \mathbb{I}\left(x^k \in F\right) R\left(x\right) \tag{2.17}$$

The meaning of $R\left(x\right)$ can be interpreted as it weights the samples drawn from $f\left(x\right)$ according to their real importance in $p\left(x\right)$ and ensures that the estimator $\hat{P}_F$ in (2.17) remains unbiased. An appropriate choice of the ISD can significantly reduce the estimator variance, or equivalently, the number of necessary samples to obtain a good estimation. However, a bad choice of the ISD can lead to the contrary and the estimator will have a worse variance compared to direct MCS. Therefore, the fundamental issue of importance sampling simulation is a choice of the ISD such that it generates more samples in the region of interest.

In the following, special properties of linear systems are described in order to construct an ISD for efficient failure probability estimation of linear dynamic systems.

2.3.2. Application to Linear Dynamic Systems

Generally, the input-output relation of a linear system can be expressed by the Duhamel integral (or convolution method) [16]. Being only valid for linear systems, the Duhamel integral is based on the principle of superposition, that is, if the system input $u(t)$ is viewed as a series of short-duration impulses, the system response at a particular time t is the sum of all previous impulse responses. Assuming zero initial conditions at time $t = 0$ without loss of generality, the uncertain system output $X_i\left(t\right),\ i = 1, \ldots, n$ can therefore be calculated as

$$X_i\left(t\right) = \sum_{j=1}^{p} \int\limits_0^t h_{ij}\left(t, \tau\right) U_j\left(\tau\right) d\tau \tag{2.18}$$

where $U_j\left(\tau\right)$ is the jth index of the uncertain system input vector at time τ and $h_{ij}\left(t, \tau\right)$ is the linear unit response function (or Green's function) of the system, which returns the ith output at time t, caused by a unit impulse applied to the system at time τ.

Assuming a discrete-time system, the system response is evaluated at discrete time steps $t_l = (l-1) \cdot \Delta t,\ l = 1, \ldots, n_t$ and for small Δt the Duhamel integral can be approximated by a sum over discrete time steps

$$X_i\left(t_l\right) = \sum_{j=1}^{p} \sum_{s=1}^{l} g_{ij}\left(t_l, t_s\right) U_j\left(t_s\right) \Delta t \tag{2.19}$$

where $g_{ij}\left(t_l, t_s\right)$ represents the discrete-time unit response function which tends to $h_{ij}\left(t_l, t_s\right)$ if the sampling interval Δt approaches zero

$$g_{ij}\left(t_l, t_s\right) \rightarrow h_{ij}\left(t_l, t_s\right) \quad \text{if } \Delta t \rightarrow 0 \tag{2.20}$$

given that the numerical integration scheme for the response calculation is consistent and stable [17].

Assume further that the uncertain input vector $U_j\left(t\right)$ is Gaussian white noise with spectral intensity S_j, that is

$$U_j\left(t_l\right) = \sqrt{\frac{2\pi S_j}{\Delta t}} Z_j\left(l\right) \tag{2.21}$$

where $Z_j\left(l\right), j = 1, \ldots, p; l = 1, \ldots, n_t$ are i.i.d. standard Gaussian random variables, which can be arranged to the $p \cdot n_t = q$-dimensional random vector

$$\mathbf{Z} = \left[Z_1\left(1\right), \ldots, Z_p\left(1\right), \ldots, Z_1\left(n_t\right), \ldots, Z_p\left(n_t\right)\right] \tag{2.22}$$

which takes values $\mathbf{z} = [z_1, \ldots, z_q]$. The joint probability distribution of $\mathbf{z}$ is hence

$$p\left(\mathbf{z}\right) = \phi\left(\mathbf{z}\right) \sim \left(2\pi\right)^{-q/2} \exp\left[-\frac{1}{2}\sum_{i=1}^{q} z_i^2\right] \tag{2.23}$$

Combining equations (2.19) and (2.21) leads to

$$X_i\left(t_l\right) = \sum_{j=1}^{p}\sum_{s=1}^{l} g_{ij}\left(t_l, t_s\right) Z_j\left(s\right) \sqrt{2\pi S_j \Delta t} \tag{2.24}$$

Let the system failure for the ith output be defined by the exceedence of the absolute system response $\left|X_i\left(t_l\right)\right|$ over a given threshold b_i at any time step t_l within the duration of study. This will be the elementary failure event in the sample space and will be denoted as

$$F_{il} = \left\{\left|X_i\left(t_l\right)\right| \geq b_i\right\} \tag{2.25}$$

For a better understanding of the elementary failure region this event can further be split into two mutual exclusive events, because F_{il} is the union $F_{il} = F_{il}^+ \cup F_{il}^-$ of the up-crossing failure event $F_{il}^+ = \left\{X_i\left(t_l\right) \geq b_i\right\}$ and the down-crossing failure event $F_{il}^- = \left\{-X_i\left(t_l\right) \geq b_i\right\}$. Since the excitation has been assumed to be symmetrical, these two cases can be treated identically. However, for the non-symmetric case the failure events F_{il}^+ and F_{il}^- have to be handled separately and the extension of the following derivations is straightforward.

Clearly, an overall failure event occurs if at least one elementary event occurs. The failure event F of interest is thus the union of the elementary failure events:

$$F = \bigcup_{i=1}^{n}\bigcup_{l=1}^{n_t} F_{il} \tag{2.26}$$

Applying the definition of the response (2.24) to the definition of F_{il}^+ it can be seen that the failure event is a linear classifier due to the linearity of the response function. The failure boundary ∂F_{il}^+ is given as a hyperplane in the q-dimensional space of $\mathbf{z}$

$$\partial F_{il}^{+} = \left\{ \mathbf{z} : \sum_{j=1}^{p} \sum_{s=1}^{l} g_{ij}\left(t_l, t_s\right) Z_j\left(s\right) \sqrt{2\pi S_j \Delta t} = b_i \right\} \tag{2.27}$$

If the probability $\phi\left(\mathbf{z}\right)$ of the points $\mathbf{z}$ lying in the failure region defined by F_{il}^{+} is taken into account, one can observe that the probability of the points decreases with its distance from the origin, because the PDF $\phi\left(\mathbf{z}\right)$ has been assumed to be radially decreasing. Therefore, there exists a point in F_{il}^{+} which has the highest probability with respect to $\phi\left(\mathbf{z}\right)$. This point is commonly called *design point* and will be denoted as $\mathbf{z}_{il}^{*}$. Intuitively, the design point is the system input at time t_l with the 'smallest energy' to push the response of the system to the limit value b_i [18]. The word energy in this case is meant to be the euclidean distance of the design point to the expected value of the system input (the origin in this case). The design point is simply the point on the boundary ∂F_{il}^{+} with the minimum distance to the origin. Minimizing $\phi\left(\mathbf{z}\right)$ conditioned on ∂F_{il}^{+} delivers the design point $\mathbf{z}_{il}^{*} \in \mathbb{R}^{q}$ for the elementary failure event F_{il}^{+} with

$$\mathbf{z}_{il}^{*} = \left\{ z_{il,j}^{*}\left(1\right), \ldots, z_{il,j}^{*}\left(n_t\right) \right\}_{j=1}^{p} \tag{2.28}$$

and the corresponding values $z_{il,j}^{*}\left(s\right)$ of the jth input at time step s are

$$z_{il,j}^{*}\left(s\right) = H\left(l-s\right) \sqrt{2\pi S_j \Delta t} \frac{g_{ij}\left(l,s\right)}{\sigma_{il}^{2}} b_i \tag{2.29}$$

where $H\left(\cdot\right)$ is the Heaviside step function: $H\left(a\right) = 1$ if $a \geq 0$ and zero otherwise, which only ensures that $z_{il,j}^{*}\left(s\right) = 0 \ \forall s > l$, as the design point at time step l is only influenced by previous time steps due to the assumed causality of the system.

σ_{il}^{2} denotes the variance of the response and can be derived from Equation (2.24):

$$\sigma_{il}^{2} = Var\left[X_i\left(t_l\right)\right] = \sum_{j=1}^{p} \left[\sum_{s=1}^{l} g_{ij}\left(t_l, t_s\right)^{2} \right] 2\pi S_j \Delta t \tag{2.30}$$

With the knowledge of the input variance, one can also easily calculate the euclidean norm of design point $\mathbf{z}_{il}^{*}$, because its calculation simplifies to

$$\beta_{il} = \left\| \mathbf{z}_{il}^{*} \right\| = \frac{b_i}{\sigma_{il}} \tag{2.31}$$

where β_{il} is often called the 'reliability index' and represents intuitively the limit value b_i in the probability space of $\mathbf{z}$.

By definition, the design point with the smallest reliability index within the duration of study is called the *global design point*.

2.3.3. Construction of an Efficient ISD for Linear Dynamic Systems

A simple method to construct an ISD is to use only the global design point, because this is the point with the highest probability in the input domain, which certainly leads to a system failure. The importance sampling density $f(\mathbf{z})$ could then simply be the input PDF $\phi(\mathbf{z})$ centered at the global design point $\mathbf{z}^*$: $f(\mathbf{z}) = \phi(\mathbf{z} - \mathbf{z}^*)$. With this ISD the samples drawn from $f(\mathbf{z})$ will have higher probability to lie in the failure region and sampling from the ISD will improve the efficiency of the failure probability estimator. However, the knowledge about the design point at each time step can be used to generate much better sampling densities.

To see this, one can investigate the contribution of the failure probability of a single failure event F_{il} to the overall failure probability. The conditional probability ratio of any two elementary failure events F_{il} and F_{js}, given that the failure occurs is:

$$\frac{P(F_{il}|F)}{P(F_{js}|F)} = \frac{P(F_{il} \cap F)/P(F)}{P(F_{js} \cap F)/P(F)} = \frac{P(F_{il})}{P(F_{js})} \tag{2.32}$$

Therefore, the relative contribution of an elementary failure event F_{il} to the overall failure event F may be measured only by their unconditional probability

$$P(F_{il}) = 2\Phi(-\beta_{il}) \tag{2.33}$$

where $\Phi(\cdot)$ is the cumulative distribution function of the standard Gaussian distribution. Equation 2.33 is due to the fact that $\mathbf{z}$ is assumed to be Gaussian distributed and the probability content of F_{il} is the sum of the probability contents of F_{il}^+ and F_{il}^-, which in turn are equal due to the symmetry of $\phi(\mathbf{z})$.

$$P\left(F_{il}^+\right) = \Phi\left(-\left\|\mathbf{z}_{il}^*\right\|\right) = \Phi(-\beta_{il}) \tag{2.34}$$

The probability content of F_{il} is therefore completely determined by the reliability index β_{il}. Certainly, the global design point has the highest probability, but the contribution of the other design points should also be considered in order to construct a well suited ISD. Empirical studies have shown (see [11]), that neighboring design points from consecutive time steps have similar probabilities and therefore especially the design points in the neighborhood of the global design point may also contribute significantly to the overall failure probability.

To take the importance of all design points into account the ISD can be constructed as a weighted sum of individual densities which favor a particular design point

$$f(\mathbf{z}) = \sum_{i=1}^{n} \sum_{l=1}^{n_t} w_{il} f_{il}(\mathbf{z}) \tag{2.35}$$

with the weight conditions

$$w_{il} \geq 0 \quad \text{and} \quad \sum_{i=1}^{n} \sum_{l=1}^{n_t} w_{il} = 1 \tag{2.36}$$

The weights are introduced to determine the contribution of each individual ISD $f_{il}(\mathbf{z})$ to the overall ISD $f(\mathbf{z})$. According to the property derived in Equation (2.32) the weights w_{il} can be chosen to be proportional to the probability content of F_{il}:

$$w_{il} = \frac{P(F_{il})}{\sum_{j=1}^{n} \sum_{s=1}^{n_t} P(F_{js})} = \frac{\Phi(-\beta_{il})}{\sum_{j=1}^{n} \sum_{s=1}^{n_t} \Phi(-\beta_{js})} \tag{2.37}$$

Since the weight w_{il} determines the relative frequency of the samples simulated from $p(\mathbf{z}|F_{il})$, the samples drawn from $p(\mathbf{z}|F_{il})$ will have relative frequencies proportional to the importance of the design point $\mathbf{z}_{il}^*$.

If the individual ISDs $f_{il}(\mathbf{z})$ have the important property that samples can be efficiently generated according to $f_{il}(\mathbf{z})$, one can also efficiently generate samples from $f(\mathbf{z})$, by first drawing a random ordered pair (I, L) from $\{(i,l)\}_{i,l}^{n,n_t}$ according to their corresponding probabilities $\{w_{il}\}_{i,l}^{n,n_t}$, and then drawing a sample from $f_{IL}(\mathbf{z})$.

According to Equation (2.17) the failure probability can then be estimated by

$$\hat{P}_F = \frac{1}{N} \sum_{k=1}^{N} \frac{\phi(\mathbf{z}^k)\, \mathbb{I}\left(r(\mathbf{z}^k) \in F\right)}{\sum_{i=1}^{n} \sum_{l=1}^{n_t} w_{il} f_{il}(\mathbf{z}^k)} \tag{2.38}$$

where $\{\mathbf{z}^k\}_{k=1}^{N}$ are N samples drawn from $f(\mathbf{z})$. For simplicity of notation, the function $r(\mathbf{z})$ is meant to transform the vector $\mathbf{z}$ into the domain of x. Since $r(\cdot)$ is defined over the domain of u, this implicitly means that $\mathbf{z}$ is first transformed to u by Equation (2.21) and then the result is applied to $r(\cdot)$. Two different ISDs using the properties described above will be discussed in following.

2.3.3.1. Weighted Sum of Gaussians (ISD A)

Refining the first idea at the beginning of this section, the ISD can also be constructed as a weighted sum of Gaussian PDFs centered at the individual design points, rather than using a single ISD centered at the global design point. This approach is a common choice of the ISD (e.g. [9]) and is described by

$$f_A(\mathbf{z}) = \sum_{i=1}^{n} \sum_{l=1}^{n_t} w_{il} \phi(\mathbf{z} - \mathbf{z}_{il}^*) \tag{2.39}$$

The weights w_{il} are chosen according to Equation (2.37). With samples $\{\mathbf{z}^k\}_{k=1}^{N}$ drawn from $f_A(\mathbf{z})$ according to Equation (2.39) the failure probability P_F can be estimated by

$$\hat{P}_F = \frac{1}{N}\sum_{k=1}^{N}\frac{\mathbb{I}\left(r\left(\mathbf{z}^k\right)\in F\right)\phi\left(\mathbf{z}^k\right)}{\sum_{i=1}^{n}\sum_{l=1}^{n_t} w_{il}\cdot\phi\left(\mathbf{z}^k-\mathbf{z}_{il}^*\right)} \tag{2.40}$$

The evaluation of the estimator in this form is not efficient and appendix A.2.1 shows how to implement the estimator efficiently. A part of it can be used in both described importance sampling methods.

2.3.3.2. Weighted Sum of Conditional PDFs (ISD B)

S.K. Au et al. [11] proposed an ISD which is more efficient than the aforementioned ISD A. They suggest generating samples from the conditional PDF $p\left(\mathbf{z}|F_{il}\right)$, that is, the probability of $\mathbf{z}$ with respect to $\phi\left(\mathbf{z}\right)$, given that $\mathbf{z}$ lies in the elementary failure region F_{il}. Put into words, the conditional PDF $p\left(\mathbf{z}|F_{il}\right)$ is the original PDF $\phi\left(\mathbf{z}\right)$ confined to F_{il} and normalized by the probability content of F_{il}:

$$p\left(\mathbf{z}|F_{il}\right) = \frac{\phi\left(\mathbf{z}\right)\mathbb{I}\left(r\left(\mathbf{z}\right)\in F\right)}{P\left(F_{il}\right)} = \frac{\phi\left(\mathbf{z}\right)\mathbb{I}\left(r\left(\mathbf{z}\right)\in F\right)}{2\Phi\left(-\beta_{il}\right)} \tag{2.41}$$

Using the definition of the conditional PDF in Equation (2.41) to construct an ISD as weighted sum of conditional PDFs yields

$$f_B\left(\mathbf{z}\right) = \sum_{i=1}^{n}\sum_{l=1}^{n_t} w_{il}\cdot p\left(\mathbf{z}|F_{il}\right) = \sum_{i=1}^{n}\sum_{l=1}^{n_t} w_{il}\frac{\phi\left(\mathbf{z}\right)\mathbb{I}\left(r\left(\mathbf{z}\right)\in F_{il}\right)}{2\Phi\left(-\beta_{il}\right)} \tag{2.42}$$

where the weights w_{il} again fulfill the conditions (2.36) of a probability distribution and are chosen according to Equation (2.37).

Applying the definition of the weights to the last equation, the ISD is given as

$$f_B\left(\mathbf{z}\right) = \frac{\phi\left(\mathbf{z}\right)}{2\sum_{j=1}^{n}\sum_{s=1}^{n_t}\Phi\left(-\beta_{js}\right)}\sum_{i=1}^{n}\sum_{l=1}^{n_t}\mathbb{I}\left(r\left(\mathbf{z}\right)\in F_{il}\right) \tag{2.43}$$

Since the denominator in Equation (2.43) is an independent factor it will also be independent of the sum over the samples if Equation (2.43) is substituted in the Equation (2.38) for the failure probability estimator:

$$\hat{P}_F = 2\sum_{j=1}^{n}\sum_{s=1}^{n_t}\Phi\left(-\beta_{js}\right)\cdot\frac{1}{N}\sum_{k=1}^{N}\frac{\mathbb{I}\left(r\left(\mathbf{z}^k\right)\in F\right)}{\sum_{i=1}^{n}\sum_{l=1}^{n_t}\mathbb{I}\left(r\left(\mathbf{z}^k\right)\in F_{il}\right)} \tag{2.44}$$

The factor

$$\bar{P}_F = 2\sum_{j=1}^{n}\sum_{s=1}^{n_t}\Phi\left(-\beta_{js}\right) \tag{2.45}$$

in Equation (2.44) is independent of the sampling process and can be interpreted as the upper bound for the failure probability. Furthermore, it is known that the samples for the estimation are drawn from $p(\mathbf{z}|F_{il})$, which means that every sample lies in the failure region and $\mathbb{I}\left(r\left(\mathbf{z}^k\right) \in F\right)$ in Equation (2.44) will always be 1. Thus, the failure probability estimator is given by

$$\hat{P}_F = \bar{P}_F \cdot \frac{1}{N} \sum_{k=1}^{N} \frac{1}{\sum_{i=1}^{n} \sum_{l=1}^{n_t} \mathbb{I}\left(r\left(\mathbf{z}^k\right) \in F_{il}\right)} \tag{2.46}$$

where $\left\{\mathbf{z}^k\right\}_{k=1}^{N}$ are independent samples drawn from $f_B(\mathbf{z})$ according to Equation 2.43. The resulting estimator therefore evaluates the number of time steps in which the response of the system to sample $\mathbf{z}^k$ is above the defined threshold b_i. The evaluation of the failure probability (2.46) is simple and computational inexpensive, because the Gaussian PDF of the input and the cumulative distribution of the Gaussian PDF has been canceled out due to the choice of the conditional PDFs $p(\mathbf{z}|F_{il})$ for the construction of the ISD. The more expensive calculation of the factor $\bar{P}_F$ is independent of the sampling process and can be calculated in advance.

Efficient Sampling from the Conditional PDF

To efficiently simulate samples from $p\left(\mathbf{z}|F_{il}^{+}\right)$ one can use the knowledge about the corresponding design point $\mathbf{z}_{il}^{*}$. Any vector $\mathbf{z}^k \sim p\left(\mathbf{z}|F_{il}^{+}\right)$ can be represented by a vector $\alpha\mathbf{u}_{il}^{*}$ which points in direction of the design point and a second vector $\mathbf{z}_{il}^{\perp}$, which is perpendicular to the first one, i.e., parallel to the hyperplane ∂F_{il}^{+}:

$$\mathbf{z}^k = \alpha\mathbf{u}_{il}^{*} + \mathbf{z}_{il}^{\perp} \tag{2.47}$$

where

$$\mathbf{u}_{il}^{*} = \mathbf{z}_{il}^{*} / \left\|\mathbf{z}_{il}^{*}\right\| = \mathbf{z}_{il}^{*} / \beta_{il} \tag{2.48}$$

is the unit vector in direction of the design point and α is a standard Gaussian random variable conditional on $\{\alpha \geq \beta_{il}\}$ with probability $p(\alpha) = \phi(\mathbf{z}) H(\alpha - \beta_{il}) / \Phi(-\beta_{js})$. The vector $\mathbf{z}_{il}^{\perp}$ which is parallel to ∂F_{il}^{+} can in turn be represented as

$$\mathbf{z}_{il}^{\perp} = \mathbf{z} - \left\langle \mathbf{z}, \mathbf{u}_{il}^{*} \right\rangle \mathbf{u}_{il}^{*} \tag{2.49}$$

where $\mathbf{z}$ is a q-dimensional standard

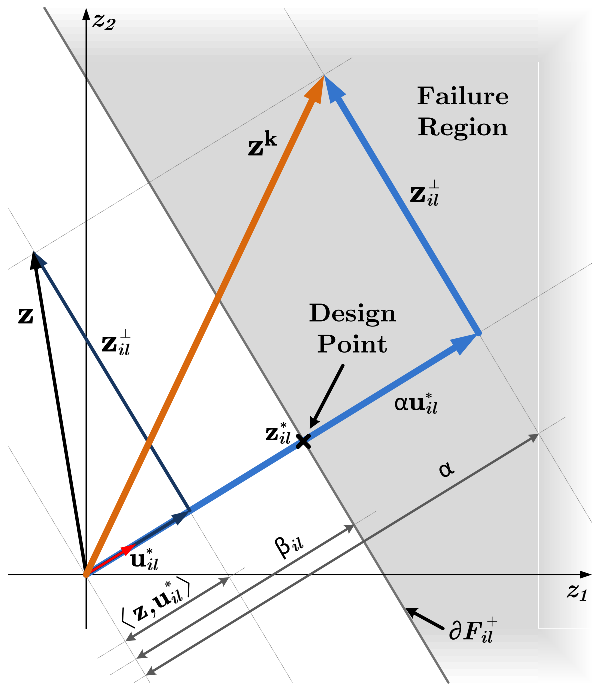


Figure 2.2.: Generation of samples from the ISD shown for the first two time steps: a sample $\mathbf{z}^k \sim p\left(\mathbf{z}|F_{il}^{+}\right)$ is generated using the design point $\mathbf{z}_{il}^{*}$ and a sample $\mathbf{z} \sim \phi(\mathbf{z})$.

Gaussian random vector. Combining Equations (2.47) and (2.49) leads to

$$\mathbf{z}^k = \mathbf{z} + \left(\alpha - \langle \mathbf{z}, \mathbf{u}_{il}^* \rangle\right) \mathbf{u}_{il}^* \tag{2.50}$$

which facilitates efficient simulation of samples $\mathbf{z}^k \sim p\left(\mathbf{z}|F_{il}^+\right)$, because one only needs a q-dimensional random Gaussian $\mathbf{z} \sim \phi\left(\mathbf{z}\right)$ which can be generated independently for each dimension and a one-dimensional random Gaussian α conditional on $\{\alpha \geq \beta_{il}\}$. This can be obtained by simulating a uniform number between $\Phi\left(\beta_{il}\right)$ and 1 and applying the inverse cumulative Gaussian distribution function to it. Given a uniform sample $a \sim U\left[0, 1\right]$ the conditional Gaussian α can be calculated as

$$\alpha = \Phi^{-1}\left(a + (1-a)\Phi\left(\beta_{il}\right)\right) \tag{2.51}$$

Note, that Figure 2.2 only shows half of the failure region for the up-crossing failure events, the other failure region, its limiting hyperplane and the the design point for the down-crossing failure events are in a symmetric position with respect to the coordinate origin. Therefore, the simulation of samples $\mathbf{z}^+ \sim p\left(\mathbf{z}|F_{il}^+\right), \mathbf{z}^- \sim p\left(\mathbf{z}|F_{il}^-\right)$ is analogous, because $-\mathbf{z}^+$ and $\mathbf{z}^-$ are identically distributed.

2.3.3.3. Comparision of the ISDs

It has already been mentioned, that the estimator ISD B is computationally efficient, since the probability $\bar{P}_F$ is independent of the sampling process and can be calculated in advance. The only necessary operation during the sampling process is simply counting the number of time steps where the response is above the threshold. In comparison, to estimate the failure probability using ISD A the evaluation of the term $\phi\left(\mathbf{z}^k\right) / \sum_{i=1}^{n} \sum_{l=1}^{n_t} w_{il} \cdot \phi\left(\mathbf{z}^k - \mathbf{z}_{il}^*\right)$ is necessary for each sample and includes the calculation of $O\left(n \cdot n_t\right)$ exponential functions. In view of the sampling process, ISD B therefore allows computationally more efficient simulation, since the estimator is much simpler. Furthermore, the shape of the ISDs compared to the original PDF reveals that ISD B is also more efficient with respect to the estimator c.o.v., because it generates all samples in the failure region. Figure 2.3 illustrates the differences between the shapes of the ISDs.

Figure 2.3(a) shows the probability distribution of interest together with the failure region for the first dimension of $\mathbf{z}$. Figure 2.3(b) shows ISD A where the Gaussians have been moved to the elementary design points. During the sampling process one of the two distributions (shown in blue) centered at the up-crossing and down-crossing design points is selected with probability 0.5. Then, the sample is drawn from the PDF centered at the design point. Due to this modification the samples from $f_A\left(\mathbf{z}\right)$ will have much more probability to lie in the failure region than the samples directly drawn from $\phi\left(\mathbf{z}\right)$. More exactly, any simulated sample $\mathbf{z}^k \sim f_A\left(\mathbf{z}\right)$ will have a probability $\geq 1/2$ to lie in the failure region F, because the probability for a sample to be generated in the elementary failure region F_{il} is $1/2 + \Phi\left(-2\beta_{il}\right)$. Hence the overall probability for a sample to be generated in the failure region is given by

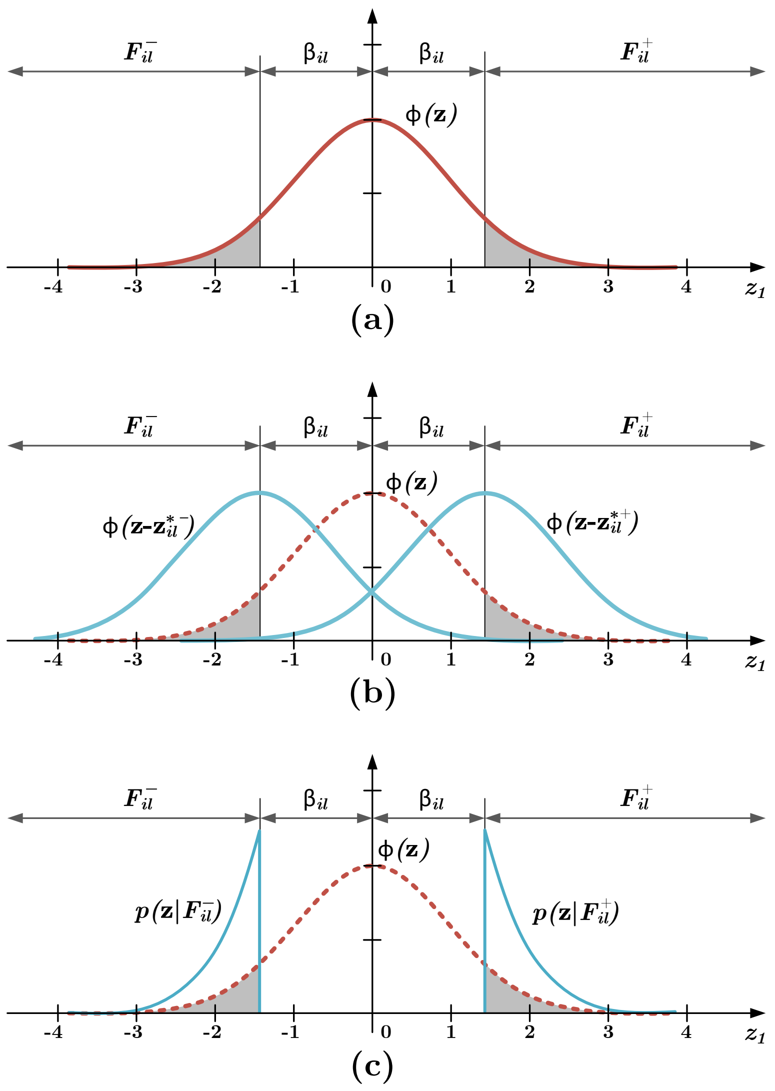


Figure 2.3.: Schematical comparison of the two different choices for the ISD. (a) PDF of interest and failure region (gray) (b) ISD A: weighted sum of Gaussians (c) ISD B: weighted sum of conditional PDFs

$$P\left(r\left(\mathbf{z}^k\right) \in F\right) = \sum_{i=1}^{n}\sum_{l=1}^{n_t} w_{il}\left(\frac{1}{2} + \Phi\left(-2\beta_{il}\right)\right) \tag{2.52}$$

For very small failure probabilities the values of $\Phi\left(-2\beta_{il}\right)$ will get closer to zero and thus

$$P\left(r\left(\mathbf{z}^k\right) \in F\right) \approx \frac{1}{2}\sum_{i=1}^{n}\sum_{l=1}^{n_t} w_{il} = \frac{1}{2} \tag{2.53}$$

Since $\Phi\left(-2\beta_{il}\right)$ is always greater than zero, $1/2$ can be viewed as a lower bound for this probability. In comparison with the simulation from the original PDF this is a significant improvement. However, Figure 2.3(c) illustrates that the ISD constructed with the conditional PDF $p(\mathbf{z}|F_{il})$ is more efficient, because the shape of ISD B is proportional to $\phi(\mathbf{z})$ and thus the importance of the samples from ISD B reflects the shape of $\phi(\mathbf{z})$. Moreover, ISD B is only defined over the failure region, so every sample from the ISD will lie in F which improves the efficiency of the estimation remarkably.

Note, that Figure 2.3 is a scheme to show the idea behind the ISDs. Since the failure probabilities are usually very small the failure region is much smaller than shown in the figure. Also, the proportions between $\phi(\mathbf{z})$ and the conditional probabilities $p\left(\mathbf{z}|F_{il}^{+}\right), p\left(\mathbf{z}|F_{il}^{-}\right)$ are incorrect in Figure 2.3(c) as all of them are probability distributions which sum up to 1 over the input domain.

2.3.4. Importance Sampling Algorithm for Linear Dynamic Systems

Algorithm 1 Importance Sampling Simulation for Linear Systems.

```
- Given is the system response function $r(\cdot)$ in form of the Duhamel integral eq.(2.19)
  with unit response functions $g_{ij}(l,s)$. The system input is Gaussian: $\mathbf{z} \sim \phi(\mathbf{z})$.
- Output is the probability estimation $\hat{P}_F$ for a system failure.

1. System analysis: Calculate unit impulse response functions $\{g_{ij}(l,s)\}_{i,j,l,s}^{n,p,n_t,n_t}$
2. Precalculation of system properties: Calculate
   - response standard deviations $\{\sigma_{il}\}_{i,l}^{n,n_t}$ by equation (2.30)
   - elementary reliability indexes $\{\beta_{il}\}_{i,l}^{n,n_t}$ by $\beta_{il} = b_i/\sigma_{il}$
   - ISD weights $\{w_{il}\}_{i,l}^{n,n_t}$ by equation (2.37)
3. Sampling and failure probability estimation
   Simulate $N$ i.i.d. samples $\left\{\mathbf{z}^k\right\}_{k=1}^{N}$ drawn from $f_A(\mathbf{z})$ / $f_B(\mathbf{z})$
   - Randomly select an output and time step index pair $(I,L)$ from $\{(i,l)\}_{i,l}^{n,n_t}$
     according to their corresponding probabilities $\{w_{il}\}_{i,l}^{n,n_t}$
   - Draw a sample $\mathbf{z}^k$ from the ISD $f_{IL}(\mathbf{z})$:
     Calculate design point $\mathbf{z}_{il}^*$ by equation (2.28)
     ISD A: Simulate $q$-dimensional random Gaussian $\mathbf{z}^k \sim \phi(\mathbf{z} - \mathbf{z}_{il}^*)$
     ISD B:
         Simulate a $q$-dimensional standard Gaussian vector $\mathbf{z}$ with i.i.d. components
         Simulate uniform random samples $a_1, a_2$ from $U[0,1]$ and calculate

         $\alpha = \Phi^{-1}(a_1 + (1-a_1)\Phi(\beta_{il}))$ and $\mathbf{u}_{il}^* = \mathbf{z}_{il}^*/\beta_{il}$
         $\mathbf{z}^k = \begin{cases} \mathbf{z} + (\alpha - \langle \mathbf{z}, \mathbf{u}_{il}^* \rangle)\,\mathbf{u}_{il}^* & \text{if } a_2 \le 1/2 \\ -\mathbf{z} - (\alpha - \langle \mathbf{z}, \mathbf{u}_{il}^* \rangle)\,\mathbf{u}_{il}^* & \text{otherwise} \end{cases}$

   Estimate failure probability $P_F$ by equation (2.40) (for ISD A)
   or by equation (2.46) (for ISD B)
```

Note, that for ISD A the precalculation of the design points should be considered, because all of them are needed for each sample that hits the failure region in order to calculate the probability estimator. Certainly, this memory and computation speed trade-off depends on the system under study and the available hardware. In contrast, for ISD B this precalculation is additional computation and memory effort and is only meaningful, if the number of samples reaches the number of design points for the system $(n \cdot n_t)$, which is usually not the case for high dimensional systems.

Finally, note that appendix A.2.1 gives suggestions for the implementation in order to improve the efficiency of the importance sampling method.

2.3.5. Statistical Properties of the Method

In general, importance sampling can be a very powerful tool to increase the efficiency of the failure probability estimation, if knowledge about the system of interest is given and applicable to construct an appropriate ISD.

Especially for the case of a linear response function the knowledge about the comparatively simple shape of the failure region can be used to construct an ISD which allows very efficient failure probability estimation. The importance sampling estimator in equation (2.15) is unbiased for an appropriate choice of the ISD as the importance sampling estimator is simply the original estimator extented with the ISD. A fundamental requirement is similarity of the support regions. The coefficient of variation δ_{IS} of the estimator is given as

$$\delta_{IS} = \frac{\Delta_{IS}}{\sqrt{N}} \tag{2.54}$$

where N is the number of samples used for the estimation and Δ_{IS} is the unit c.o.v. of the importance sampling estimator

$$\Delta_{IS} = \frac{Var_f\left[\mathbb{I}\left(x \in F\right) R\left(x\right)\right]}{E_f\left[\mathbb{I}\left(x \in F\right) R\left(x\right)\right]} = \frac{Var_f\left[\mathbb{I}\left(x \in F\right) R\left(x\right)\right]}{P_F} \tag{2.55}$$

where the second quotient assumes unbiasedness of the estimator.

Note, that although the input distribution has been assumed to be Gaussian the method will generally work for symmetrical decreasing distribution functions, but the (efficient) sampling from conditional PDF may be more difficult.

Properties of ISD B and the Estimator

Since all parts of the failure region contribute to the ISD, the estimator (2.46) is unbiased, which can shown if the definition of $p\left(\mathbf{z}|F_{il}\right)$ in equation (2.41) is applied to the failure probability definition based on importance sampling in equation (2.17). Then, by drawing the factor $\bar{P}_F$ in front of the integral (similar to eq.(2.44)) the failure probability is given as

$$P_F = \bar{P}_F \cdot E_f\left[\frac{1}{\sum_{i=1}^{n}\sum_{l=1}^{n_t}\mathbb{I}\left(r\left(\mathbf{z}^k\right)\in F_{il}\right)}\right] \tag{2.56}$$

which can be used to show that $E_f\left[\hat{P}_F\right] = P_F$.

Generally, the choice of the ISD to be the original PDF conditional on the failure event is the optimal choice of ISD in sense that it minimizes the variance of the failure probability estimator [4]. According to equation (2.15) the optimal ISD is

$$f_{opt}\left(x\right) = \arg\inf_{f} Var_f\left[\frac{\mathbb{I}\left(x\in F\right)p\left(x\right)}{f\left(x\right)}\right] = p\left(x|F\right) = \frac{\mathbb{I}\left(x\in F\right)p\left(x\right)}{P_F} \tag{2.57}$$

Since ISD B is constructed using the elementary conditional failure probabilities $p\left(\mathbf{z}|F_{il}\right)$ in a weighted mixture, the ISD is close to the optimal ISD $p\left(\mathbf{z}|F\right)$. Indeed, the constructed ISD B is optimal for the case where the elementary failure events $\{F_{il}\}_{i,l}^{n,n_t}$ are mutually exclusive for all samples, that is, $\mathbb{I}\left(r\left(\mathbf{z}\right)\in F_{il}\right) = 1$ for only one pair (i,l) and $\mathbb{I}\left(r\left(\mathbf{z}\right)\in F_{j,s}\right) = 0$ for $j \neq i$ or $s \neq l$. Then,

$$\sum_{i=1}^{n}\sum_{l=1}^{n_t}\mathbb{I}\left(r\left(\mathbf{z}^k\right)\in F_{il}\right) = 1 \tag{2.58}$$

for all samples and applied to failure probability estimator (2.46) this yields

$$P_F = \hat{P}_F = \bar{P}_F \tag{2.59}$$

Furthermore, due to the choice of the conditional PDFs as $p\left(\mathbf{z}|F_{il}\right)$, the support region of the ISD grows and shrinks in the same way as the elementary failure region, if the limit value b_i is varied (see also Figure 2.3 c). As a result of this property, the smaller the failure region is, the more efficient is the estimator based on ISD B, because a smaller failure region needs less samples to be explored. Thus the estimator is perfectly suited to estimate small probabilities, while on the other hand, the estimator will be less efficient for larger failure probabilities.

2.4. Subset Simulation with Markov Chain Monte Carlo

Similar to the importance sampling approach the idea of subset simulation is to generate more samples within the failure region, but - in contrast to importance sampling - knowledge about the shape of the failure region is not necessary.

2.4.1. Basic Concept

The fundamental idea behind subset simulation is to replace the simulation of rare failure events by a sequence of simulations of more frequent events. To achieve this, the simulation for failure region F is replaced by an iterative scheme of simulations for a nested sequence of failure regions $F_1 \supset F_2 \supset \ldots \supset F_m = F$, such that $F_k = \bigcap_{i=1}^{k} F_i,\ k = 1, \ldots, m$. In other words, the failure region is partitioned in an increasing sequence of subsets.

For instance, the failure region can be defined as the exceedance of an engineering demand parameter over a given capacity, that is, $F = \{x \in \mathcal{X} : g(x) \geq b\}$, where $g : \mathbb{R}^n \to \mathbb{R}$ maps a system state to an engineering demand parameter. Thus, the nested sequence of failure regions can be defined as $F_i = \{x \in \mathcal{X} : g(x) \geq b_i\}$ with the sequence of increasing limit values $b_1 < b_2 < \ldots < b_m = b$.

Let for convenience F denote both, the failure event and its corresponding failure region in the domain of system states. Then, with the definition of the conditional probabilities the failure probability can be defined as a sequence of conditional failure probabilities:

$$\begin{aligned} P_F = P(F_m) &= P\left(\bigcap_{i=1}^{m} F_i\right) \\ &= P\left(F_m \middle| \bigcap_{i=1}^{m-1} F_i\right) P\left(\bigcap_{i=1}^{m-1} F_i\right) \\ &= P(F_m|F_{m-1})\, P\left(\bigcap_{i=1}^{m-1} F_i\right) = \ldots \\ &= P(F_1) \prod_{i=2}^{m} P(F_i|F_{i-1}) \end{aligned} \tag{2.60}$$

Equation (2.60) shows, that the definition of the failure probability defined over the nested sequence of failure regions defines a first order Markov chain. This property will be used later on to for efficient sample generation. Now it remains to calculate all the probabilities for each failure region. The first one, $P(F_1)$, is independent of other failure regions and can be estimated by direct MCS:

$$P(F_1) \approx \hat{P}_1 = \frac{1}{N_1} \sum_{k=1}^{N_1} \mathbb{I}\left(x^{(1,k)} \in F_1\right) = \frac{R_1}{N_1} \tag{2.61}$$

where $\left\{x^{(1,k)}\right\}_{k=1}^{N_1}$ is the set of samples for the first failure region and each of them is distributed according to $p(x)$ and R_i is the number of samples lying in F_i. For the conditional failure probabilities, the same estimator is used:

$$P(F_i|F_{i-1}) \approx \hat{P}_i = \frac{1}{N_i} \sum_{k=1}^{N_i} \mathbb{I}\left(x^{(i,k)} \in F_i\right) = \frac{R_i}{N_i} \tag{2.62}$$

but now each of the samples $\left\{x^{(i,k)}\right\}_{k=1}^{N_i}$ has to be distributed according to the conditional PDF $p(x|F_{i-1})$. This can be achieved efficiently using a Markov chain Monte Carlo method based on the Metropolis algorithm, which will be explained in the next subsection.

A remaining problem is to determine the partial failure regions F_i, or equivalently, the limit values b_i appropriately. For efficient estimation of failure probabilities it is favorable, that all conditional failure probabilities approximately have the same size, such that the amount of information coming from each failure region is roughly the same. Since it is difficult to determine the limit values b_i a priori and to satisfy this desire at the same time, S.K. Au [4] suggested to choose a particular value p_0 for the conditional failure probabilities instead:

$$\hat{P}_i = p_0 \qquad \forall i = 1, \ldots, m-1 \tag{2.63}$$

where a value of $p_0 = 0.1$ has been found to yield good efficiency [4, p.95]. Note, that the last conditional failure probability $\hat{P}_m$ is calculated according to the estimator (2.62) for the given limit value $b_m = b$. The other limit values are found after ordering the samples $\left\{x^{(i,k)}\right\}_{k=1}^{N_i}$ in ascending order and are calculated as:

$$b_i = g\left(x^{(i,j)}\right) \qquad \text{with } j = \lfloor (1-p_0) \cdot N_i \rfloor, \quad \forall i = 1, \ldots, m-1 \tag{2.64}$$

The limit value b_i defines the intermediate failure region F_i, which is the sampling domain for the next stage. The process of reopening further subsets of failure regions can be repeated, as long as the failure event of interest lies within the current intermediate failure region, i.e. $b_i < b$.

The final estimator for the failure probability, which will be calculated according to Equation (2.60), can hence also be calculated as a power of p_0 (due to Equation 2.63):

$$\hat{P}_F = \prod_{i=1}^{m} \hat{P}_i = p_0^{m-1} \hat{P}_m \tag{2.65}$$

Roughly speaking, Equation (2.65) can be interpreted in a way that the number $m-1$ of failure regions defines 'the order' of the failure probability with respect to p_0 and only the conditional failure probability of the last failure region determines its quantity. The advantage of this approach is that all failure regions F_i can be determined by only one parameter p_0. On the other hand, a disadvantage is that the number m of subsets (failure regions) is not known a priori and therefore the total number of samples is difficult to estimate in advance. Finally, the subset simulation procedure can be summarized in the the following algorithm.

Algorithm 2 subset simulation with MCMC.

```
- Given is the system response function $r(\cdot)$ and the probability distribution $p(u)$ for
  the system input.
- Output is the probability estimation $\hat{P}_F$ for a system failure.

A. Stage 1: Generate root samples using direct MCS
   $i := 1$
   Simulate $N_1$ independent input samples $\left\{u^{(1,k)}\right\}_{k=1}^{N_1}$ according to $p(u)$ and
   calculate response samples $\left\{x^{(1,k)}\right\}_{k=1}^{N_1}$ using response function $r(\cdot)$.
   Sort samples $\left\{x^{(1,k)}\right\}_{k=1}^{N_1}$ in ascending order wrt. $g\left(x^{(1,k)}\right)$
   Calculate first limit value: $b_1 := g\left(x^{(1,j)}\right)$ with $j = \lfloor(1-p_0)\cdot N_1\rfloor$,  $R_1 := p_0 N_1$

B. Stage $i$: Estimate $P_i$ (i=1,...,m-1) and possibly calculate new subsets recursively:
   while ($b_i < b$)                                        ▷ a new subset can be opened up
      $\hat{P}_i = p_0$                                      ▷ probability is predefined by $b_i$
      $R_i$ samples exceeded threshold $b_i$ and
      are assigned as mother samples $\left\{x_M^{(i+1,k)}\right\}_{k=1}^{R_i}$
      $i := i+1$                                            ▷ open up a new subset
      Generate $N_i - R_{i-1}$ offspring samples using the mother samples from
      the previous stage, to have samples $\left\{x^{(i,k)}\right\}_{k=1}^{N_i}$ distributed as $p(x|F_{i-1})$
      Sort samples $\left\{x^{(i,k)}\right\}_{k=1}^{N_i}$ in ascending order wrt. $g\left(x^{(i,k)}\right)$
      $b_i := g\left(x^{(i,j)}\right)$ with $j = \lfloor(1-p_0)\cdot N_i\rfloor$,  $R_i := p_0 N_i$
   end while
   The last subset has been reached and $b_i$ is given as $b_i = b_m = b$ $(m := i)$
   $\hat{P}_m = \frac{R_m}{N_m}$, where $R_m$ is the number of samples exceeding threshold $g(x) \geq b$.

C. Finally, calculate the failure probability estimation: $\hat{P}_F = \prod_{i=1}^{m} \hat{P}_i$
```

The recursive subset simulation procedure is illustrated in Figure 2.4 and shows which samples are used as mother samples for the following stage and that the recursion ends if the failure event of interest fails to lie within the next intermediate failure region. This also clarifies why the subset simulation procedure only yields advantage for small failure probabilities.

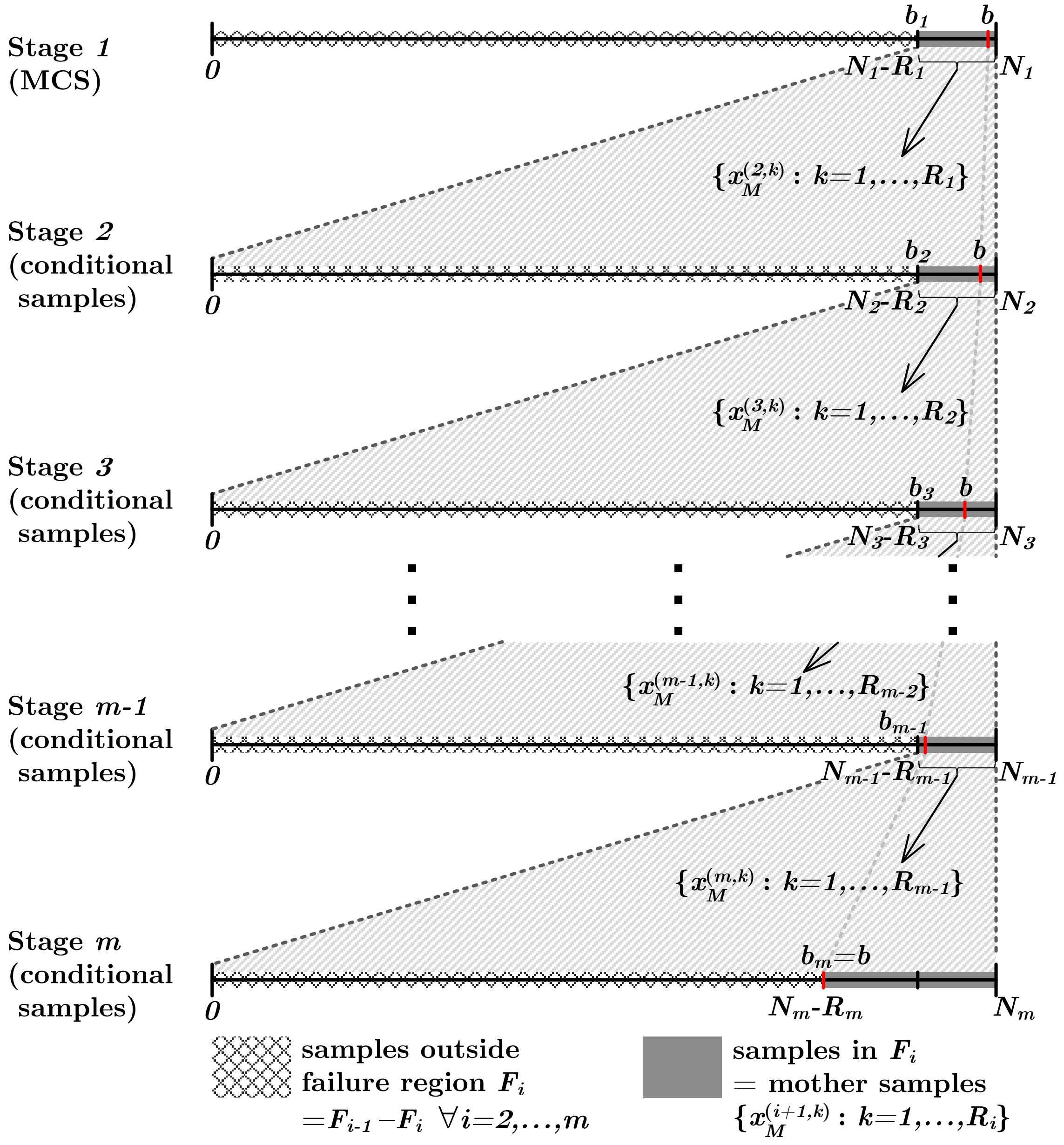

Stage 1 (MCS)
0
b_1
b
N_1-R_1
N_1
$\{x_M^{(2,k)} : k=1,\ldots,R_1\}$
Stage 2 (conditional samples)
0
b_2
b
N_2-R_2
N_2
$\{x_M^{(3,k)} : k=1,\ldots,R_2\}$
Stage 3 (conditional samples)
0
b_3
b
N_3-R_3
N_3
$\{x_M^{(m-1,k)} : k=1,\ldots,R_{m-2}\}$
Stage m-1 (conditional samples)
0
b_{m-1}
N_{m-1}-R_{m-1}
N_{m-1}
$\{x_M^{(m,k)} : k=1,\ldots,R_{m-1}\}$
Stage m (conditional samples)
0
$b_m=b$
N_m-R_m
N_m
samples outside failure region F_i $=F_{i-1}-F_i \ \forall i=2,\ldots,m$
samples in F_i = mother samples $\{x_M^{(i+1,k)} : k=1,\ldots,R_i\}$

Figure 2.4.: Subset simulation scheme illustrating the recursive procedure.

What remains is the question how to generate $N_i - R_{i-1}$ out of R_{i-1} mother samples. In practice, if the division N_i/R_{i-1} always yields an integer value n_i (this depends on the chosen N_i and p_0), each mother sample can be selected to simulate $n_i - 1$ independent offspring samples. If this condition does not hold, one can easily create a strategy how to distribute the remaining 'rest' of needed offsprings over the list of mother samples. However, the easiest way is to simulate $N_i - R_{i-1}$ one by one and to select a new mother sample uniformly from the set of mother samples each time. This strategy is applicable for any proportion of N_i and R_{i-1} and will also have the least influence on the bias of the estimator. Finally, the efficient generation of a conditional sample distributed as $p\left(x|F_{i-1}\right)$ using a given mother sample x_M will be explained in the following subsection.

2.4.2. Markov Chain Monte Carlo Simulation

In general, for any arbitrary PDF $p(x)$, an ergodic Markov chain can be constructed and used to generate samples approximately distributed as $p(x)$. Consider again the task to generate samples $\left\{x^{(i,k)}\right\}_{k=1}^{N_i}$, $i = 2, \ldots, m$ which are distributed according to the conditional PDF

$$p(x|F_{i-1}) = \frac{p(x)\,\mathbb{I}(x \in F_{i-1})}{p(F_{i-1})} \tag{2.66}$$

These are samples distributed as $p(x)$ which lie in the failure region F_{i-1}. It is also possible to generate such samples with direct MCS and take only the samples lying in F_{i-1}, but this would not improve anything since on average it needs $1/P(F_{i-1})$ samples before one such sample occurs. Fortunately, the Metropolis-Hastings algorithm provides the properties for an efficient simulation.

2.4.2.1. Metropolis-Hastings Algorithm

First introduced by Metropolis et al. in 1953 [19] and generalized by Hastings in 1970 [20], the so-called Metropolis-Hastings (MH) algorithm is a rejection sampling algorithm used to generate samples which are distributed according to an arbitrary probability distribution $p(x)$, which may be hard to sample from directly. Therefore, a (conditional) proposal distribution $q(x'|x^{i-1})$ (which may be easy to sample from) is used to generate a sample x' which depends on the mother sample x^{i-1} from the previous algorithm step $i-1$. To decide whether a new sample x' is accepted or rejected the following ratio is calculated:

$$\rho = \begin{cases} \min\left[1, \frac{p(x')q\left(x^{i-1}|x'\right)}{p(x^{i-1})q(x'|x^{i-1})}\right] & \text{if } \; p(x^{i-1})\,q(x'|x^{i-1}) > 0 \\ 1 & \text{otherwise} \end{cases} \tag{2.67}$$

The new sample x' is then accepted ($x^i := x'$) with probability ρ and rejected ($x^i := x^{i-1}$) with probability $1-\rho$. One easily recognizes, that the acceptance ratio (2.67) is independent of $q(\cdot)$ for symmetric proposal distributions.

A drawback of the MH algorithm is that it is not applicable to high dimensional problems, because the expected step size of the MCMC simulation grows with the dimensionality of the PDFs [21]. For the MH algorithm, large steps will almost always end up in regions with low probability and therefore the samples will be rejected. This zero acceptance phenomenon leads generally to very low acceptance ratios ρ, which means that the MCMC samples will highly depend on each other and the sample domain is no longer explored appropriately during the simulation.

2.4.2.2. Sample Generation using a Modified Metropolis-Hastings Algorithm

To tackle the zero acceptance problem of the original MH algorithm, [4] proposed a slightly modified version of the MH algorithm. The main difference to its original version is that the generation of sample proposals and their rejection is done independently for each dimension of the sample vector rather than for the whole n-dimensional sample vector as in the original version. With this modification the algorithm also works for high dimensions. Due to the fact that MH algorithm generates conditional samples by using samples from the previous iteration (mother samples), it can be used to generate samples close to the mother sample and also distributed as $p(x)$. This property makes the MCMC-based sample generation more efficient compared to direct MCS. To see this, one has to consider that only samples which lie in the (current) failure region are used as mother samples. Therefore, a sample, which is closely located to a mother sample (being in failure region), will have much higher probability also to lie within the failure region than a sample generated by direct MCS. What remains, is the fact that the Metropolis algorithm also generates samples distributed as $p(x)$. To obtain conditional samples according to $p(x|F_{i-1})$, the samples from $p(x)$ are simply rejected, if they do not lie within the failure region F_{i-1}.

Remember, that the PDF of the samples is actually given by $p(u)$ and $p(x)$ can be calculated with the response function $r(\cdot)$. To adapt the MH algorithm for high dimensions let $u_j, j \in \{1, \ldots, p\}$ denote the j-th dimension of vector u. The final algorithm to efficiently generate new samples which lie in the current failure region now looks as follows:

Algorithm 3 Generation of a conditional sample $x \sim p(x|F_{i-1})$

```
Input is a mother sample $u^{i-1} = \left\{u_1^{i-1}, \ldots, u_p^{i-1}\right\}$ with response $x^{i-1}$
Output is a new sample $x^i \sim p(x|F_{i-1})$ which depends on $x^{i-1}$ (due to MCMCS)

1. Generate a sample $x^* \sim p(x)$:
   for all dimensions $j \in \{1, \ldots, p\}$
      a) Generate a candidate $u'_j$ according to $q_j\left(u'_j|u_j^{i-1}\right)$
      b) Calculate the acceptance ratio:
         $$\rho_j = \begin{cases} \min\left[1, \frac{p_j\left(u'_j\right)q_j\left(u_j^{i-1}|u'_j\right)}{p_j\left(u_j^{i-1}\right)q_j\left(u'_j|u_j^{i-1}\right)}\right] & \text{if } p_j\left(u_j^{i-1}\right)q_j\left(u'_j|u_j^{i-1}\right) > 0 \\ 1 & \text{otherwise} \end{cases}$$
      c) Draw a uniform sample $a$ from $U(0,1)$
      d) Define $u_j^i$ using the acceptance ratio:
         $$u_j^* = \begin{cases} u'_j & \text{if } a \le \rho_j \\ u_j^{i-1} & \text{otherwise} \end{cases}$$
   end for
   $x^* = r(u^*)$

2. Ensure that $x^i \sim p(x|F_{i-1})$:
   $$x^i = \begin{cases} x^* & \text{if } x^* \in F_{i-1} \\ x^{i-1} & \text{otherwise} \end{cases}$$
```

The purpose of the first rejection in step (1.d) of the algorithm is to keep the samples generated by the Markov chain distributed as $p(u)$, which may not be the case for a sample proposed by $q(u'|u^{i-1})$. Since the algorithm assumes the last sample u^{i-1} to be distributed as $p(u)$ this property can be ensured for the next sample by simply repeating the previous sample. Therefore, MCMC simulation needs a 'warm up' phase, if the initial sample of the Markov chain is not distributed as $p(u)$, because with every Markov chain iteration the current sample will be more independent of the initial sample and the MH algorithm will actually generate samples according to $p(u)$. Note that this is not necessary within the subset simulation scheme, because the samples in the first stage, which are the mothers and initial samples for the following stages, are generated by direct MCS according to $p(u)$. Finally, note that the generated Markov chains are actually short, since they will be as long as there will be subset recursion stages m.

Within the subset simulation framework, the MH algorithm iterates over all subsets $i = \{2, \ldots, m\}$. Assuming that R_{i-1} samples in the previous iteration have passed the intermediate threshold b_{i-1}. In iteration i these R_{i-1} samples are used as mother samples to generate $N_i - R_{i-1}$ offspring samples. Together with the mother samples, N_i are used to estimate conditional probability $P(F_i|F_{i-1})$, because all samples (mothers and offsprings) lie within F_{i-1}. Thus, in each iteration i, the sample generation algorithm is called $N_i - R_{i-1}$ times to provide N_i samples $\left\{x^{(i,k)}\right\}_{k=1}^{N_i}$, $i = 2, \ldots, m$ to estimate the failure probability for F_i.

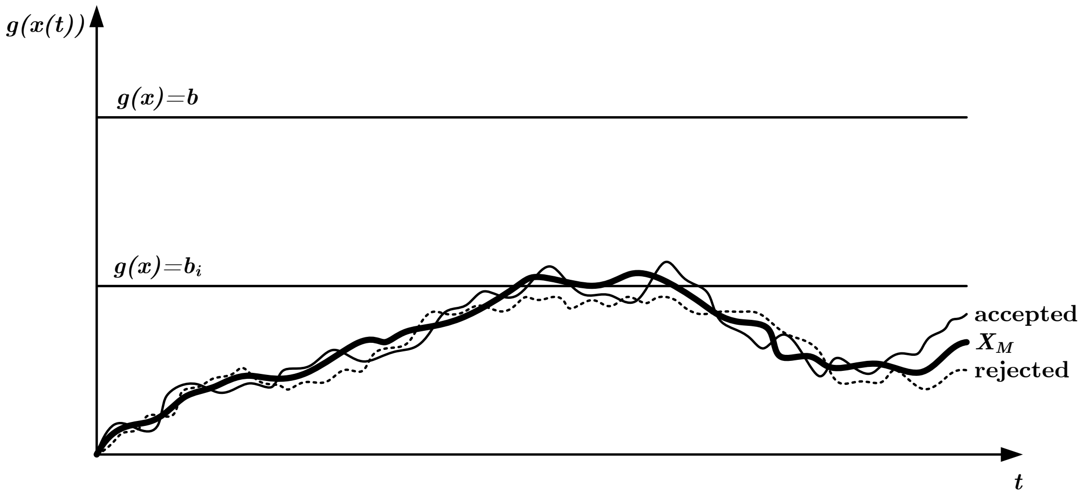


Figure 2.5.: Accepted and rejected candidate offspring trajectories (according to [1]).

Figure 2.5 illustrates the rejection of candidate offspring samples (step 2 in algorithm 3) for the case of studying a dynamic system over time, i.e. the samples are system trajectories. After the generation of a candidate trajectory x_C using x_M, the candidate may be rejected to ensure its distribution according to $p(x|F_{i-1})$. In case of a rejection, the mother trajectory is simply repeated. In sum, the better efficiency to generate samples from $P(F_i|F_{i-1})$ using MCMC is only supported by the assumption that the failure region is an union of somehow compact structures, that is, given a point $x \in F$ and a small distance value ϵ, the local neighborhood of x, defined by $\mathcal{N}_x = \{a \mid a \in \mathcal{X},\ \|x - a\| < \epsilon\}$, has a high probability lie entirely in the failure region as well. Fortunately, this assumption seems to hold for many engineering systems of interest for reasonable values of ϵ. The parameter ϵ is similar to the support region of the proposal function, which may also be controlled by a parameter. However, if the assumption does not hold, the algorithm still works, but its computational efficiency will drop to the one of direct MCS (omitting the fact that the sample dependencies of MCMC also reduce efficiency wrt. the failure region exploration).

2.4.2.3. Reusage of Samples during Sampling Stages

In literature the first publications about the subset simulation technique [4, 13, 22] do not use the samples which passed the intermediate threshold level b_i in stage i for the estimation of the partial failure probability $P_{i+1} = P\left(F_{i+1}|F_i\right)$ of the next stage $i+1$. Thus, if a number of N is used for every stage in a simulation over m stages, the total number of samples will be $N_T = mN$. However, since the samples from stage i, which passed the intermediate threshold level b_i are also distributed as $p\left(x|F_i\right)$ they can also be used to estimate $P\left(F_{i+1}|F_i\right)$. Since this approach reduces the total number of samples without reducing the c.o.v. of the individual estimators $\hat{P}_i$, it is natural to take the advantage of this approach. In the case that all intermediate limit values b_i are determined by the a priori chosen partial failure probability p_0, the total number of samples is then given as

$$N_T = N_m + (1-p_0)\sum_{i=1}^{m-1} N_i \tag{2.68}$$

If the number of samples for each stage is equal, i.e. $N = N_1 = \ldots = N_m$, the last equation simplifies to

$$N_T = (m - p_0 m + p_0)N \tag{2.69}$$

This will also hold for the variants of the subset simulation procedures explained in the following subsections.

2.4.3. Statistical Properties of the Estimator

The estimator $\hat{P}_F$ is unbiased [13, p.268] given the assumption that the proposal PDF has a widely similar shape and support region as the input PDF. Its variation is derived over the variation of each estimator $\hat{P}_i$ for simulation stage i. For the first level, the variation and the c.o.v. of the estimator $\hat{P}_1$ is equivalent to that of the MCS estimator:

$$Var\left[\hat{P}_1\right] = \frac{(1-P_1)P_1}{N_1} \qquad \text{and c.o.v.: } \delta_1 = \sqrt{\frac{1-P_1}{P_1 \cdot N_1}} \tag{2.70}$$

The variation for the estimators P_i $(i = 2, \ldots, m)$ which make use of MCMC simulation is influenced by the correlation of the Markov chain samples and is given as

$$Var\left[\hat{P}_i\right] = \frac{(1-P_i)P_i}{N_i}(1+\gamma_i) \qquad \text{and c.o.v.: } \delta_i = \sqrt{\frac{1-P_i}{P_i \cdot N_i}(1+\gamma_i)} \tag{2.71}$$

The variable γ_i describes the correlation of the MCMC samples and the term $N_i/(1+\gamma_i)$ can be interpreted as the effective number of independent samples. Thus, $\gamma_i = 0$ for the case of independent samples and for Markov chain Monte Carlo the correlation is generally $\gamma_i > 0$.

The overall c.o.v. of the $\hat{P}_F$ estimator can be well approximated with the formula for the uncorrelated case:

$$\delta_{P_F} = E\left[\frac{\hat{P}_F - P_F}{P_F}\right] \approx \sqrt{\sum_{i=1}^{m} \delta_i^2} \tag{2.72}$$

Efficiency of the method

The following equation gives an approximation to the total number of samples needed to estimate P_F with particular accuracy, given as c.o.v. δ:

$$N_T \approx |\log P_F|^r \cdot \frac{(1+\gamma)(1-p_0)}{p_0\,|\log p_0|^r\,\delta^2} \tag{2.73}$$

where the exponent $r \leq 3$ specifies the correlation of the probabilities P_i. Compared to direct MCS, where the number of needed samples is proportional to $N_T \sim 1/P_F$, subset simulation provides a significant efficiency improvement with $N_T \sim |\log P_F|^r$, for fixed p_0, δ.

Advantages

The first step of subset simulation is direct MCS which effectively explores the sampling domain, for higher accuracy MCMC is used to generate more samples only within the failure region. The effectiveness to explore the intermediate failure regions can still be controlled by changing the shape of the proposal PDF. Therefore, subset simulation is widely independent of the shape of the failure region. Due to the approach to determine the intermediate failure regions (i.e. their bounds) adaptively according to fixed conditional probabilities the choice of the intermediate threshold levels can be effectively controlled by just one parameter. In sum, the original problem of sampling very small failure probabilities is replaced by a series of simulations in which the failure events occur more frequently and can therefore be estimated efficiently with a small number of samples. Another advantage of the subset simulation procedure is that one simulation run over m stages, provides estimates for all threshold levels having probabilities between 0 and 1, where the resolution in the probability space is p_0^{i-1}/N_i for every stage $i = 1, \ldots, m$. Thus, one failure estimation delivers an estimation of all possible threshold levels as the probability for a particular threshold can be obtained by means of interpolation methods.

Disadvantages

The choice of the proposal PDF is difficult and also depends on the sampling problem. If the support region of the proposal PDF is too broad, the samples will have lower probability of lying in the failure region and many samples may be rejected. Since the mother sample is repeated in that case, this will introduce high dependencies between samples and also influence the exploration of the failure region. On the other hand, if the support region of the proposal PDF is too small, the samples will highly depend on each other and the failure region is poorly explored as well. Therefore the choice of the proposal PDF is a difficult trade-off to obtain samples having a high probability of lying in the failure region and also effectively explore the failure region at the same time. The choice of the proposal PDF does not only effect the efficiency of the MCMC algorithm it may also influence the ergodicy of the MCMC procedure. If the failure region is not explored properly by the MCMC procedure, the estimator based on the conditional properties may be biased. For example, this may happen if a failure region is separated from another failure region and has not been explored in the first MCS stage. Further, if the distance of the separated failure region is large than proposal function support region, the failure region can never be explored by the MCMC procedure. Due to the adaptive choice of the intermediate threshold levels b_i one does not know in advance how many iterations will be needed to reach the limit value of interest b. Hence, the total number of samples and thus the total computational effort is unknown in advance.

2.5. Subset Simulation with Splitting

The subset simulation with splitting approach (abbreviated as SS/S) realizes the same idea as the subset simulation with MCMC (abbreviated as SS/MCMC) with the difference that conditional samples can be generated differently for dynamic reliability problems, in particular, the first passage point problem. The splitting approach makes the use of MCMC unnecessary and therefore eliminates the problem to choose a proper proposal density, which highly influences the efficiency and accuracy of the SS/MCMC method.

Therefore, the original subset simulation procedure explained in the last section (2.4) remains mainly the same except for the step, where the conditional samples are generated based on the given mother samples from the previous subset simulation stage (subsection 2.4.2.2).

Let again the failure regions F_i be described by threshold values b_i for an engineering demand parameter $g(x)$, i.e. $x \in F_i \Leftrightarrow g(x) \geq b_i$, where all limit values form an ordered sequence of increasing limit values $b_1 < b_2 < \ldots < b_m = b$.

In the first stage, direct MCS is used again to estimate the independent probability P_1 of the first subset. The estimator is similar to (2.61) with the difference that the failure event is now defined over the system trajectory, checking whether a first excursion point exists:

$$P_1 \approx \hat{P}_1 = \frac{1}{N_1} \sum_{k=1}^{N_1} \mathbb{I}\left(\max_t \left[g\left(x^{(1,k)}(t)\right)\right] \geq b_1\right) = \frac{R_1}{N_1} \tag{2.74}$$

Similarly, the samples $\left\{x^{(1,k)}(t)\right\}_{k=1}^{N_1}$ of the first stage are simulated by direct Monte Carlo simulation. For the following stages the conditional samples $\left\{x^{(i,k)}(t)\right\}_{k=1}^{N_i}$ are generated according to the splitting approach which will be explained in the following. However, the estimator for the conditional failure probabilities is similar to (2.74):

$$P_i \approx \hat{P}_i = \frac{1}{N_i} \sum_{k=1}^{N_i} \mathbb{I}\left(\max_t \left[g\left(x^{(i,k)}(t)\right)\right] \geq b_i\right) = \frac{R_i}{N_i} \tag{2.75}$$

2.5.1. Generation of conditional samples with splitting

Due to the specialization to a first passage problem, the samples are known to have a shape of a trajectory and the generation of a conditional sample given a mother trajectory can also be done efficiently in the following way. A mother excitation u_M, where its corresponding trajectory x_M is known to lie in failure region F_i, can be split into the partial excitation and partial trajectory before the first passage point S, denoted by u_M^- and x_M^-, respectively, and also denote those after the first passage point by u_M^+ and x_M^+. Let, for simplicity, S contain the time coordinate and the corresponding state value of the first passage point.

To simulate an offspring trajectory x_O, which is also distributed as $p\left(x|F_i\right)$, the partial trajectory x_M^- is copied from the mother, i.e. $x_O^- = x_M^-$ (by $u_O^- = u_M^-$). This will ensure that $x_O \in F_i$, regardless of the new partial trajectory $\widetilde{x}^+$ after S. Hence, if $\widetilde{u}^+$ is generated according to $p\left(u^+|u^-\right) = p\left(u^+, u^-\right)/p\left(u^-\right)$ it follows that the new offspring trajectory $x_O = \left\{x_M^-, \widetilde{x}^+\right\}$ is also distributed as $p\left(x|F_i\right)$. Since the failure event occurs, given that the partial excitation x_M^- reaches the first passage point, it follows that $P\left(F_i|\widetilde{u}^+, u_M^-\right) = 1$ and $P\left(F_i|u_M^-\right) = 1$. Using this knowledge in Bayes's theorem, it turns out that the distribution of the future excitation $\widetilde{u}^+$, given u_M^-, is independent of the failure event:

$$p\left(\widetilde{u}^+|F_i, u_M^-\right) = \frac{P\left(F_i|\widetilde{u}^+, u_M^-\right)}{P\left(F_i|u_M^-\right)} p\left(\widetilde{u}^+|u_M^-\right) = p\left(\widetilde{u}^+|u_M^-\right) \tag{2.76}$$

Therefore, $\widetilde{u}^+$ can be simply generated by direct Monte Carlo simulation.

The generation of an offspring trajectory according to the splitting approach is illustrated in the following Figure 2.6. The first part of the offspring trajectory x_C is equal to its mother trajectory x_M until the first passage point is reached. The second part after the first passage point is generated by direct MCS and is thus completely independent of its mother trajectory.

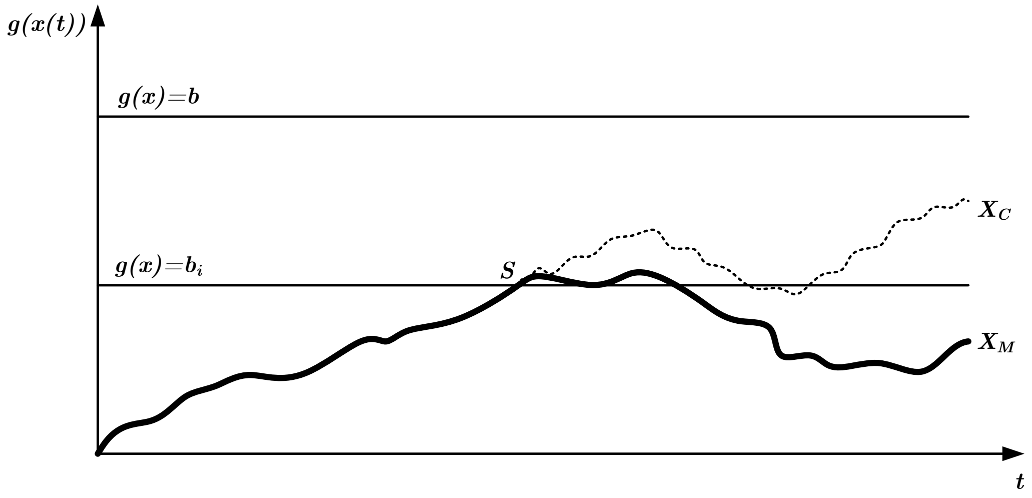


Figure 2.6.: Generation of an offspring trajectory (according to [1]).

The idea of the splitting algorithm can be summarized in a way that the creation of subsets to order a set of samples is further broken down to the sub-level of the samples. The property which has been used in general subset simulation is that sample responses which exceeded the threshold b_i have also exceeded threshold $b_{i-1} < b_i$. For the first excursion problem, one can find such a property on the sample level: A trajectory part which has exceeded a threshold b_i has also exceed the threshold level $b_{i-1} < b_i$ at the same or an earlier time point.

2.5.2. Final Algorithm

Using the sampling generation method explained in the last subsection and the same principles as in SS/MCMC, the subset simulation with splitting algorithm can be described as follows:

Algorithm 4 Subset simulation with splitting.

```
- Given is the response function $r(\cdot)$ of a causal system and a probability
  distribution $p(u)$ for the system input.
- Output is the probability estimation $\hat{P}_F$ for a system failure.

A. Stage 1: Generate root samples using direct MCS
    $i := 1$
    Simulate $N_1$ i.i.d. trajectories $\left\{x^{(1,k)}(t) \;:\; t = t_l, l = 1, \ldots, n_t,\;\; k = 1, \ldots, N_1\right\}$
    by drawing samples from $p(u)$ and calculate their response with function $r(\cdot)$
    Sort samples $\left\{x^{(1,k)}(t)\right\}_{k=1}^{N_1}$ in ascending order wrt. $\max_t \left[g\left(x^{(1,k)}(t)\right)\right]$
    Calculate $b_1 := \max_t \left[g\left(x^{(1,j)}(t)\right)\right]$ with $j = \lfloor(1 - p_0) \cdot N_1\rfloor$, $R_1 := p_0 N_1$

B. Stage $i$: Estimate $P_i$ (i=1,...,m-1) and possibly calculate new subsets recursively:
    while $(b_i < b)$                                   ▷ a new subset can be opened up
        $\hat{P}_i = p_0$                                ▷ probability is predefined by $b_i$
        $R_i$ trajectories exceeded threshold $b_i$: assign them as mother
        trajectories $\left\{x_M^{(i+1,k)}(t)\right\}_{k=1}^{R_i}$ and record their first passage points $\left\{S^{(i+1,k)}\right\}_{k=1}^{R_i}$
        $i := i + 1$                                     ▷ open up a new subset
        forall $N_i - R_{i-1}$ offsprings that need to be created:
            Uniformly select a mother trajectory $x_M$ having index $k$
            Create offspring as $x_O = \left\{x_M^-, \widetilde{x}^+\right\}$ by applying $r(\cdot)$ to the partial mother
            excitation $u_M^-$ and the new simulated partial excitation $\widetilde{u}^+ \sim p(u^+|u^-)$,
            which starts after the first passage point $S^{(i,k)}$
        end for
        Sort samples $\left\{x^{(i,k)}(t)\right\}_{k=1}^{N_i}$ in ascending order wrt. $\max_t \left[g\left(x^{(i,k)}(t)\right)\right]$
        $b_i := \max_t \left[g\left(x^{(i,j)}(t)\right)\right]$ with $j = \lfloor(1 - p_0) \cdot N_i\rfloor$, $R_i := p_0 N_i$
    end while
    The last subset has been reached and $b_i$ is given as $b_i = b_m = b$ $(m := i)$
    $\hat{P}_m = \frac{R_m}{N_m}$, where $R_m$ is the number of samples exceeding threshold $g(x) \geq b$.

C. Finally, calculate the failure probability estimation: $\hat{P}_F = \prod_{i=1}^{m} \hat{P}_i$
```

Dealing with a discrete-time state-space system

For the conditional sample generation it has been derived that the trajectories can be freely simulated with direct MCS starting from the first passage point. When using a discrete time model the first passage point S is generally not hit by a sample point. Therefore, one can argue that the simulation of the offspring trajectories can either be started from the sample point S^- right before the FPP, or from the sample point S^+ right after the FPP. The first option again introduces a rejection step, because the simulated trajectory might not exceed the given threshold. However, for reasonable time sampling S^- should be very close to S and rejection rate be nearly zero. Nevertheless, starting the simulation from S^+ is easier to implement and does not affect the computational efficiency.

2.5.3. Statistical Properties of the Estimator

The estimator $\hat{P}_F$ is an unbiased estimator of P_F [1, p.1566] based on the assumption that the space after the first passage points is reasonable long to ensure an appropriate exploration of the failure region. This requires first a reasonable length of the system trajectories and second that the distribution of the FPP times over the system trajectory is not concentrated at the end of the trajectory. The variation of the estimator $\hat{P}_i$ can be approximated under the assumption that the first passage points $\left\{S^{(i+1,k)}\right\}_{k=1}^{R_i}$ do not strongly depend on each other, which should hold for large number of samples $N_1, \ldots, N_{i-1}$. With this assumption the variance can be approximated as

$$Var\left[\hat{P}_i^2\right] \approx \frac{P_i(1-P_i)}{N_i}\left(1+\gamma_i\right) \tag{2.77}$$

where γ_i describes the correlation between samples and reduces the number of samples N_i to the effective number of independent samples used for the estimation. The c.o.v. for $\hat{P}_i$, denoted by δ_i, is given by

$$\delta_i = \sqrt{\frac{1-P_i}{P_i \cdot N_i}\left(1+\gamma_i\right)} \tag{2.78}$$

This leads to the overall c.o.v. δ for subset simulation with splitting:

$$\delta^2 \approx \sum_{i=1}^{m} \delta_i = \frac{P_1(1-P_1)}{N_1} + \sum_{i=2}^{m} \frac{P_i(1-P_i)}{N_i}\left(1+\gamma_i\right) \tag{2.79}$$

Equation (2.79) gives only an estimation of the c.o.v.. Using the same assumption Ching et. al [1] therefore also derived estimators for the lower and upper bounds of the c.o.v., which are given as

$$\frac{(1-P_1)}{P_1 N_1} + \sum_{i=2}^{m}\left[\frac{1-P_i}{P_i N_i}\right] \leq \delta^2 \leq \frac{(1-P_1)}{P_1 N_1} + \sum_{i=2}^{m}\left[\frac{1-P_i}{P_i N_i}\left(1+E\left[\frac{N_i}{R_{i-1}}\right]\right)\right] \tag{2.80}$$

The lower bound is defined by the c.o.v. values that would be obtained if each of the conditional probabilities were estimated using direct MCS. Under the assumption of independent samples they can be summed up to the overall c.o.v. that would obtained, if all conditional failure probabilities were estimated using direct MCS. However, due to repetition of the time series before the FPP time, the samples are generally dependent. Hence, this lower bound will not be reached in practice. Similar to SS/MCMC the intermediate threshold values b_i can be chosen a priori or adaptively during the simulation. For the latter case, i.e. the conditional probabilities are defined a priori with value p_0.

Advantages

In contrast to SS/MCMC the generated offsprings are in the failure region, because the partial excitation until the first passage point is simply copied and therefore no further rejection is needed. In case of a rejection of candidate trajectory in the SS/MCMC framework, the mother trajectory is simply copied, which introduces stronger dependencies of samples between the simulation stages. In SS/S this step is omitted and a new offspring trajectory is always generated (and accepted), which leads to a better exploration of the failure region after the first passage point. Consequently, due to the free simulation of each sample after the first passage point, every offspring sample is distinct from its mother sample.

While in the MCMC framework the offsprings are generally correlated with their neighbors, the SS/S framework generates independent partial trajectories for each subset. The trajectory space after the first passage point S is effectively explored, since all partial offspring trajectories $\widetilde{x}^{+}$ are generated independently. The trajectory space before S is explored by MCS in previous subsets.

The SS/S framework does not require MCMC simulation and therefore avoids the problem of choosing an appropriate proposal PDF. Except of the choice for the partial failure probabilities p_0 the algorithm is parameterless. Furthermore, the computational effort for SS/S is less than that for MCMC since only parts of the trajectories need to be generated and the first part is simply copied, which is faster than simulation. Also, depending on $r(\cdot)$, it may be possible to avoid expensive response calculation for this first part.

Disadvantages

Depending on the system to be studied, i.e., the response function, the first passage point may occur very late during the time of study, especially for very small failure probabilities. This means, that only small parts of the system trajectories are used to explore the failure region during the sampling process. Hence, the main drawback of the SS/S algorithm is that it copies the trajectory parts before the first passage points and relies the exploration of the trajectory space before the first passage points on common Monte Carlo simulation. Therefore, SS/MCMC may outperform SS/S and vice versa with respect to the c.o.v. of the failure probability estimates. Moreover, SS/S is less general than SS/MCMC as it requires a causal system and the condition that an excitation conditioned on the past excitation can be easily sampled.

2.6. Hybrid Subset Simulation

Subset simulation with hybrid Markov chain and splitting (abbreviated SS/H) combines subset simulation with Markov chain Monte Carlo (SS/MCMC) and subset simulation with splitting (SS/S). The idea for a combination of the two methods has arisen from the theoretical and empirical studies of the statistical properties of the estimators. Mainly, the c.o.v. of SS/MCMC estimator is sometimes smaller than the SS/S estimator and vice versa. Therefore, the following hybrid subset simulation method tries to combine the advantages of both methods.

This is because the two methods produce different types of dependencies between offspring-mother and offspring-offspring relations. To see this, consider again the partition of a trajectory x into a partial trajectory before x^- and after x^+ the first passage point S. In SS/H the generation of the offspring $x_O = \left\{x_O^-, x_O^+\right\}$ after the first passage point is completely independent of the mother sample since direct MCS is used to generate x_O^+. This explores the failure region much better than MCMC simulation and therefore SS/H outperforms SS/MCMC in the second part, because the samples which are generated using MCMC simulation introduce dependencies between the offspring trajectories and their mother trajectories. Finally, these dependencies, i.e. correlations between samples lead to higher (co)variance values and thus reduce the efficiency of the sampling method. Considering the generation of the first part of the offspring trajectory x_O^- the SS/H method simply copies this part from its mother trajectory $x_O^- = x_M^-$, which means that the correlation for this part of the sample could not be stronger. On the other hand, SS/MCMC generates this part as well with MCMC simulation, which also introduces dependencies to its mother sample, but they are by far fewer than a direct copy. Hence, the advantages and disadvantages of the two methods with respect to their correlation for the partial trajectories are completely opposite.

Keeping this properties in mind, the idea which comes up straightly, is to generate the first part of the trajectory x_O^- using MCMC simulation to reduce the correlation between samples (compared to simply copying it). The second part after the first passage point, x_O^-, is then generated using the method used in SS/H, that is, direct MCS, which does not introduce any correlation at all - in contrast to MCMC simulation. Besides the mixture of the aforementioned algorithms, the main change in the implementation is that the trajectory generation according to MCMC needs to be monitored to find out whether a first passage point has been reached to abort the generation. If a first passage point (FPP) has been found before the generation normally ends, the generation is continued with direct MCS. The hybrid method for subset simulation is illustrated in figure 2.7 (also compare with figures 2.5 and 2.6) and explained in detail in the following algorithm.

2.6.1. Algorithm

Algorithm 5 Hybrid subset simulation

```
- Given is the response function $r(\cdot)$ of a causal system and a probability
  distribution $p(u)$ for the system input.
- Output is the probability estimation $\hat{P}_F$ for a system failure.

A. Stage 1: Generate root samples using direct MCS
   $i := 1$
   Simulate $N_1$ i.i.d. trajectories $\left\{x^{(1,k)}(t) \ : \ t = t_l, l = 1, \ldots, n_t, \ \ k = 1, \ldots, N_1\right\}$
   by drawing samples from $p(u)$ and calculate their response with function $r(\cdot)$
   Sort samples $\left\{x^{(1,k)}(t)\right\}_{k=1}^{N_1}$ in ascending order wrt. $\max_t \left[g\left(x^{(1,k)}(t)\right)\right]$
   Calculate $b_1 := \max_t \left[g\left(x^{(1,j)}(t)\right)\right]$ with $j = \lfloor(1-p_0) \cdot N_1\rfloor$, $R_1 := p_0 N_1$

B. Stage $i$: Estimate $P_i$ (i=1,...,m-1) and possibly calculate new subsets recursively:
   while $(b_i < b)$                                    ▷ a new subset can be opened up
      $\hat{P}_i = p_0$                                  ▷ probability is predefined by $b_i$
      $R_i$ trajectories exceeded threshold $b_i$: assign them as mothers $\left\{x_M^{(i+1,k)}(t)\right\}_{k=1}^{R_i}$
      $i := i + 1$                                       ▷ open up a new subset
      forall $N_i - R_{i-1}$ offsprings that need to be created:
         Uniformly select a mother trajectory $x_M$ having index $k$
         Generate offspring pointwise with MCMC until a FPP has been reached
         if FPP has been reached:
            the first part $\widetilde{x}^-$ of trajectory $x_O = \{\widetilde{x}^-, \widetilde{x}^+\}$ is completed
         else repeat the first part of the mother trajectory: $\widetilde{x}^- = x_M^-$
         Calculate $\widetilde{x}^+$ after generating $\widetilde{u}^+$ according to $p(u^+|u^-)$
      end for
      Sort samples $\left\{x^{(i,k)}(t)\right\}_{k=1}^{N_i}$ in ascending order wrt. $\max_t \left[g\left(x^{(i,k)}(t)\right)\right]$
      $b_i := \max_t \left[g\left(x^{(i,j)}(t)\right)\right]$ with $j = \lfloor(1-p_0) \cdot N_i\rfloor$, $R_i := p_0 N_i$
   end while
   The last subset has been reached and $b_i$ is given as $b_i = b_m = b$ $(m := i)$
   $\hat{P}_m = \frac{R_m}{N_m}$, where $R_m$ is the number of samples exceeding threshold $g(x) \geq b$.

C. Finally, calculate the failure probability estimation: $\hat{P}_F = \prod_{i=1}^{m} \hat{P}_i$
```

2.6.2. Statistical Properties of the Estimator

Subset simulation with hybrid Markov chain and splitting combines the advantages of both SS/MCMC and SS/S, that is, it reduces the estimation c.o.v. due to the reduction of sample dependencies which exist in both methods. Therefore, the SS/H estimator can be expected to have the same or a lower c.o.v. than both of the sampling methods it is derived from.

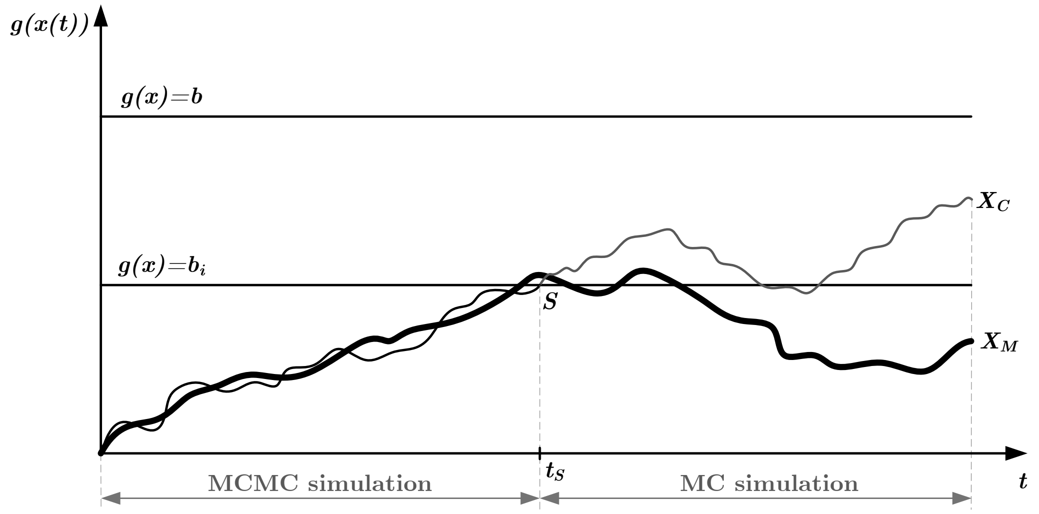


Figure 2.7.: Generation of an offspring trajectory using hybrid MCMC and MCS.

Ching et al. [14] derived approximations for c.o.v. of the hybrid method estimator which are useful for theoretic conclusions but impractical to decide about how many samples are needed to achieve a predefined accuracy of the estimation. Therefore, they have shown that the lower and upper bounds for the c.o.v. which were derived for SS/S are also valid for the hybrid method. Again under the assumption that the number of samples per stage N_i is large, the c.o.v. δ should be within the following bounds:

$$\frac{(1-P_1)}{P_1 N_1} + \sum_{i=2}^{m} \left[\frac{1-P_i}{P_i N_i}\right] \leq \delta^2 \leq \frac{(1-P_1)}{P_1 N_1} + \sum_{i=2}^{m} \left[\frac{1-P_i}{P_i N_i}\left(1 + E\left[\frac{N_i}{R_{i-1}}\right]\right)\right] \tag{2.81}$$

Advantages

SS/H combines the advantage that SS/MCMC has lower c.o.v. before the FPP and that SS/S has lower c.o.v. after the FPP. As a result, SS/H always has a lower or equal c.o.v. compared to both methods, SS/MCMC and SS/S. Nonetheless, the hybrid method again introduces the necessity of a proposal function, but SS/H is less sensible to the choice of proposal function and their parameters which will be shown in the evaluation.

Disadvantages

As already mentioned, due to the usage of MCMC simulation the problem of choosing an appropriate proposal function arises again, even if SS/H is more robust to the choice of the proposal PDF it is not independent of its choice. Moreover, the computational cost of SS/H is higher since it again introduces a rejection step, because it may happen that the first part, which is simulated with MCMC, never reaches the required limit value. This means, that the complete trajectory is simulated with MCMC but finally discarded and as in SS/S the first part of the mother trajectory is repeated.

3. Implementation

This chapter gives an overview of the software package which provides all sampling methods described above. Since computing performance is an important issue to solve large-scaled engineering problems this work is also focused on an efficient software implementation and the use of parallel computing techniques.

3.1. Optimization of the Subset Simulation Procedure

For the dynamic analysis, the amount of data handled by the subset simulation algorithms increases easily with the size of the structure under study. Usually the resource management is a trade-off between memory usage and calculation time, but also the selection of data structures and processing schemes may affect the usage of system resources.

Generally, all subset simulation methods can be implemented in an iterative or an recursive scheme to process through the sampling stages. While this choice has no influence on the time complexity, it has an effect on the memory complexity. In a recursive scheme, the mother samples have to be kept in memory for the next stage and cannot be released before the recursion stops and so the total memory usage will be $O(mN)$, where m is the number of simulation stages and N is the number of samples in each stage. In the iterative version, the mother samples can be released after the generation of the offspring samples. The memory usage decreases to $O(N)$, because only a maximum of $2N$ samples have to be kept in memory. A recursive implementation should hence be avoided.

Using a different data structure, the memory usage can be further reduced, especially for the case that the corresponding responses are also saved. In simple Monte Carlo simulation, the storage of samples is not necessary, since the only information needed is whether a sample caused a system failure. This information can simply be hold within a counter variable. But as stated above, in all subset simulation methods some of the samples in each stage are needed to generate new descendants. Furthermore, it is also beneficial to store the system response for the case that a generated offspring candidate is rejected and its mother sample is repeated. This applies for both methods, SS/MCMC and SS/H. For SS/S, the storage of the response also avoids expensive response calculations, because the first part of a mother sample is repeated for every new offspring sample. Since the response function is deterministic, the first part of the mother sample response can be copied as well. Succinctly, in order to avoid computationally expensive system analysis, the storage of samples and responses is advantageous for all subset simulation methods described in this thesis. In dynamic analysis each sample and response is a long time series and this number multiplies with the number of inputs and the degrees of freedom of the system, respectively. Even for a moderate number of samples and medium size structures the amount of memory to store samples and responses easily reaches several Gigabytes, which is still in the order of significant magnitude on todays computers.

Since the limit values b_i are chosen adaptively, it is not known in advance which of the samples will be needed as mother samples for the next stage. Therefore, the intuitive way to implement subset simulation is to generate R_i samples for stage i, sort them and define the limit value b_i with the a priori chosen conditional probability p_0. This will need memory for R_i samples per stage i.

However, the memory usage can be reduced to p_0R_i, because only the p_0R_i samples with largest system response are needed for the next stage. For example, this can be achieved by sorted insertion in a list with a maximum length of p_0R_i elements during the sample generation process. Samples with smaller system responses are therefore displaced downwards the list and are finally discarded, if they no longer fit into the list. While a discarded sample will not be needed as a mother sample in the next stage, its corresponding maximum response value may still be necessary for the partial failure probability estimation. It is therefore suggested to replace the straightforward implementation by a single list for samples and responses by a combination of two lists with the following properties. The first list saves samples and responses as in the simple implementation, but its size is limited to size p_0R_i and the element with the lowest maximum system response is removed, if the sampling process tries to insert a new sample with higher response value into the full list. The displaced sample and response can be released and only the maximum response value needs to be stored in the second list which has size $(1 - p_0)R_i$.

While sorted insertion has a time complexity of $O(n^2)$ the desired functionality can also be obtained by a priority queue, which keeps the elements in a semi-sorted order, if it makes use of a heap, i.e. a self balancing binary tree. In short, a priority queue is the generalization of the queue and the stack data structures. Each element which is added to the list needs an priority value that is chosen by the user. Then only the element with the highest priority in the queue can be read or removed. For further information see [23] or [24, pp.138-142]. In this case, the final complexity to obtain the maximum responses in order is $O(n \log n)$, which is also the lower bound for comparison based sorting algorithms. A reduction of the memory usage can therefore be achieved without significant loss of performance. Figure 3.1 illustrates the described methodology compared to a simple implementation.

Consequently, the use of memory resources is reduced by the fact, that the elements of the second list are much smaller. Since the value of p_0 is suggested to be around 0.1 the memory saving can be significant, while the change of the data arrangement does not change the time complexity.

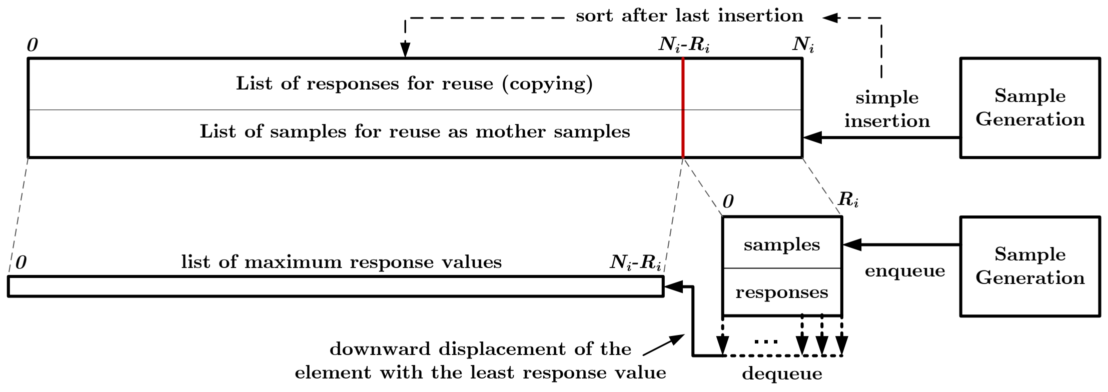

sort after last insertion
0
N_i-R_i
N_i
List of responses for reuse (copying)
List of samples for reuse as mother samples
simple insertion
Sample Generation
0
R_i
samples
responses
enqueue
Sample Generation
0
list of maximum response values
N_i-R_i
downward displacement of the element with the least response value
dequeue

Figure 3.1.: Illustration of the memory savings: simple storage in a list (upper part) vs. a combination of two lists (lower part).

3.2. Parallelization

In order to apply recently developed simulation techniques for reliability based methods to large scale applications, parallel computing is an important part of current research in this area. Due to the use of simulation methods the computational expensive structural analysis is usually part of the simulation of each sample. Generally, the generation of independent samples is a task which can be parallelized to reduce computational time. After explaining some general aspects, this section discusses the parallelization of the simulation methods described in chapter 2.

3.2.1. Preliminary

In general, parallelization can be achieved in different ways. From the hardware point of view parallelization may be achieved by instruction level parallelism, parallel computers, multicore processors, distributed computing, cluster computing, massive parallel processors or grid computing. However, it depends on the task to decide which parallelization method is reasonable and worth the effort. Loosely speaking, one could say the effort increases enormously with the scale of physical distribution, since data synchronization and workload balancing gets more and more complicated.

As generally known, pure Monte Carlo simulation is an so-called embarrassingly parallel problem, which is the simplest form of a parallelization problem, because there are no dependencies between samples and no data needs to be exchanged or synchronized during the sampling process. In contrast, the subset simulation methods need multiple data synchronization steps in between recursion levels. Because of available hardware resources and for simplicity, the focus is kept on symmetric multiprocessing (SMP) and multi-core computing, which is easy to implement and synchronization as well as communication is less complicated. Moreover, the needed hardware is available on todays standard PC's and it is thus widely applicable. This form of parallelism can be achieved by so-called multi-threading which is explained in the next subsection. A general property for task parallelization is given by the following rule.

Amdahl's law of parallelization

The speedup S is the most important measure in parallel computing. It describes how much faster the program is due to the parallelization and is calculated as the proportion of the program runtime T_P in parallel mode and the program runtime T_S in sequential mode.

$$S = \frac{T_S}{T_P} \tag{3.1}$$

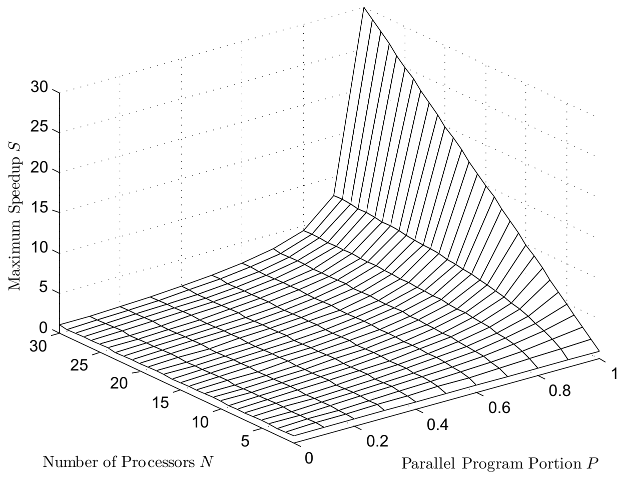


Figure 3.2.: Illustration of Amdahl's law.

Amdahl's law [25] provides an upper bound for the maximum speedup which can be achieved by parallelization. Let P be the proportion of the program which can be parallelized, which means that the proportion $(1 - P)$ remains sequential, then the maximum speedup S_{max} by using N processing units is given as

$$S_{max} = \frac{1}{(1 - P) + \frac{P}{N}} \tag{3.2}$$

This means, for instance, if 90% of the program is parallizable, the parallel program cannot be more than 10 times faster than the sequential program, even if an infinite number of processors is used. Figure 3.2 shows the upper bounds for the theoretical speedup and illustrates that only for parallel proportions close to 100% the speedup grows reasonable with the number of assigned processors. Fortunately, the parallelizable proportion of the subset simulation algorithms can become close 100% - depending on the problem - because the proportion will increase for larger engineering problems, since the computational expensive function $r(\cdot)$ is part of parallizable sampling generation proportion. This will be discussed in more detail and illustrated by examples in the evaluation part 4.5.

3.2.2. Multithreading

As processes are defined to perform tasks, threads maybe viewed to perform subtasks simultaneously with other threads. They provide a simple abstraction layer to achieve (pseudo-)parallelism with respect to processor instructions. On a single processor computer threads are executed sequentially by rapidly switching between several threads which may give the illusion of simultaneity. If the number of parallel threads is indeed processed in parallel the actual speedup depends finally on the scheduler of the operating system and the number of physically provided processing units. However, the theoretical speedup factors are upper bounds which are usually not reached, because of different influence factors, e.g. interruption by the operating system, waiting for resources, communication, synchronization, etc.

From the programmers point of view a thread is defined by a code procedure which is, for example, initiated by the main program and then runs completely independently from the the main program. The thread finally ends and looses all its resources when the end of the thread procedure is reached. To avoid time consuming regeneration of threads in between subset simulation levels the threads are designed as worker threads being in an endless loop, either awaiting a new tasks or an external abortion signal. The typical scheme is shown in the following pseudo code example:

Algorithm 6 Processing Scheme of a Worker Thread.

```
while (true)                                              ▷ endless loop
    sleep and wait to be woken up from the main thread
    if (signal to end the thread) return      ▷ leave the loop to end the thread
    work: start the subtask procedure
    signal the main thread that the subtask is finished
end while
```

All simulation methods have been parallelized using this kind of abstract procedure. The only difference is the subtask procedure, which needs to be defined additionally to the sequential algorithm. The described worker thread scheme is therefore just a generalization to make threads possibly reusable if similar work needs to be done several times. This scheme is beneficial for all subset simulation methods, since they perform a similar sampling process in every stage.

3.2.3. Parallelization of the Subset Simulation Process

What all subset simulation methods have in common is the easily parallelizable Monte Carlo simulation in the first stage. In particular, that will be the sampling generating step in section A of algorithms 2, 4 and 5. Generally, all sampling processes are implemented to distribute the workload equally among a given number of threads. If the processing speed of the parallel components is different, the number of threads can be simply increased. A subtask is defined as the generation of $\lfloor N_i/v \rfloor$ samples of a particular subset simulation stage i, where N_i is the total number of samples to be generated in stage i and v is the number of subtasks among the workload is distributed. Due to the fact that N_i/v may be a non-integer, the generation of possibly remaining $a < v$ samples is distributed among the first a working tasks.

For the other simulation stages of the subset simulation methods the parallelization is more difficult, since the sampling results have to be synchronized at the end of each stage. Following the most important rule in parallelization - the result of a parallel program must be equal to its sequential equivalent - read and write processes on shared data structures must be protected by some kind of synchronization mechanism. This is usually achieved by exclusive resource usage, i.e. resource locking, which in turn has performance disadvantages, because threads have to wait until resources are available. However, for the simple implementation of the subset simulation methods it is possible to construct a completely wait free parallel section, that is, without any resource locking. For the memory optimization, explained in section 3.1, resource locking becomes necessary for the sample result list (priority queue), but further resource locking may be avoided by the approach described in the following.

During the sampling generation the threads usually have to write into the same result list. This can be avoided by providing each thread its own result list, which in turn can be easily connected after the parallel code section is left. If the same approach of moving the synchronization of data out of the parallel code sections is also used for all other output data (e.g. statistic and performance data) there will be no concurrent writing of data. All output data then have to be merged after leaving the parallel code section.

Due to the separation of sampling method and sampling problem the sampling method needs to call unknown user code of the sampling problem within the parallel code section. Since this is certainly the computationally most expensive part, this code section should not be locked entirely and two possibilities remain to deal with that problem: 1. The user needs to provide threadsafe code for the sampling problem, or 2. All data structures of the sampling problem which may be written concurrently during the sampling process need to cloned before and synchronized afterwards to avoid race conditions. Furthermore, if the code sections of the sampling problem act as a state machine, which may be favorable for some numerical integration methods, option 2 remains as the only possibility and also provides easier programming conditions for the end user. In sum, the parallelization requires the following additional effort:

- *Before entering the parallel code section*, the workload must be split and distributed among all threads. Additionally, each thread must be provided with its own result data structure and its own copy of volatile data of the sampling problem.
- *After leaving the parallel code section*, the private result data of each thread must be merged into a single result.

This approach makes it easier for the user (i.e. the one who provides the sampling problem) to create code which is able to run in parallel and this approach is even necessary, if the functions of the sampling problem act as a state machine. As an example, for a dynamic subset simulation problem, the user needs to provide a response function, which calculates the response of just a single time point given the previous time points and responses. This is because for SS/S and SS/H the samples and responses are calculated partially. If, for example, the Newmark method is assigned for the response calculation, the first and second derivative of the response from the last time point is necessary for the calculation. To save memory resources, it is natural to handle these values as a variable state during the response calculation of a whole sample (part). While this works fine for the sequential program, the parallel simulation of different samples will use the same function to generate them pointwise and would corrupt the response function's state values. The user is therefore provided with a general data structure for data that may be used concurrently during parallel computation. This data structure is automatically copied for parallel processing and assigned to the corresponding thread. However, the user is free to use this data structure and can additionally or alternatively use common protection methods to avoid race conditions. Depending on the problem under study and its particular implementation it is up to the user to decide whether to use the proposed cloning scheme of sensitive data structures or to use resource locking techniques which may slow down the parallel computing. Finally, it is noted that the data cloning approach needs extra memory resources and may need extra calculation time after the parallel section. However, besides the simpler implementation, a gain in performance may be achieved especially for data fields which are accessed frequently by a large number of parallel processes.

Figure 3.3 illustrates the described process of automatic user data multiplication for the case that the workload is distributed over n threads. In the simplest case, the user simply moves the data storage of sensitive data into the provided container and during the sampling process each user method is automatically provided with the correct copy in the case of parallel computation. Consequently, this approach facilitates parallel computation from the user point of view as in most cases the user does not even take notice of possible parallel computation. In practice, there may remain some simple rules, which can be found in the code documentation, that is, for example, each of the data structures shown in figure 3.3 is provided with a private instance of a random generator and the user is asked to use them if necessary, since their functions are not thread-safe.

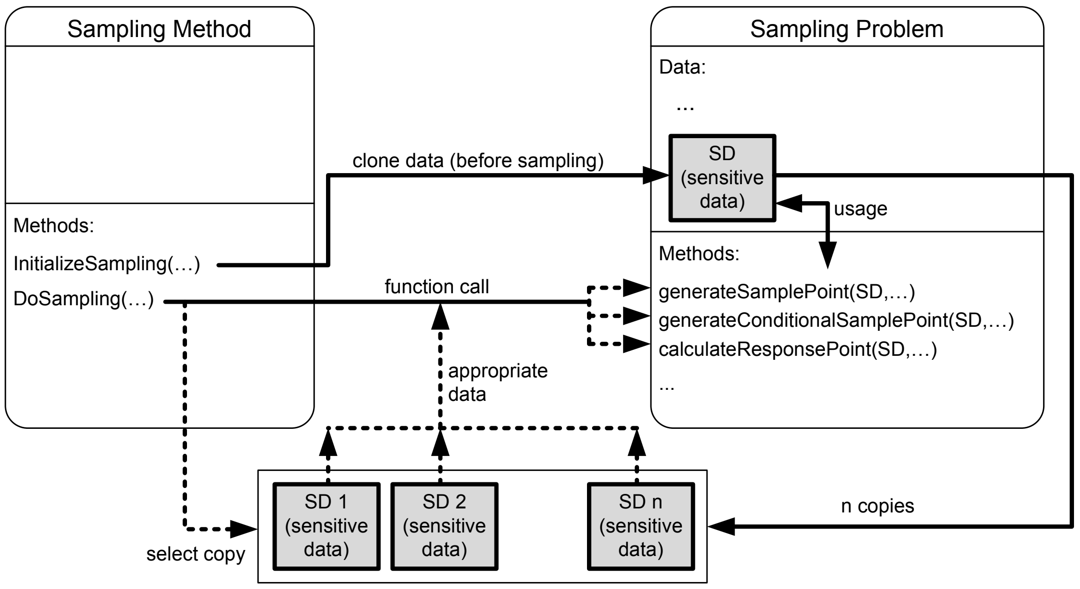


Figure 3.3.: Data processing scheme during parallel computation.

As mentioned before, due to the basic similarity of the subset simulation methods their parallelization is analog to the following example which shows a part of the subset simulation algorithm with splitting. The first part, the Monte Carlo simulation is not shown and needs to be parallelized separately with own functions and threads. In the second part only the interior code section of the sampling generation loop can be parallelized as shown in the following.

Algorithm 7 Part of subset simulation with splitting (Algorithm 4).

```
  ⋮
B. Stage $i$: Estimate $P_i$ (i=1,...,m-1) and possibly calculate new subsets recursively:
  while ($b_i < b$)                                        ▷ a new subset can be opened up
    $\hat{P}_i = p_0$                                       ▷ probability is predefined by $b_i$
    $R_i$ trajectories exceeded threshold $b_i$: assign them as mother
    trajectories $\left\{x_M^{(i+1,k)}(t)\right\}_{k=1}^{R_i}$ and record their first passage points $\left\{S^{(i+1,k)}\right\}_{k=1}^{R_i}$
    $i := i + 1$                                            ▷ open up a new subset
    forall $N_i - R_{i-1}$ offsprings that need to be created:
      // »»» parallelizable code section starts
      Uniformly select a mother trajectory $x_M$ having index $k$
      Create offspring as $x_O = \left\{x_M^-, \widetilde{x}^+\right\}$ by applying $r(\cdot)$ to the partial mother
      excitation $u_M^-$ and the new simulated partial excitation $\widetilde{u}^+ \sim p(u^+|u^-)$,
      which starts after the first passage point $S^{(i,k)}$
      // ««« parallelizable code section ends
    end for
    Sort samples $\left\{x^{(i,k)}(t)\right\}_{k=1}^{N_i}$ in ascending order wrt. $\max_t\left[g\left(x^{(i,k)}(t)\right)\right]$
    $b_i := \max_t\left[g\left(x^{(i,j)}(t)\right)\right]$ with $j = \lfloor(1-p_0)\cdot N_i\rfloor$, $R_i := p_0 N_i$
  end while
  ⋮
```

The parallel section needs to be encapsulated in an extra method and will finally be called by the assigned worker thread. After all, the parallel version of the algorithm is illustrated in the following partial algorithm

Algorithm 8 Parallelized part of subset simulation with splitting (Algorithm 4).

```
  ⋮
B. Stage $i$: Estimate $P_i$ (i=1,...,m-1) and possibly calculate new subsets recursively:
  while ($b_i < b$)                                        ▷ a new subset can be opened up
    $\hat{P}_i = p_0$                                       ▷ probability is predefined by $b_i$
    $R_i$ trajectories exceeded threshold $b_i$: assign them as mother
    trajectories $\left\{x_M^{(i+1,k)}(t)\right\}_{k=1}^{R_i}$ and record their first passage points $\left\{S^{(i+1,k)}\right\}_{k=1}^{R_i}$
    $i := i + 1$                                            ▷ open up a new subset
    Split workload and prepare private data structures for parallel threads
    Start all threads to run parallel code sections
    Wait until all threads have finished
    Merge data results of all threads
    Sort samples $\left\{x^{(i,k)}(t)\right\}_{k=1}^{N_i}$ in ascending order wrt. $\max_t\left[g\left(x^{(i,k)}(t)\right)\right]$
    $b_i := \max_t\left[g\left(x^{(i,j)}(t)\right)\right]$ with $j = \lfloor(1-p_0)\cdot N_i\rfloor$, $R_i := p_0 N_i$
  end while
  ⋮
```

Before the synchronization can start one needs to ensure, that all worker threads have finished their work. Therefore, the main thread has to wait for all workers to finish. The rest of the algorithm is similar to the sequential version.

3.2.4. Parallelization of Importance Sampling

The importance sampling technique which uses ISD B is already very efficient, but system analysis is still needed. In importance sampling the system analysis is a preprocessing step and is not part of the actual sampling process. As shown in the importance sampling algorithm in subsection 2.3.4 the whole importance sampling procedure can be divided into to three steps. All of these steps need to calculate values which are independent of each other, but the results of previous step (if any) are needed. Hence, each of the steps can be parallelized by itself, but the steps need to stay in a sequential order. The actual run time proportion of each step depends on the sampling problem and the ISD. For instance, the system analysis (step one) and the value precalculation (step 2) has been found to have the largest program proportion for ISD B for all test examples and reasonable sample numbers (≤ 500). This may change for larger problems. In contrast the more complex estimator evaluation for A usually makes the sampling process (step 3) to the essential part of the total run time. In the implementation all three steps of the algorithm have been parallelized in a similar way as discussed in the last subsections. Since the parallelization also needs additional computation time for the workload distribution, the speedup for a particular step may also depend on the problem under study and the selected ISD.

3.2.5. Scalability of the Methods

In this work the parallelization of the sampling methods has been focused on single computers with several processing units, that is, small-scale parallelization. This drastically simplifies the effort for data exchange and synchronization. Since the data exchange in large scale computation is usually the bottleneck of efficiency, the parallelization scalability of a particular method is therefore mainly defined by frequency of necessary data exchange and by the amount of data that needs to be exchanged.

The subset simulation methods are well scalable up to certain number, since most of the data can be generated independently and does not need to be exchanged. Data synchronization is only needed in the change over from one stage to the other. If the number of parallel processes is less or equal then the considered number of mother samples in each stage, the amount of necessary data exchange will be limited, but will increase noticeably is the number of processors is larger. Another important issue in this aspect is the storage and re-usage of calculated responses. For large degree of freedom systems the amount of data for the responses will be large and this optimization will highly influence the scalability and may not be considered.

For importance sampling, one can generally say that the scalability of the parallelization of the importance sampling method is worse compared to the subset simulation methods. Although, the three steps of the method can be parallelized separately, all steps have strong data dependencies. For small scale parallelization as tested in this work, this is not an issue since the data exchange can be efficiently realized by means of shared memory resources. For large scale distributed computing, however, the method will need a huge effort for data transfer and synchronization.

3.3. Software Architecture

3.3.1. Encapsulation of Generality

From the mathematical point of view the increasing efficiency of the sampling methods described in chapter 2 is bought with the loose of generality. From the engineering point of view the methods are stepwise specialized for reliability estimation of dynamic structures, i.e. the first excursion problem. While MCS simulation is generally applicable to any kind of probability distribution, SS/MCMC is only applicable to problems which need to estimate the conditional probability $P\left(x \mid g\left(x\right) \geq b\right)$. While the shape or type of the samples x can be arbitrary, SS/S and SS/H further assume that samples are time series, which can be split and which are long enough that a re-simulation of the last trajectory part is sufficient to explore the failure probability space over a few sampling stages. The following figure 3.4 gives an overview about the generality of the sampling methods.

Note, that pure importance sampling has the same generality as Monte Carlo simulation, but it is difficult to find an appropriate importance sampling density for more general cases. Therefore, this figure shows the generality of the importance sampling method which has been explained in this thesis.

Clearly, only simulation methods of the same generality and all more general simulation methods are applicable to a given failure estimation problem. Thus, if this structure is also applied to the computing software, the program will be able to know which simulation methods are applicable and any of these methods can be applied without changing the definition of the sampling problem.

3.3.2. Software package

The software package has been designed to achieve a maximum value of code reusability and providing an efficient implementation of all simulation methods at the same time. Figure 3.5 illustrates the most important classes with respect to the sampling methods discussed before. Furthermore, it shows the type of sampling problems covered by the different methods and their direct relation to the program source code. Note that the class structure directly maps the generality of the methods (compare figure 3.4). For the sake of readability and brevity more detailed information about the software can be found in the source code documentation which consists of a fully detailed UML-diagram containing all methods and data fields.

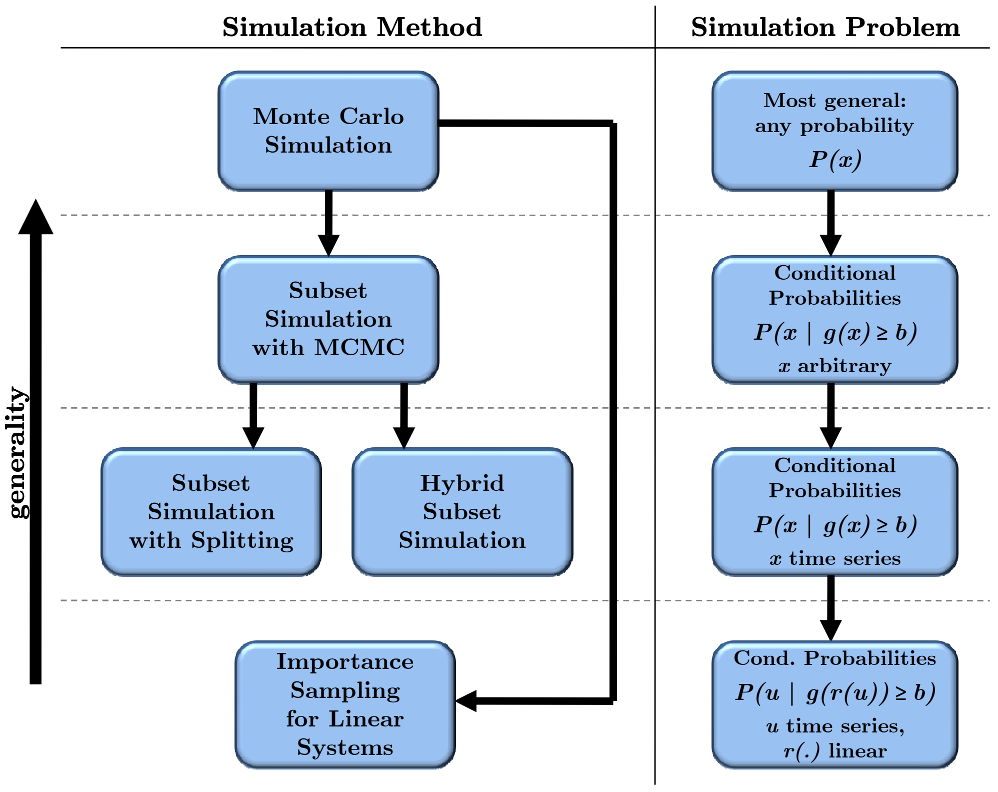

Simulation Method
Simulation Problem
Monte Carlo Simulation
Subset Simulation with MCMC
Subset Simulation with Splitting
Hybrid Subset Simulation
Importance Sampling for Linear Systems
generality
Most general: any probability $P(x)$
Conditional Probabilities $P(x \mid g(x) \geq b)$ x arbitrary
Conditional Probabilities $P(x \mid g(x) \geq b)$ x time series
Cond. Probabilities $P(u \mid g(r(u)) \geq b)$ u time series, $r(.)$ linear

Figure 3.4.: Generality overview of the presented methods.

The encapsulation of different levels of generality facilitates code reusability, makes it easier to add new sampling methods or problems and it abstracts the common properties of sampling methods and problems which makes it easier to implement graphical user interfaces (GUI). Furthermore, the proposed software structure separates the sampling method from the problem to be solved. This highly increases the reusability of source code since it is often favorable to solve different sampling problems with the same estimation method or the other way around: investigate different sampling methods for a particular sampling problem. The introduced abstraction layer simplifies both directions of testing.

For instance, for all sampling methods described in this thesis any sampling problem must provide the following properties:

- The generation of a sample $u \sim p(u)$
- The calculation of a response x for a given sample u, i.e. function $r(\cdot)$

The type of the samples and responses can be arbitrary and therefore these requirements are extended for the subset simulation methods by a partial order relation over $\mathcal{X}$. This relation is necessary to sort the samples and to identify the mother samples for the next sampling stage. The partial order relation is defined by $q : \mathcal{X} \times \mathcal{X} \to \{-1, 0, 1\}$, where -1 and 1 respectively mean that the first argument is smaller or larger than the second and 0 means equality of the arguments. The subset splitting method again extends the

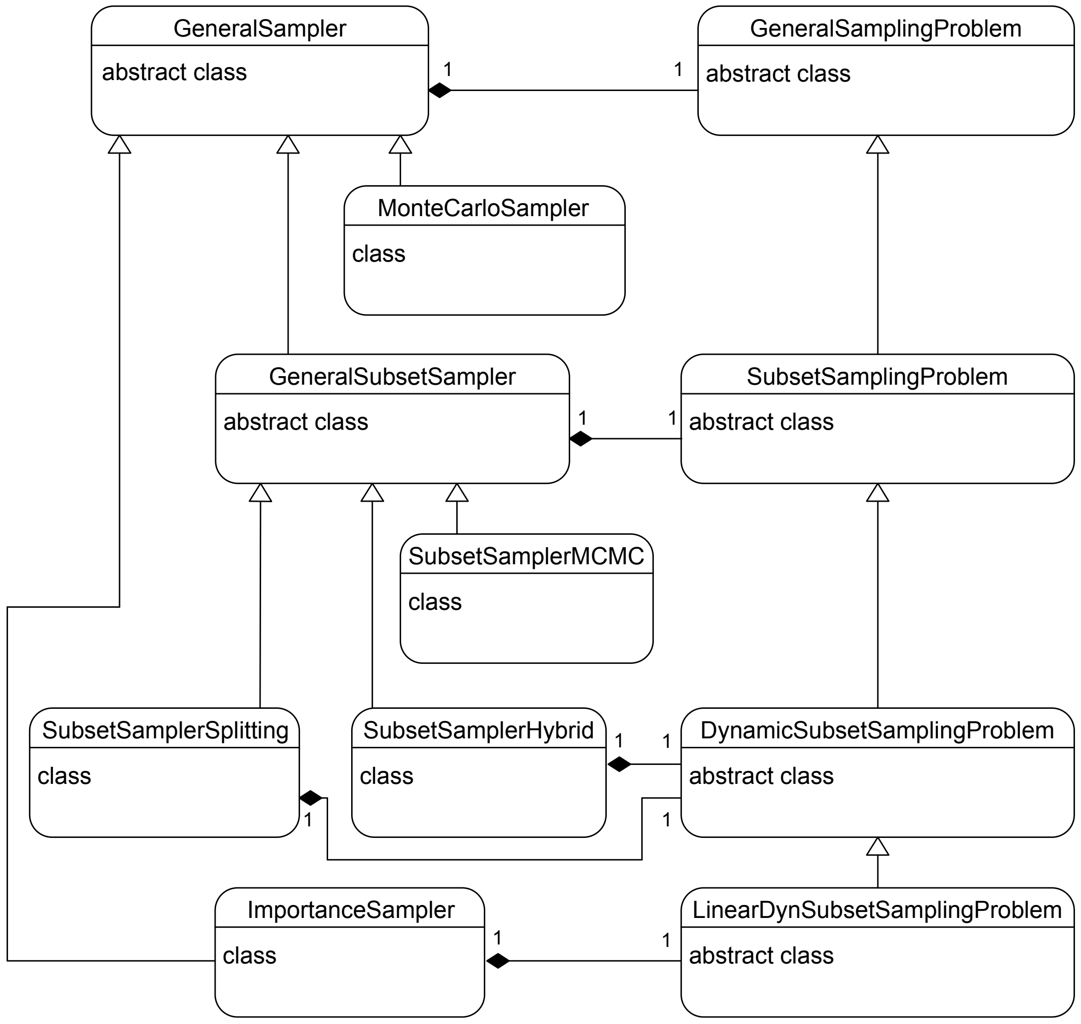


Figure 3.5.: UML-diagram showing the most important classes of the software package and their relations.

requirements to provide the sample generation and response calculation functions for a single time step, because this method works upon time series. In this way each sampling method defines its minimum requirements and any sampling problem which satisfies these requirements can be solved with the corresponding or more general sampling method.

The software package has been developed in C#, because it drastically simplifies memory management, GUI-development and generally accelerates the software development process. It is widely platform independent, capable of being integrated with other programming languages and easily portable. A very simple, but powerful and extendible graphical user interface (GUI) has been developed in order to simplify the testing process of examples and methods.

4. Evaluation

This chapter consists of several experiments to investigate the efficiency advantages of the sampling methods. The focus of this chapter is to show similarities, differences and general properties of the sampling methods. Therefore all examples are first explained and later used to investigate particular properties.

The subset simulation algorithms are derived in a general way so that a particular number of samples N_i can be used to estimate the partial failure probability P_i in stage i. Since the final estimator is the product of each intermediate failure probability it is desirable that their contribution is approximately similar, which can be achieved if the partial failure probabilities are chosen a priori. Then, it is natural to estimate each of the partial failure probabilities with the same accuracy, which means that the number of samples in each stage can be chosen as $N = N_1 = N_2 = \cdots = N_m$. Therefore, this choice has been made for all of following experiments.

4.1. Example 1: Simple Benchmark

To show the principles of the subset simulation method and to illustrate intermediate outcomes, the SS/MCMC method is applied to a simple 2-dimensional numeric example. Two input variables u_1, u_2 are considered in this example, which are summarized by the 2×1 vector u (thus, the input space is $\mathcal{U} = \mathbb{R}^2$). The system response function $r(\cdot)$ returns a 1-dimensional vector ($\mathcal{X} = \mathbb{R}$) and is defined as

$$x = r(u) = \frac{2}{3}u_1u_2 + 5 \cdot e(u, 1, 1) + 8 \cdot e(u, -2, 0.5) + 15 \cdot e(u, 2, -2) \tag{4.1}$$

where $e : \mathbb{R}^{2\times1\times1} \to \mathbb{R}$ is defined as the following exponential function:

$$e(u, a_1, a_2) = \exp\left[-(u_1 - a_1)^2 - (u_2 - a_2)^2\right] \tag{4.2}$$

The input domain for both input variables is defined by the interval $[-3, 3]$. The uncertain input is modeled with a standard Gaussian distribution with zero-mean and unit variance, that is, $p(u_j) = q_j(u_j \mid 0)$ with $\sigma_j^2 = 1, \quad j = 1, 2$ according to Equation (4.3).

The limit state function is defined by the identity function $g(x) = x$ and its limit value to define a system failure is $b = 10$. The adaptive proposal density function for the subset simulation procedure is also modeled by a Gaussian distribution for each dimension j:

$$q_j\left(u_j \mid y_j\right) = \frac{1}{\sqrt{2\pi}\sigma_j} \exp\left[-\frac{1}{2}\frac{\left(u_j - y_j\right)^2}{\sigma_j^2}\right] \tag{4.3}$$

During MCMC simulation the Gaussian is centered at the mother sample and the same variance $\sigma_j^2 = 1$ as for the input is used for both dimensions $j = 1, 2$.

The probability for a system failure P_F is estimated with SS/MCMC using the partial failure probability value $p_0 = 0.1$. To estimate the partial probabilities $N = 3000$ samples are used in each sampling stage. The number of stages end up with $m = 4$ during the simulation, which results to the overall number of samples $N_T = 11\,000$

Sampling Process

The sampling process and generation of conditional samples according to the SS/MCMC method is illustrated in the following Figure 4.1. The goal is to estimate the probability that the response exceeds $b = 10$ being shown as a plane in Figure 4.1(a).

Furthermore, Figure 4.1(a) shows the response function and the set of samples, which has been generated by direct MCS as the first stage of SS/MCMC procedure. The samples are hence distributed all over the input domain according to the input PDF $p\left(u\right)$. All these samples are used to estimate the first partial failure probability value P_1, which is the probability that the response exceeds b_1. The samples exceeding b_1 are used as mother samples to generate more conditional samples which are also distributed as $p\left(x|F_1\right)$ (shown in (b)). This process is continued until the next intermediate threshold is larger than b, which is the case 4.1(d). The intermediate limit value is $b_3 = 8.95$. After the generation of more samples distributed as $p\left(x|F_3\right)$ the next intermediate threshold according to p_0 is found to be $b_4 = 11.67$, which is greater than the limit value $b = 10$. The samples generated in this stage are hence sufficient to estimate the probability $P\left(F|F_3\right)$, because its value is smaller than p_0.

The figure is meant to illustrate the principles of subset simulation which are the same for the three variants of the method. The difference is only defined by the way, the conditional samples are generated. The methods SS/S and SS/H are specialized for high dimensional inputs and are thus not applicable to this sample problem.

The results of the estimation are compared to the ones obtained using direct MCS with a number of samples $N_T = 100\,000$. Figure 4.2 shows the failure probability graph for both estimations. The failure probability estimations for the limit value $b = 10$ are 2.176×10^{-3} and 2.221×10^{-3} for SS/MCMC and direct MCS, respectively.

The figure shows that both methods lead to the same results as the failure probability graphs are nearly identical. The 'bumps' in the failure probability graph are due to the shape of the objective function. More interesting than the failure probability graph is the coefficient of variation of both estimators in comparison, which is shown in Figure 4.3.

Additionally to the c.o.v. values obtained from the estimation the figure shows the

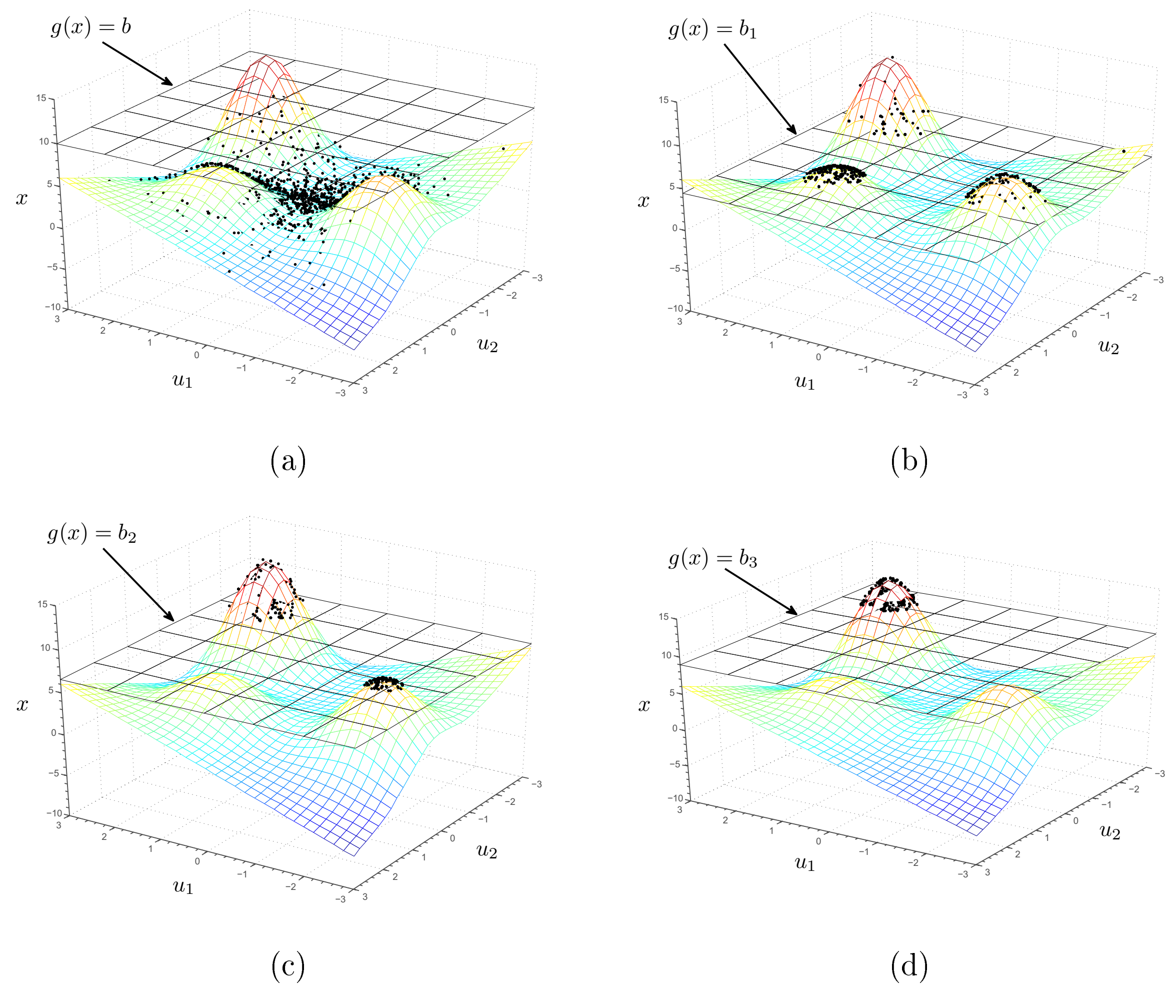


Figure 4.1.: Subset sampling process during 4 stages: (a) stage 1: MCS samples and the limit of interest ($b = 10$); (b),(c),(d) stage 2-4: generation of conditional samples with response values greater than b_1, b_2, b_3, respectively.

theoretical c.o.v. for a total number of samples $N_T = 11000$ and $N_T = 100000$. One can see that the c.o.v. obtained by SS/MCMC increases linearly over the logarithmic scale of the failure probabilities while the c.o.v. values from direct MCS increases exponentially. For very small failure probabilities the c.o.v. values from SS/MCMC even outperform the ones of MCS with 100 000 samples, which used almost 10 times more samples and thus response function evaluations. The figure also shows the theoretical c.o.v. values that would be obtained if direct MCS is used with the same number of samples, i.e. $N_T = 11\,000$. This illustrates the advantage of the method to estimate small failure probabilities. However, the smallest probability that could be estimated by MCS with such a number of samples is $P_F = 0.91 \times 10^{-4}$.

The example has shown, that SS/MCMC is much more efficient to estimate small probabilities than direct MCS. Only a small fraction of samples is necessary to obtain the same estimation results. This is very valuable if the evaluation of the objective

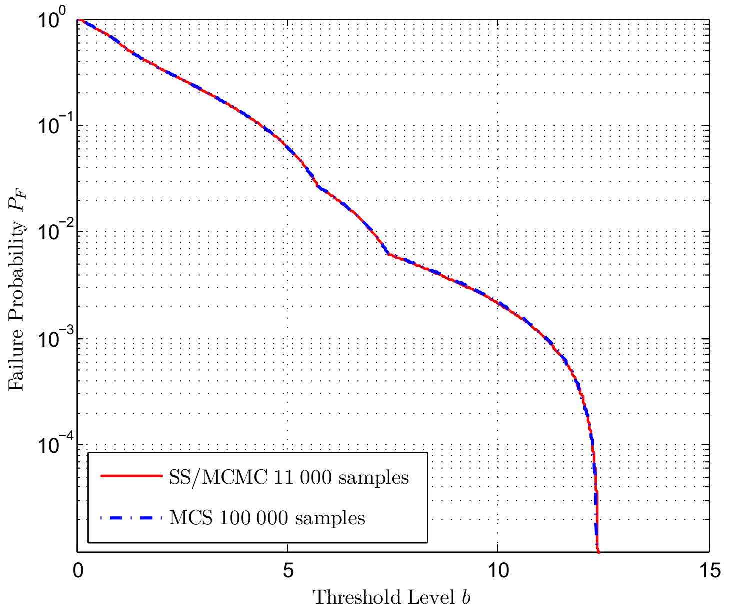


Figure 4.2.: Results of the failure probability estimation.

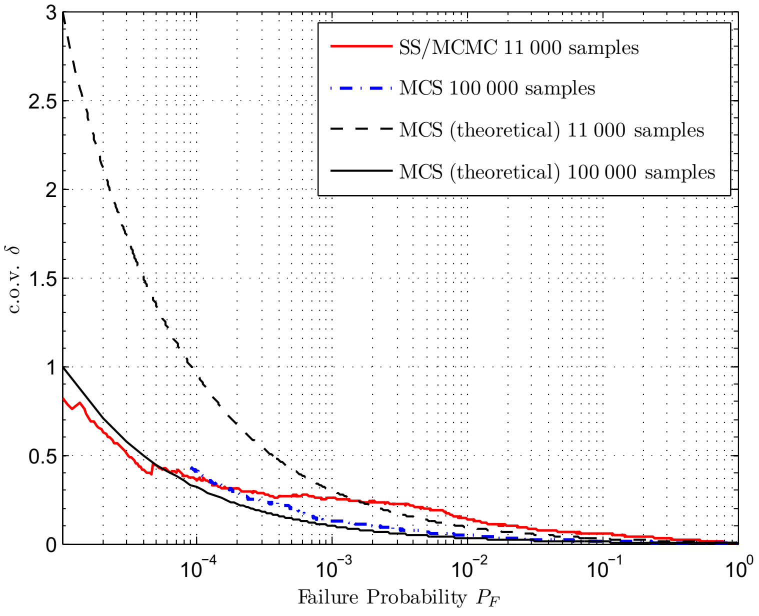


Figure 4.3.: Coefficient of variation for failure probability estimation.

function $r(\cdot)$ is computationally expensive since the function needs to be evaluated once for each sample.

4.2. Dynamic System Examples

Three dynamic systems are studied with the described sampling methods to estimate their first-excursion failure probabilities. The system input $u(t)$ for these examples is a Gaussian white noise process with spectral intensity S_0. Using a sampling interval of $\Delta t = 0.02$s, the system is studied over a time of $T = 30$ s, such that the number of time steps is $n_t = T/\Delta t + 1 = 1501$. The stochastic input $u(t)$ is hence a sequence of random variables $U(t) = \left\{\sqrt{2\pi S_0/\Delta t} \cdot Z(t_l) \ : \ l = 1, \ldots, n_t\right\}$, where S_0 denotes the spectral intensity and $Z(t_l)$ is a sequence of independent and identically distributed standard Gaussian random variables with zero-mean and unit variance.

4.2.1. Example 2: SDOF Linear Oscillator

To illustrate the simulation method on a simple dynamic system, a single-degree of freedom (SDOF) linear oscillator is considered. The mathematical model of the oscillator is shown in Figure 4.4. The behavior of this spring-mass system is described by the mass displacement over time $x(t) \in \mathbb{R}$. The system is subjected to white noise excitation $u(t) \in \mathbb{R}$ with spectral intensity $S_0 = 1\text{m}^2/\text{s}^3$ and the equation of motion is given by:

$$\ddot{x}(t) + 2\zeta\omega\dot{x}(t) + \omega^2 x(t) = u(t) \tag{4.4}$$

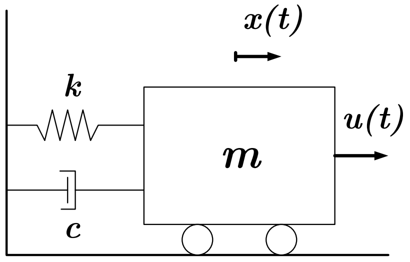


Figure 4.4.: Model of a one degree of freedom oscillator.

where $\omega = 2\pi$ rad/s (1Hz) is the natural frequency of the system and $\zeta = 5\%$ is the damping ratio. The system is assumed to start from rest with initial states $x(0) = 0$ and $\dot{x}(0) = 0$. Conveniently, Equation (4.4) can be solved with numerical integration methods [26]. In the implementation the Newmark method (appendix A.1.1) and the central difference method have been tested. However, the response can also be obtained using the Duhamel integral form (see Equation (2.19)). The corresponding unit response function $g_{ij}(t_l, t_s) = g(t_l - t_s)$ is shown in Figure 4.5 and is described by

$$g(t) = \frac{e^{-\zeta\omega t}}{\omega_d} \sin \omega_d t \tag{4.5}$$

where $\omega_d = \omega\sqrt{1-\zeta^2}$ is the damped natural frequency.

The failure region is defined by the exceedance of the system response over the threshold value b within the duration of study $T = 30$s, that is

$$F = \left\{x \ : \ \max_t |x(t)| \geq b\right\} \tag{4.6}$$

The intermediate failure regions $\{F_i\}$ are defined by the increasing sequence of inter-

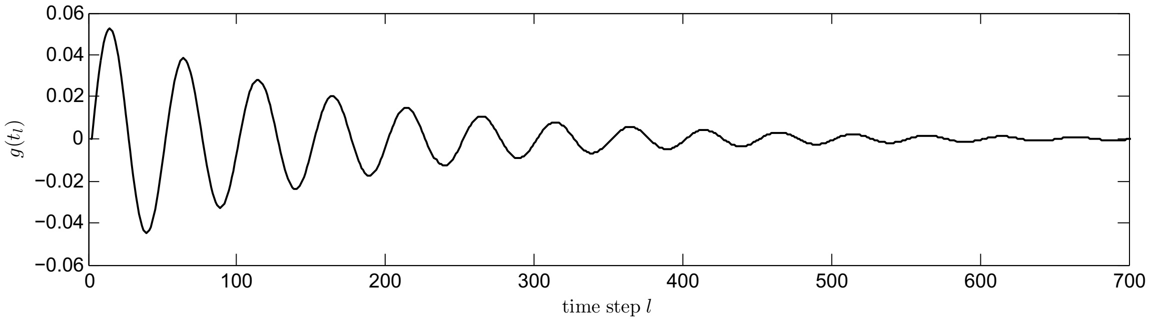


Figure 4.5.: Impulse response function.

mediate threshold values $b_1 < b_2 < \ldots < b_m = b$, which are chosen adaptively during simulation, such that $\hat{P}_i = p_0 = 0.1 \;\; \forall i = 1, \ldots, m-1$. Thus,

$$F_i = \left\{ x \;:\; \max_t |x(t)| \geq b_i \right\} \tag{4.7}$$

Figure 4.6 shows characteristics of the system response to the white noise input by the mean system response and the corresponding standard deviation. It can be seen, that the standard deviation reaches its maximum quickly, which is due to the damping factor of 5%.

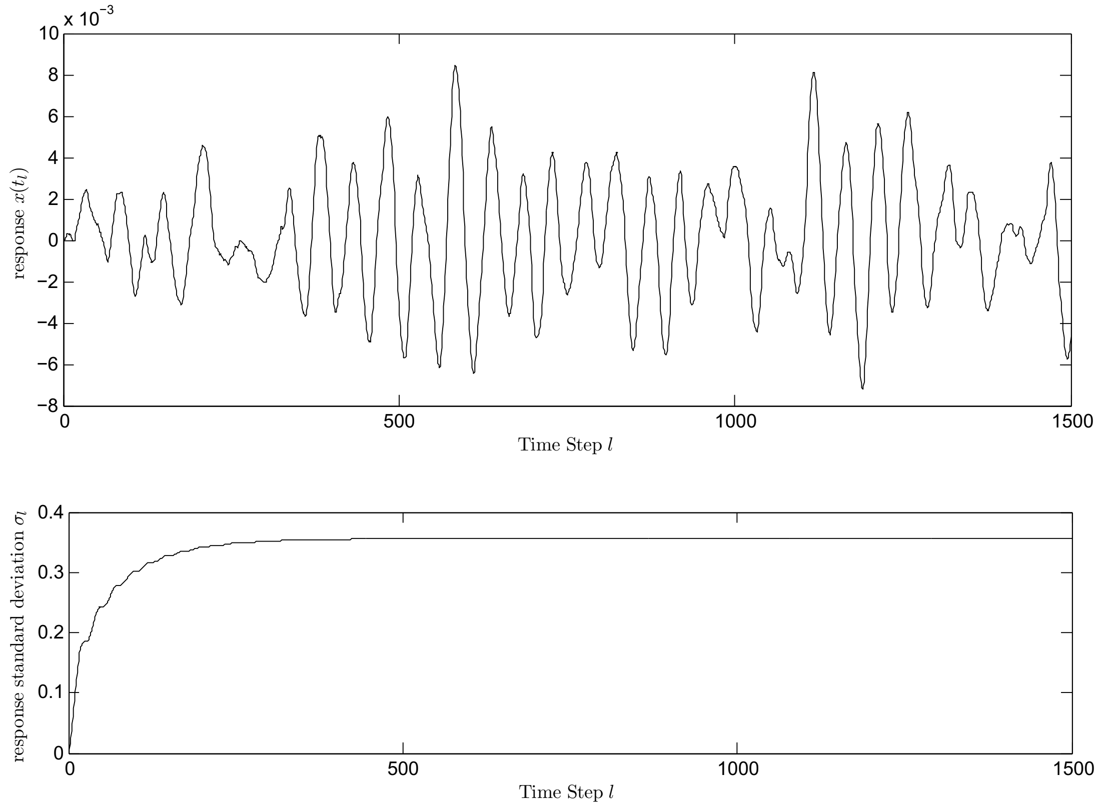


Figure 4.6.: Response mean value and standard deviation of the SDOF oscillator.

4.2.2. Example 3: Linear Shear Building

An n-story linear shear building is subjected to earthquake motion modeled by a non-stationary stochastic process. The various floors are rigid masses and assumed to be damped classically. The columns are flexible but assumed to be massless. Given are the floor masses for each story $m_1, m_2, \ldots, m_n$, the linear interstory stiffness values $k_1, k_2, \ldots, k_n$ and the damping values $c_1, c_2, \ldots, c_n$. The system is modeled as a n-degree of freedom oscillator and is shown in Figure 4.7 To describe the dynamics of the system the mass, stiffness and damping values are written as matrices:

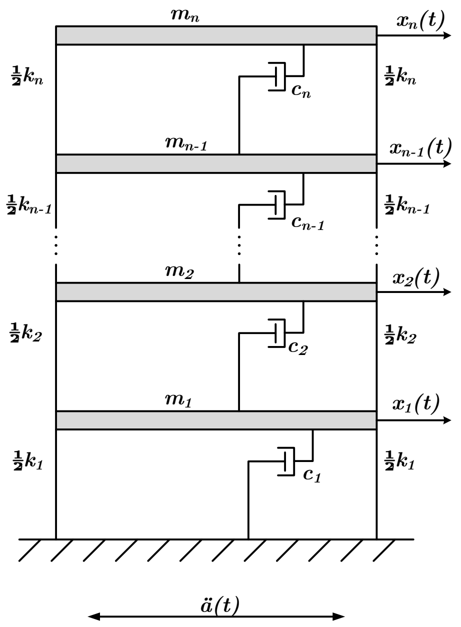


Figure 4.7.: n-story linear shear building model.

$$[M] = \begin{bmatrix} m_1 & 0 & \cdots & 0 \\ 0 & m_2 & 0 & \vdots \\ \vdots & 0 & \ddots & 0 \\ 0 & \cdots & 0 & m_n \end{bmatrix}_{n\times n} \tag{4.8}$$

$$[K] = \begin{bmatrix} k_1+k_2 & -k_2 & \cdots & 0 \\ -k_2 & k_2+k_3 & \ddots & \vdots \\ \vdots & \ddots & \ddots & -k_n \\ 0 & \cdots & -k_n & k_n \end{bmatrix}_{n\times n} \qquad [C] = \begin{bmatrix} c_1+c_2 & -c_2 & \cdots & 0 \\ -c_2 & c_2+c_3 & \ddots & \vdots \\ \vdots & \ddots & \ddots & -c_n \\ 0 & \cdots & -c_n & c_n \end{bmatrix}_{n\times n}$$

The displacement of each story, denoted by the $n \times 1$ vector $\{x(t)\}$, is described by n-coupled equations of motion which are in matrix form:

$$[M]\{\ddot{x}(t)\} + [C]\{\dot{x}(t)\} + [K]\{x(t)\} = -[M]\{1\}\ddot{a}(t) \tag{4.9}$$

where $\ddot{a}(t)$ denotes the base excitation.

With the solution of the eigenvalue problem $([K] - \omega^2 [M])\phi = 0$ one obtains n eigenvalues ω_i^2 and corresponding eigenvectors ϕ_i, which can be arranged in columns to form the $n \times n$ modal matrix $[\phi] = [\{\phi_1\}, \{\phi_2\}, \ldots, \{\phi_n\}]$. With the substitution

$$\{x(t)\} = [\phi]\{\eta(t)\} \tag{4.10}$$

the displacement vector is described by a series of normal modes and with the left-wise multiplication with matrix $[\phi]^T$ Equation (4.9) becomes:

$$[\phi]^T [M] [\phi] \{\ddot{\eta}(t)\} + [\phi]^T [C] [\phi] \{\dot{\eta}(t)\} + [\phi]^T [K] [\phi] \{\eta(t)\} = -[\phi]^T [M] \{1\} \ddot{a}(t) \quad (4.11)$$

Since the eigenvectors $\{\phi_j\}, \ j = 1, 2, \ldots, n$ are mutually orthogonal with respect to the matrices $[M]$ and $[K]$, the matrices in Equation (4.11) take diagonal forms with the following properties:

$$\begin{aligned} \{\phi_j\}^T [M] \{\phi_j\} &= M_j \\ \{\phi_j\}^T [C] \{\phi_j\} &= 2\zeta_j \omega_j M_j \\ \{\phi_j\}^T [K] \{\phi_j\} &= \omega_j^2 M_j \end{aligned} \quad (4.12)$$

where M_j is the jth modal mass and ω_j^2, ζ_j respectively are the eigenvalue and damping ratio corresponding to the eigenvector $\{\phi_j\}$. With these properties Equation (4.11) describes n-decoupled equations of motions:

$$\ddot{\eta}_j(t) + 2\zeta_j \omega_j \dot{\eta}_j(t) + \omega_j^2 \eta_j(t) = -\frac{\{\phi_j\}^T [M] \{1\}}{M_j} \ddot{a}(t) \qquad j = 1, 2, \ldots, n \quad (4.13)$$

The n equations in (4.13) may again be solved with any numerical integration method. Note that the usage of the decomposition method (see e.g. [27]) can be beneficial, if the oscillation of the structure is essentially ruled by the first v of n eigenmodes. Especially for low frequency loadings like earthquakes or wind the computational efficiency can be improved significantly with little loss of accuracy by only calculating these modes [16, p.198]. In this case the response vector $\{x(t)\}$ is approximated by

$$\{x(t)\} \approx \sum_{j=1}^{M \ll n} \{\phi_j\} \eta_j(t) \quad (4.14)$$

and then only M (instead of n) independent single degree of freedom systems have to be solved to obtain a good approximation of the response vector $\{x(t)\}$.

After the eigenmode decomposition, the system behavior is also described by the Duhamel integral (see Equation (2.19)) with the unit response function $g_{i1}(t_l, t_s) = g_i(t_l - t_s)$ which is given as

$$g_i(t) = \sum_{l=1}^{M \ll n} \frac{\phi_{il} \{\phi_l\}^T \{1\}}{\{\phi_l\}^T [M] \{\phi_l\}} \frac{e^{-\zeta_l \omega_l t}}{\omega_{d_l}} \sin \omega_{d_l} t \quad (4.15)$$

where $\omega_{d_l} = \omega_l \sqrt{1 - \zeta_l^2}$ is the damped natural frequency for the underdamped case.

The earthquake motion $\ddot{a}(t)$ is modeled by Clough-Penzien [28] filtered white noise $u(t)$ modulated by envelope function $e(t)$:

$$\begin{aligned} &\ddot{a}(t) + 2\zeta_{s2}\omega_{s2}\dot{a}(t) + \omega_{s2}^2 a(t) = 2\zeta_{s1}\omega_{s1}\dot{a}_1(t) + \omega_{s1}^2 a_1(t) \\ &\ddot{a}_1(t) + 2\zeta_{s1}\omega_{s1}\dot{a}_1(t) + \omega_{s1}^2 a_1(t) = e(t)u(t) \end{aligned} \tag{4.16}$$

The dominant and the lower-cutoff frequencies of the spectrum are $\omega_{s1} = 15.7$ rad/s (2.5 Hz) and $\omega_{s2} = 1.57$ rad/s (0.25 Hz), respectively. The corresponding damping parameters are $\zeta_{s1} = 0.6$ and $\zeta_{s2} = 0.8$.

The envelope $e(t)$ function is assumed to be quadratically for the first 4 s, then be constant at unity for 20 s, and finally decays exponentially starting from $t = 24$s, that is,

$$e(t) = \begin{cases} (t/4)^2 & \text{if } 0 \le t \le 4 \\ 1 & \text{if } 4 < t \le 24 \\ \exp\left[-(t-24)^2/2\right] & \text{if } 24 < t \le 30 \end{cases} \tag{4.17}$$

The spectral intensity for the Gaussian white noise $u(t)$ in (4.16) is assumed to be $S_0 = 2.5 \times 10^{-3}$ m^2/s^3. The envelope function as well as a Gaussian noise sample and its corresponding filter output is shown in Figure 4.8.

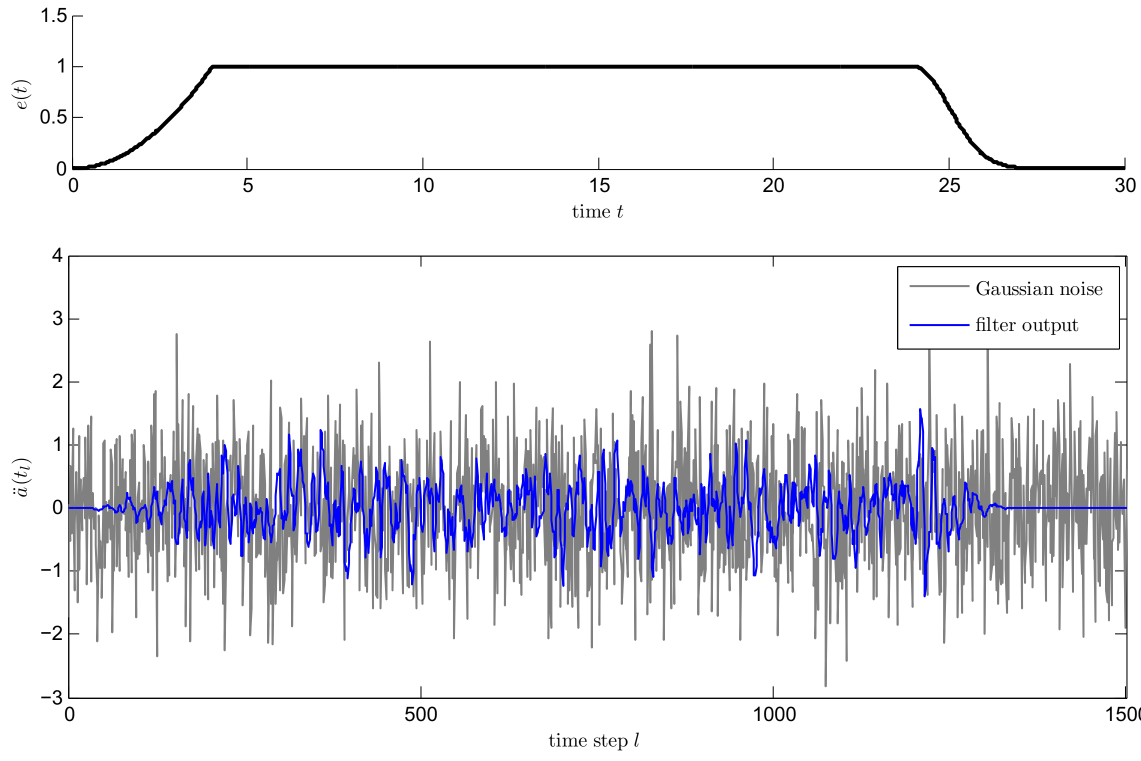


Figure 4.8.: Envelope function $e(t)$ and example input and output of the Clough-Penzien filter modulated by $e(t)$.

The failure event is defined as the exceedance of the interstory drift of any of the stories above a given threshold level b within the first 30 s. Therefore, the system response $\{x(t)\}$ is transformed into local coordinates $\left\{x^L(t)\right\}$ by

$$\left\{x^L(t)\right\} = [T]\{x(t)\} \tag{4.18}$$

where $[T]$ describes a linear transformation matrix given by

$$[T] = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ -1 & 1 & 0 & \cdots & 0 \\ 0 & -1 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & -1 & 1 \end{bmatrix}_{n\times n} \tag{4.19}$$

The interstory drift for the jth story at time t will be denoted by $x_j^L(t)$, $j = 1, \ldots, m$ and the failure region is defined by

$$F = \left\{x \ : \ \max_{j=1,\ldots,n} \max_{t} \left|x_j^L(t)\right| \geq b\right\} \tag{4.20}$$

Again, the increasing sequence of intermediate threshold values $b_1 < b_2 < \ldots < b_m = b$ defines the intermediate failure events $\{F_i\}$. The limit values are chosen adaptively during simulation, such that $\hat{P}_i = p_0 = 0.1 \ \forall i = 1, \ldots, m-1$. Thus, the intermediate failure region for sampling stage i is

$$F_i = \left\{x \ : \ \max_{j=1,\ldots,n} \max_{t} \left|x_j^L(t)\right| \geq b_i\right\} \tag{4.21}$$

Note, that for the Duhamel integral a modified impulse response function can be derived to obtain the response in local coordinates. Also, the filter equations in (4.16) can be transformed into a similar form as the response function in Equation (2.19) to make the filtered input also applicable to importance sampling [5]. However, this transformation requires the solution of a complex eigenvalue problem, which is currently not supported by the software. Therefore, two cases of input excitation are considered in this example. That is, the structure is subjected to

- **Case 1:** modulated Gaussian white noise, i.e. $\ddot{a}(t) = e(t)u(t)$
- **Case 2:** Clough-Penzien filtered modulated Gaussian white noise and $\ddot{a}(t)$ is given as the solution of the filter equations (4.16).

Considered is a 5-story linear shear building with the mass and stiffness values given in Table 4.1. The fundamental frequency of the structure is 1.25 Hz. In all example all modes have been used for the response calculation, i.e. $M = 5$ and the damping ratios for the assumed classical damping are chosen to be 5% for all modes.

story i	mass $m_i \times 10^3$ [kg]	stiffness $k_i \times 10^6$ [N/m]
1	45.4	41.1
2	45.4	38.5
3	45.4	33.4
4	45.4	25.6
5	45.4	15.2

Table 4.1.: Properties of the 5-story linear shear building.

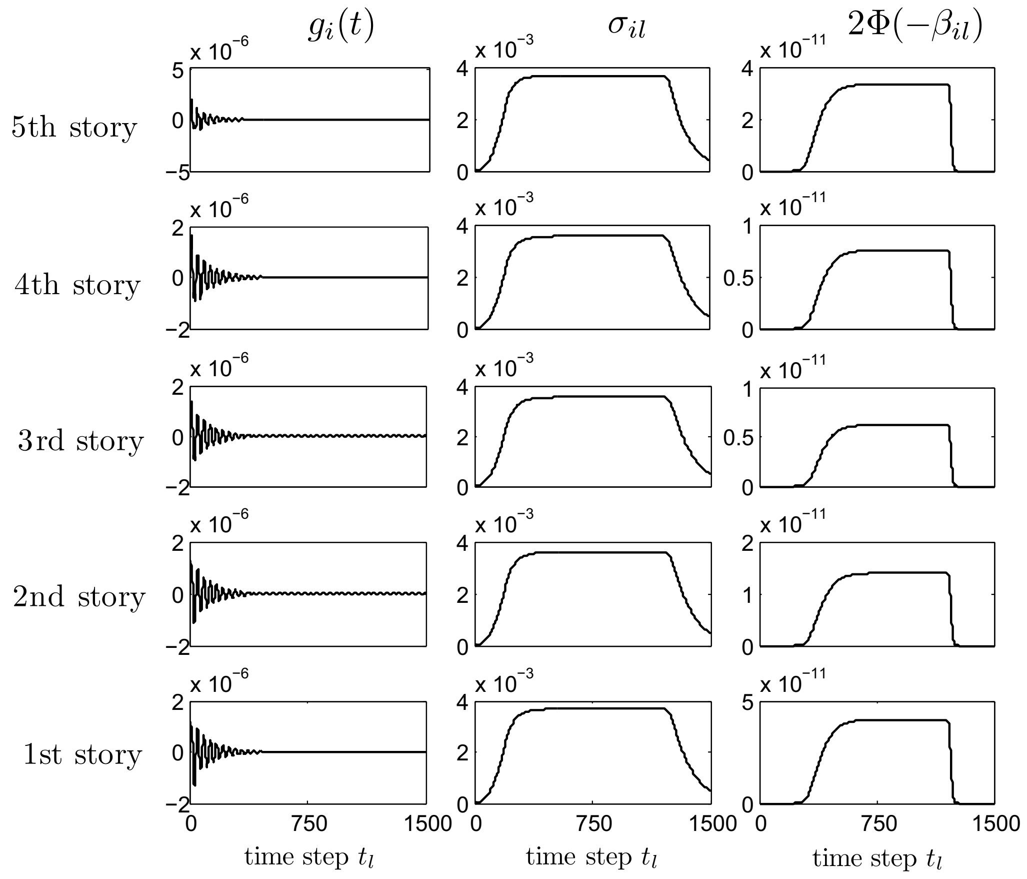


Figure 4.9.: Impulse response $g_i\,(t)$, standard deviation σ_{il}, elementary failure probability $2\Phi\,(-\beta_{il})$ of the the building - case 1.

Some characteristics of the structure are shown in Figure 4.9. The first column shows the impulse response function $g_i\,(t)$ for each story. The second column shows the standard deviation σ_{il} for the interstory drift response of the structure to the modulated Gaussian white noise (case 1), i.e. the standard deviation of the ith component of vector $\left\{x^L\,(t)\right\}$. The third column shows the elementary failure probability that the interstory drift response exceeds threshold level b at time step l, that is $P\left(\left|x_i^L\,(t)\right| \geq b\right) = 2\Phi\,(-b/\sigma_{il})$, with a threshold level $b = 0.024 = 6.5 \times \max_l \sigma_{1l}$.

4.2.3. Example 4: Non-Linear Shear Building

In order to make structures more resistant to earthquakes, it is desirable to reduce their interstory displacements during an earthquake base excitation. This can be achieved, if the structure is enforced with special devices which have a non-linear hysteretic behavior. The linear shear building model from the previous example is therefore extended with a non-linear hysteretic device in each story. The mathematical model of the non-linear shear building is shown in Figure 4.10.

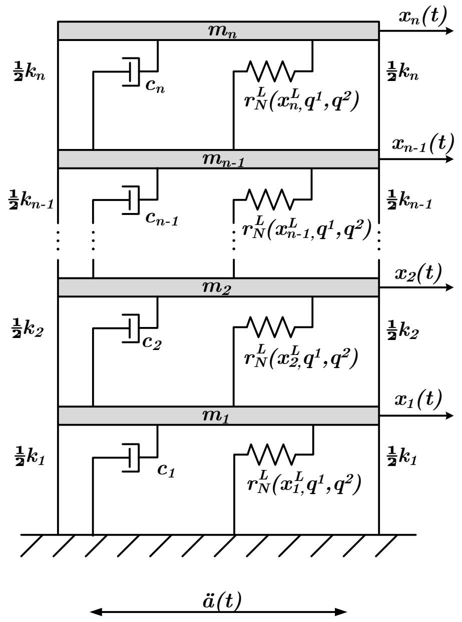


Figure 4.10.: n-story non-linear shear building model.

Accordingly, the equation of motion of the linear model is extended with an additional component which represents the non-linear restoring force $\{r_N(t)\} = [R]\{q(t)\}$ and Equation (4.9) becomes

$$[M]\{\ddot{x}(t)\} + [C]\{\dot{x}(t)\} + [K]\{x(t)\} + [R]\{q(t)\} = -[M]\{1\}\ddot{a}(t) \tag{4.22}$$

where $[R]$ is a transformation matrix and $\{q(t)\}$ is actually a function of the displacements $\{x(t)\}$ and the velocities $\{\dot{x}(t)\}$, i.e. $\{q(\{x(t)\},\{\dot{x}(t)\})\}$. The mass, stiffness and damping matrices are the same as defined in Equation (4.8). The non-linear components in the structure are described locally and the non-linear restoring forces are defined in terms of local coordinates as

$$\left\{r_N^L(t)\right\} = \left[R^L\right]\left\{q\left(\left\{x^L(t)\right\},\left\{\dot{x}^L(t)\right\}\right)\right\} \tag{4.23}$$

The relation between local displacements $\left\{x^L(t)\right\}$ and velocities $\left\{\dot{x}^L(t)\right\}$ to the displacements $\{x(t)\}$ and velocities $\{\dot{x}(t)\}$ in global coordinates can be described by a linear coordinate transformation matrix $[T]$, such that

$$\left\{x^L(t)\right\} = [T]\{x(t)\} \text{ and } \left\{\dot{x}^L(t)\right\} = [T]\{\dot{x}(t)\} \tag{4.24}$$

Therefore, the restoring forces in the global coordinate system are described by their local counterparts as

$$\{r_N(t)\} = [T]^T\left\{r_N^L(t)\right\} = [T]^T\left[R^L\right]\left\{q\left(\left\{x^L(t)\right\},\left\{\dot{x}^L(t)\right\}\right)\right\} \tag{4.25}$$

According to Equations (4.10)-(4.13) the modal analysis of Equation (4.22) can be performed in the same way and finally leads to n-decoupled equations of motions

$$\ddot{\eta}_j(t) + 2\zeta_j\omega_j\dot{\eta}_j(t) + \omega_j^2\eta_j(t) = -\frac{\{\phi_j\}^T[M]\{1\}}{M_j}\ddot{a}(t) - \frac{\{\phi_j\}^T[R]}{M_j}\{q(\{x(t)\},\{\dot{x}(t)\})\}$$
$$j = 1, 2, \dots, n \tag{4.26}$$

Equation 4.26 can then be solved with a numerical integration scheme, but because of the dependence of $\{x(t)\}$ in the right-hand side, the equation must be solved in an iterative scheme which is described in appendix A.1.2.

As mentioned before, the restoring forces for each story are described locally. For simplicity in notation the index for the story is dropped. The restoring force $r_N^L(t)$ is described by

$$r_N^L(t) = k_d\big(x^L(t) - q^1 + q^2\big) = k_d u_N(t) \tag{4.27}$$

where k_d denotes the initial stiffness of the device, $x^L(t)$ is the relative displacement, q^1 and q^2 describe the plastic elongations of the device and $u_N(t)$ is a auxiliary variable which is used for brevity in the following equations. The plastic elongations q^1 and q^2 are specified by the first-order differential equations

$$\dot{q}^1 = \dot{x}^L(t)g_1\left(x^L(t), u_N(t)\right) \tag{4.28}$$
$$\dot{q}^2 = -\dot{x}^L(t)g_2\left(x^L(t), u_N(t)\right) \tag{4.29}$$

where

$$g_1\left(x^L(t), u_N(t)\right) = H\left(\dot{x}^L(t)\right)\Big[H\left(u_N(t) - u_y\right)\frac{u_N(t) - u_y}{u_p - u_y} \tag{4.30}$$
$$\times H\left(u_p - u_N(t)\right) + H\left(u_N(t) - u_p\right)\Big] \tag{4.31}$$

and

$$g_2\left(x^L(t), u_N(t)\right) = H\left(-\dot{x}^L(t)\right)\Big[H\left(-u_N(t) - u_y\right)\frac{-u_N(t) - u_y}{u_p - u_y} \tag{4.32}$$
$$\times H\left(u_p + u_N(t)\right) + H\left(-u_N(t) - u_p\right)\Big] \tag{4.33}$$

$H(\cdot)$ again denotes the Heaviside step function, u_y specifies the onset of yielding and $k_d u_p$ is the maximum restoring force of the device. The restoring force introduces energy dissipation in the system response which causes the non-linearity. The hysteretic properties of the restoring force are illustrated in Figure 4.11, which shows also the meaning of the parameters u_y, u_p and k_d.

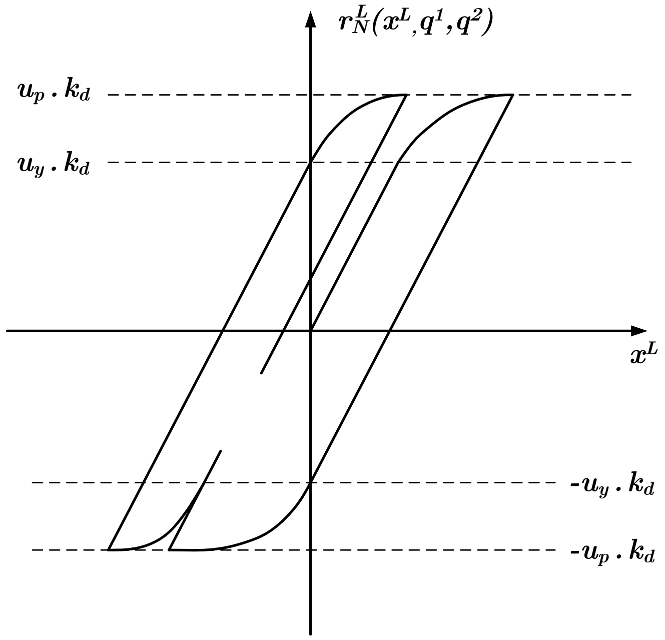


Figure 4.11.: Hysteretic displacement-restoring force.

Considered is the same 5-story building as in example 3, that is, all mass, damping and stiffness values are the same. The earthquake excitation is modeled as Clough-Penzien filtered white noise modulated by envelope function $e(t)$, with the same parameters as in example 3 - case 2. The unfiltered excitation of the non-linear shear building is not considered in this example. The only difference is the additional non-linear device in each story. The parameters for the non-linear device are chosen to be equal for all stories and the following values have been used for all hysteric devices:

$$\begin{aligned} k_d &= 1.0 \times 10^6 \text{ N/m} \\ u_p &= 6.0 \times 10^{-3} \text{ m} \\ u_y &= 0.7 u_p \end{aligned} \tag{4.34}$$

The effect of the non-linear elements is essentially a reduction of the maximum interstory displacement values. This finally leads to a reduction the response standard deviation σ_{il} which is shown in Figure 4.12 for the first story of the building.

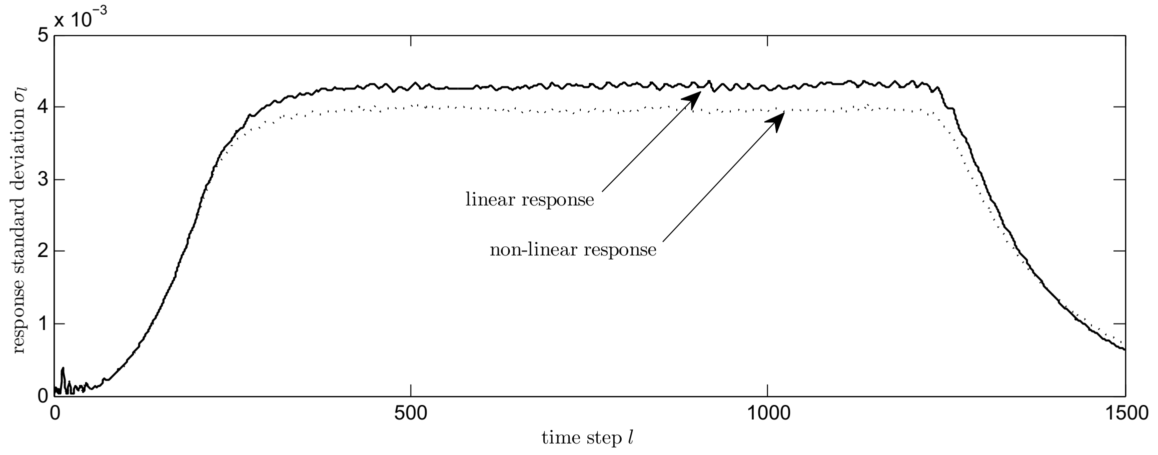


Figure 4.12.: Estimated standard deviation σ_{1l} of the linear and the non-linear shear building for the first story (10 000 samples).

One can see that on average the non-linear response is lower than the linear response. For larger values of the non-linear device stiffness k_d the standard deviation will further decrease. This will be shown later in subsection 4.4.3, Figure 4.42. The response standard deviations for the other stories are very similar and not shown here.

4.3. Comparison of Simulation Methods

To evaluate the correctness of the subset simulation methods, they are compared with standard MCS using 100000 samples. For all subset simulation methods $N = 1000$ samples have been used for each simulation stage and the conditional failure probabilities have been set to $\hat{P}_i = p_0 = 0.1,\ \forall i = 1,\dots,m-1$. This choice for the conditional failure probabilities is suggested by the authors of [4, 13, 1, 14] and is empirically justified in subsection 4.4.1. Consequently, the probability of intermediate failure events in each stage i is 10^{-i} and 100 of the 1000 samples in each level are taken as mother samples for the next stage. For the fixed number of simulation stages $m = 4$ the total number of samples is $N_T = 3700$. For SS/MCMC and SS/H the adaptive uniform proposal function with an interval size of $2l_j = 1$ has been used centered at the current mother sample. This choice is also discussed in subsection 4.4.1. These parameters have been used for all examples applied to the subset simulation methods, if not noted otherwise.

4.3.1. Failure Probability Estimations

This subsection shows results of the failure probability estimations for the described examples. In order to investigate variance and bias of the methods 50 independent simulation runs have been computed with all methods and for all examples. Figure 4.13 shows the average of the probability estimations for the SDOF oscillator example.

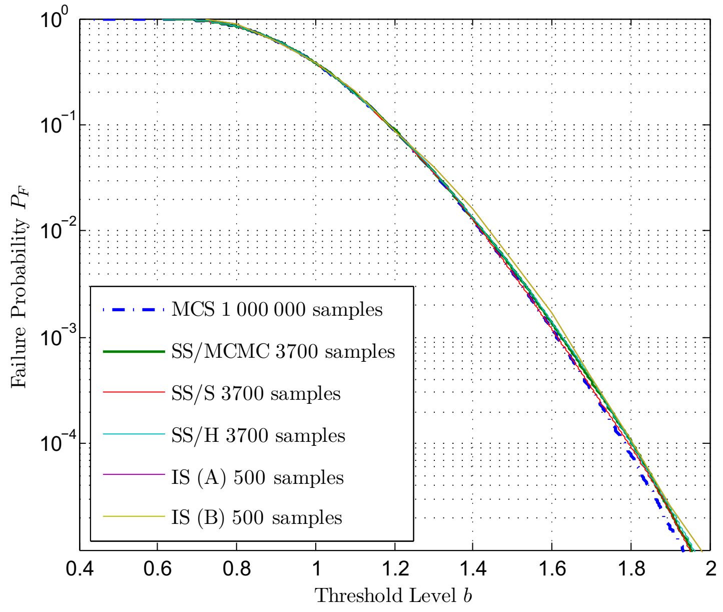


Figure 4.13.: Failure probability estimation for the SDOF example.

One can see in Figure 4.13 that the results of all simulation methods are very similar. Note, that MCS and all subset simulation methods deliver a probability estimation for each sample and its corresponding maximum response value. Therefore, the failure probability graph consists of as the number of samples used for the estimation. In contrast, the probability estimation with importance sampling works differently, every sample contributes directly to the overall failure probability estimation. To obtain an approximation of the whole failure probability graph the simulation process needs to be repeated for different threshold levels. In Figure 4.13 the importance sampling probabilities have been calculated for 17 threshold values between 0.5 and 2.1 in steps of 0.1. The subset simulation methods are therefore better suited for a general probability study of a structural system, while the importance sampling methods are fast to compute a single failure probability, e.g. within an reliability-based optimization framework. An advantage of this kind of probability estimation is that the overall failure probability estimation can be calculated after each sample. This allows to track the convergence of the estimation, which can be used for an automatic termination of the sample process if a desired c.o.v. value has been reached. This approach is not applicable to the subset simulation methods. Figure 4.14 shows example trajectories for single probability estimation runs for three different threshold levels.

It can be seen that the convergence of the importance sampling estimator is really fast, especially for ISD B where about 20 or 30 samples are sufficient to obtain a good estimation for small failure probabilities. Note, that the number of necessary samples to obtain a desired c.o.v. value decreases with smaller failure probabilities for ISD B. The estimator for ISD A also converges fast but generally more samples are needed due to higher variance values. Figure 4.15 shows the importance sampling estimation

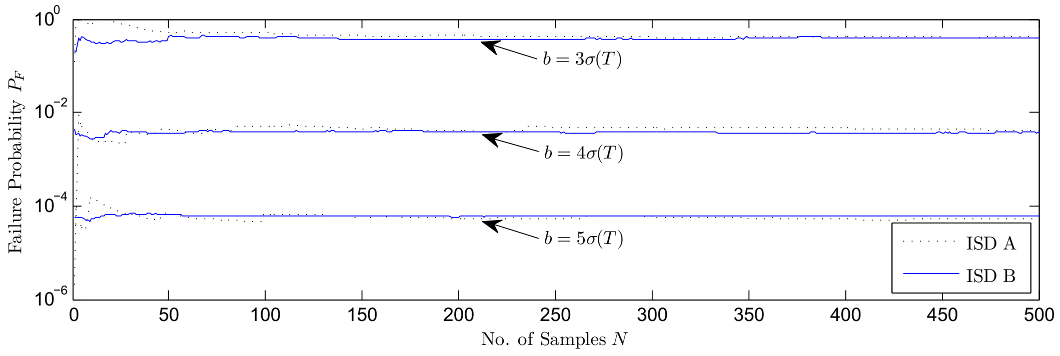


Figure 4.14.: Failure probability estimation history with importance sampling for the SDOF example.

after each sample for the linear shear building example (case 1). One can see that the estimation convergence for the multi-degree of freedom system is similar to the single degree of freedom system.

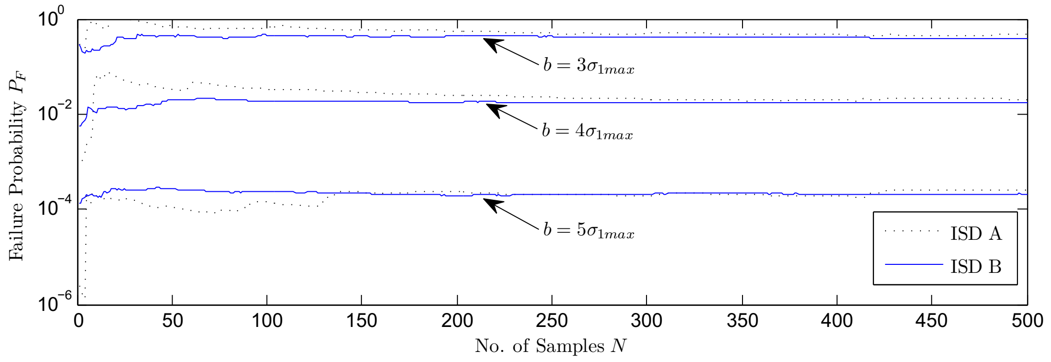


Figure 4.15.: Failure probability estimation history with importance sampling for the linear shear building - case 1.

Figure 4.15 also shows that the estimator with ISD A needs more samples for smaller failure probabilities, because the estimator variance increases with smaller failure probabilities. The failure probability graph for the linear shear building - case 1 - is illustrated in Figure 4.16 and shows that all sampling methods are unbiased.

In order to estimate the failure probability graph for the linear shear building (case 1) with importance sampling, 15 threshold values between 0.01 and 0.024 in steps of 0.001 have been considered. The failure probability graph thus shows the linear interpolation of those estimated probability values. Figure 4.17 shows the results of the failure probability estimations for the linear shear building with filtered input excitation considered in case 2.

All sampling methods yield very similar results. The failure probability graphs are almost identical and can hardly be distinguished in the figure. Figure 4.18 shows the corresponding failure probability graph for the non-linear shear building example. By comparing Figures 4.17 and 4.18 one can see that maximum response values are higher for the the Clough-Penzien filtered input (case 2) in comparison to the responses ob-

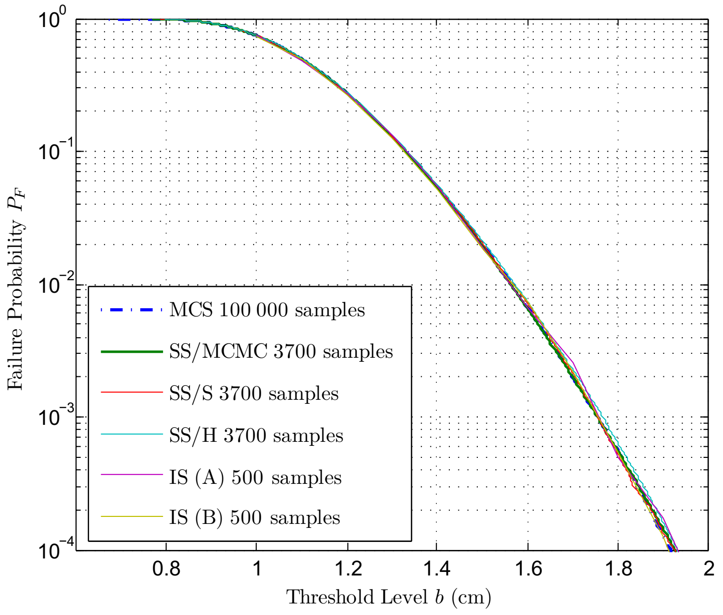


Figure 4.16.: Failure probability estimation for the linear shear building - case 1.

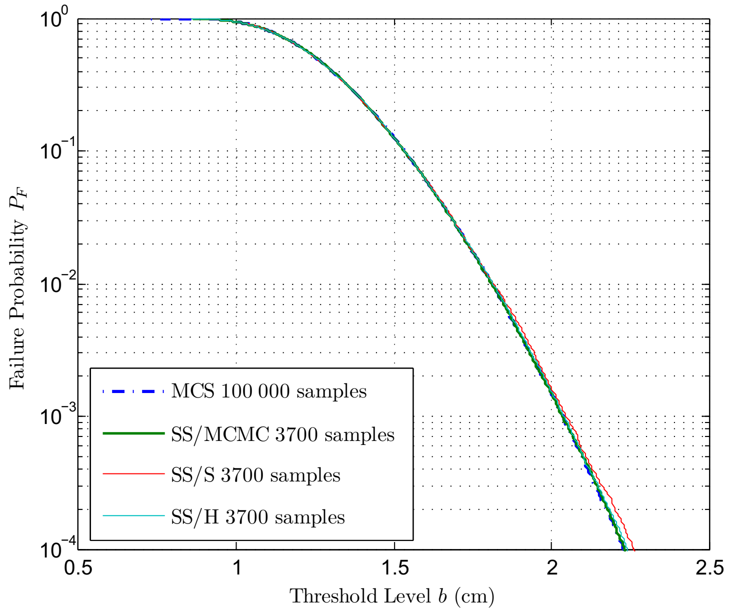


Figure 4.17.: Failure probability estimation for the linear shear building - case 2.

tained with the modulated Gaussian white noise input, although the amplitudes of the unfiltered input are larger. In sum, several examples have shown that all simulation methods deliver the same results for small probabilities as direct Monte Carlo simulation, which supports the statement that all simulation methods are unbiased. The differences between the methods, however, can be seen by investigating the efficiency of the estimators, that is, their variance values.

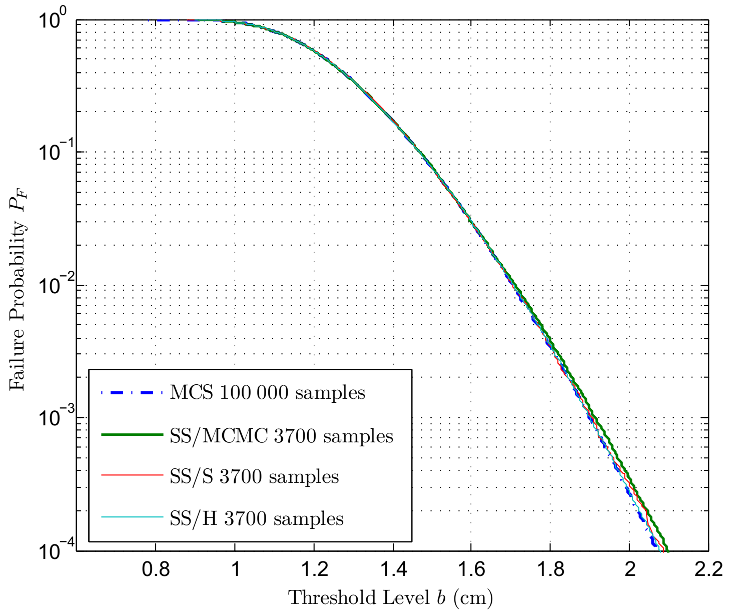


Figure 4.18.: Failure probability estimation for the non-linear shear building.

4.3.2. Variance of the estimators

In the following the c.o.v. values of the estimated failure probabilities are investigated. As already mentioned the c.o.v. values have been calculated with the results of 50 independent simulation runs. The following Figure 4.19 shows the c.o.v. values corresponding to the failure probability estimation for the SDOF oscillator shown in Figure 4.13.

Figure 4.19 demonstrates that SS/S has the largest and SS/H has the lowest c.o.v. values among the subset simulation methods for the SDOF oscillator example. The results of the importance sampling are not comparable with the ones from the subset methods. However, it illustrates their efficiency and shows that importance sampling with ISD B has almost the opposite behavior of MCS, because it has a comparatively bad performance to estimate high failure probabilities and gets more and more efficient the smaller the probability is. This illustrates the 'inverse' behavior of ISD B which has been discussed in subsection 2.3.3.3. It also shows that the estimator with ISD B clearly outperforms the one with ISD A with respect to the estimation c.o.v..

Figure 4.21 shows the c.o.v. value of both importance sampling estimator during the sampling process. Again, the estimator with ISD B is more efficient. One can also observe that the estimator c.o.v. for ISD A increases noticeably with smaller failure probability values. In contrast, the estimator c.o.v. for ISD B gets smaller for smaller probabilities. This can be also be seen in Figure 4.22, which shows the c.o.v. of the estimation for linear shear building (case 1) during the importance sampling procedure.

Furthermore, Figure 4.21 shows that the c.o.v. values for ISD A have increased with the complexity of the problem (compare Fig. 4.20), while the estimator c.o.v. for ISD

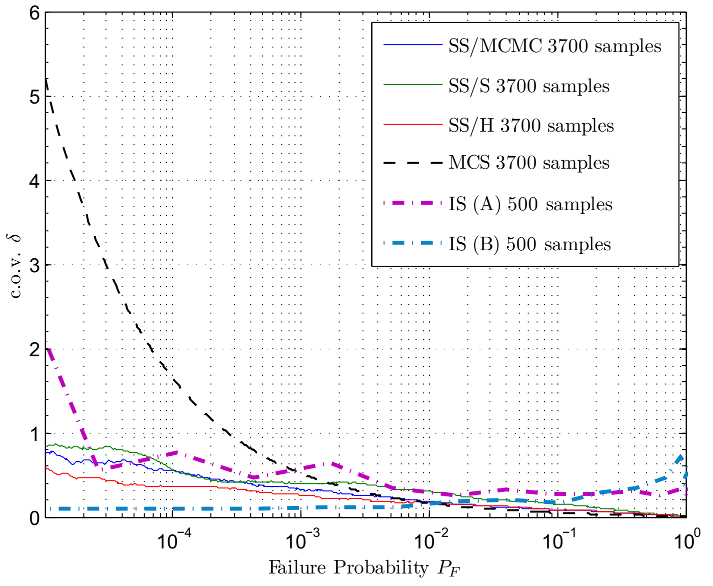


Figure 4.19.: Coefficient of variation for the SDOF example.

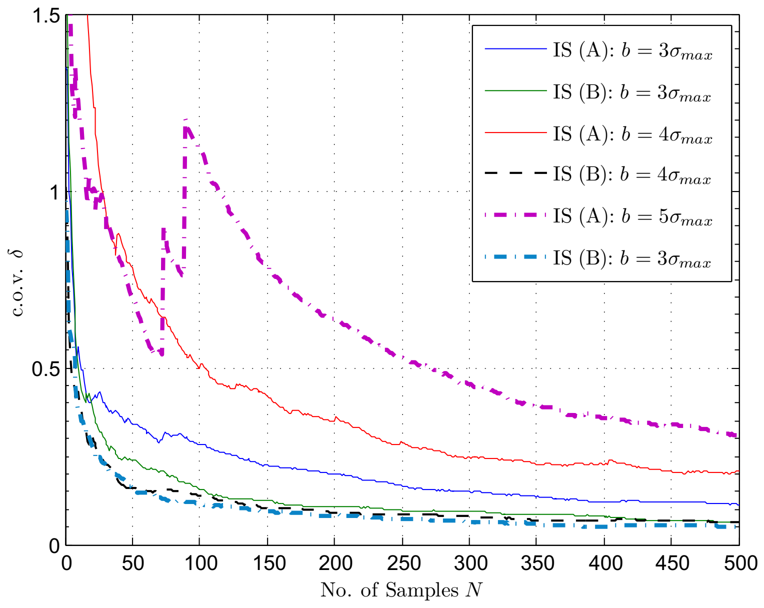


Figure 4.20.: Coefficient of variation for the SDOF example.

B is not affected.

Figure 4.22 shows the c.o.v. values for the linear shear building example (case 1) in comparison with the subset methods. One can see that the importance sampling estimator with ISD A roughly yield similar c.o.v. values as the subset simulation methods, which has used more than seven times more samples (see also Figure 4.19).

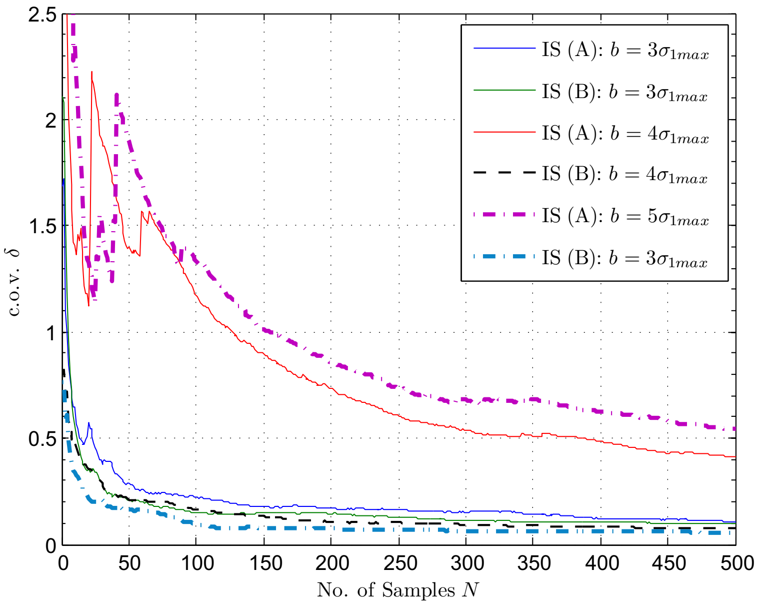


Figure 4.21.: c.o.v. value during importance sampling for the SDOF example.

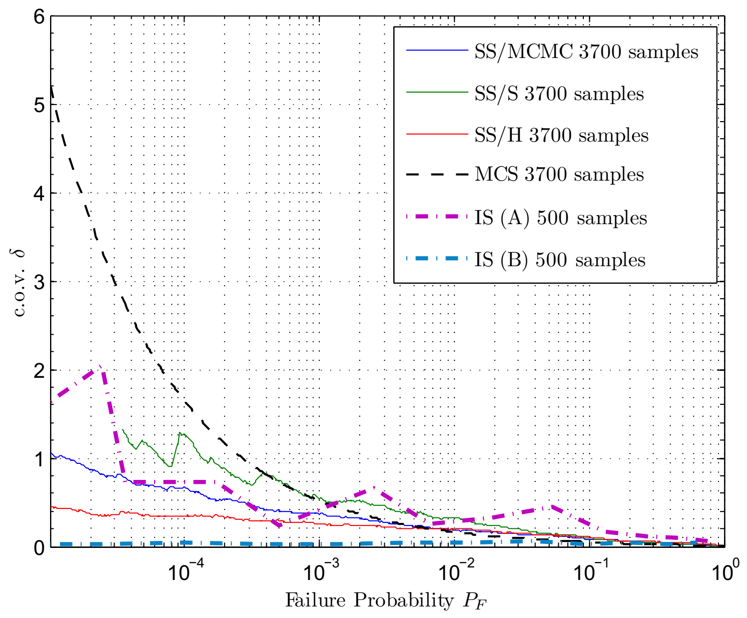


Figure 4.22.: c.o.v. values during importance sampling for the linear shear building example - case 1.

The c.o.v. values for the filtered excitation (case 2) are very similar and are shown in Figure 4.23.

Again, it can be seen that SS/S has the worst performance followed by SS/MCMC. The hybrid method combines the advantages of both methods and usually reaches better

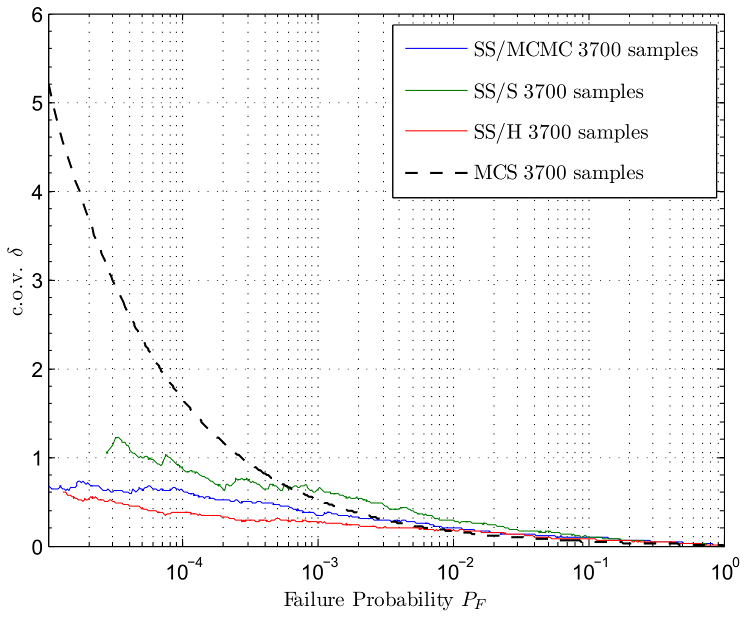


Figure 4.23.: Coefficient of variation for the linear shear building example - case 2.

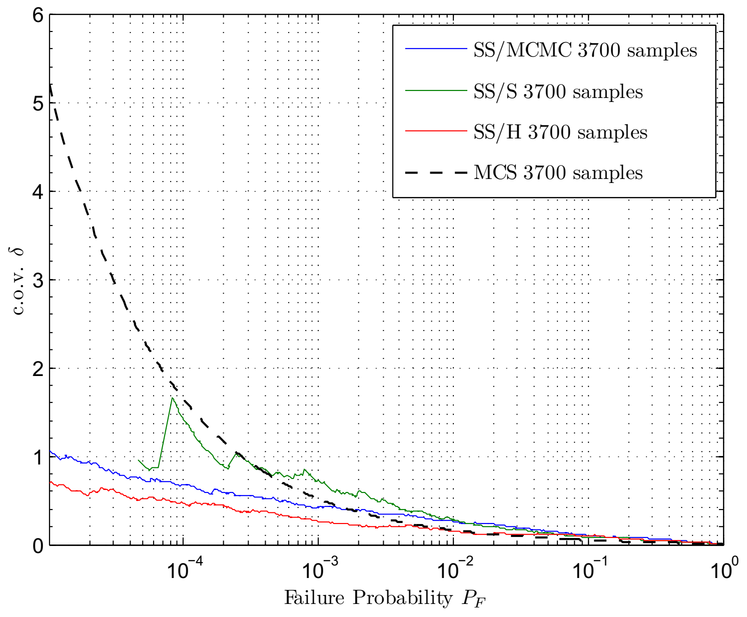


Figure 4.24.: Coefficient of variation for the non-linear shear building example.

c.o.v. values. This is also true for the estimation results of the non-linear structure which are shown in Figure 4.24. It shows similar results for SS/MCMC and SS/H, but with an even worse performance of the splitting method. This, however, may not only be caused by the sampling method, but also by special properties of the example. This will be further discussed later on (subsection 4.4.3).

In sum, it has been shown that the importance sampling methods clearly outperform the subset simulation methods due to the usage of explicit knowledge about the failure region. The hybrid method usually delivers estimations with the lowest c.o.v. value and seems to be the best choice in order to minimize the estimation c.o.v.. However, the simulation methods also have different computational efficiency which needs to be accounted in order to compare the simulation methods with respect to their overall efficiency. This is further investigated in subsection 4.5.1.

4.3.3. Modification of Samples

A property that almost all described sampling methods have in common is that they 'modify' a given sample to increase the probability that its corresponding response provokes a failure state. In the following this process is shown for the high dimensional samples used in the SDOF oscillator example in order to illustrate the dependencies between these samples.

For SS/MCMC the modified version of the Metropolis-Hastings algorithm should be used for high dimensional samples. Then, each dimension is separately modified using the proposal density. For the applied adaptive uniform proposal function the changes in each dimension are only in a small vicinity which is defined by the proposal function support region. Figure 4.25 shows the last part of three samples from different simulation stages. Thus, the samples are in a sequential mother-child relationship. The first sample (stage 1) is generated by MCS and the second and the third sample are, respectively, realizations of the generated Markov chains separately for each dimension. It can be seen that the dependencies of the samples are clearly visible, especially for the corresponding responses.

Figure 4.25 has illustrated that SS/MCMC introduces dependencies between sample values of consecutive states over the whole duration of study, but separately for each dimension. Note, that a part of these sample values directly take the same value as its mother sample due to the rejection step in the Metropolis-algorithm, which introduces even stronger dependencies.

In the SS/S method, however, the first part before the first passage point is copied and the second part of the excitation is freely simulated by direct MCS. Thus, the dependencies only occur by their extremes as fully dependent and completely independent. Figure 4.26 shows the last part of three consecutive simulated samples from different simulation stages generated with SS/S. It can be seen that all three samples are equal until the first intermediate passage point at time step 1243 and in turn, the third sample is equal until the second intermediate passage point at time step 1268.

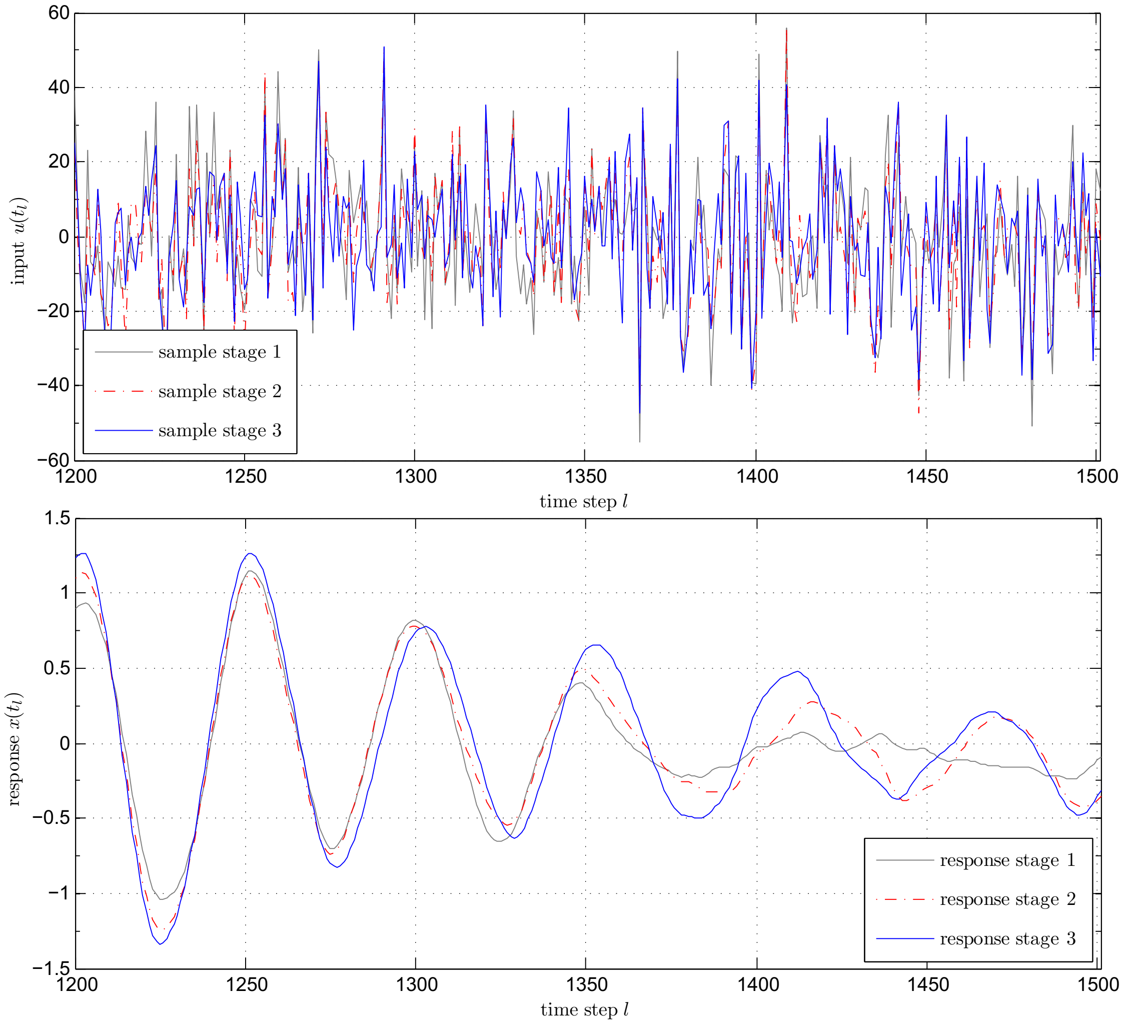


Figure 4.25.: Samples and responses generated by SS/MCMC.

Obviously, the combination of both methods, which is used in SS/H creates less dependencies between consecutive samples, because the first part until the FPP is only dependent by the Markov chains (instead of copying) and the second part is independent due to direct MCS (instead of the Markov chain dependency). This finally leads to a more efficient estimation in the sense of a lower c.o.v. value.

In the importance sampling scheme which uses the conditional PDF $p\left(\mathbf{z}|F_{il}\right)$ as importance sampling density (ISD B) a similar behavior of sample modification can be observed during the sample generation process in which a q-dimensional sample, conditional on the elementary failure event F_{il}, is generated using an unconditional q-dimensional Gaussian sample. This transformation process facilitates efficient simulation of conditional samples and has already been illustrated for the two-dimensional case in Figure 2.2. However, Figure 4.27 shows an example for this transformation for the high-dimensional case for the SDOF oscillator example. The figure shows the design point at time step $l = 192$ for a threshold value $b = 1.8$. Note, that the figure shows

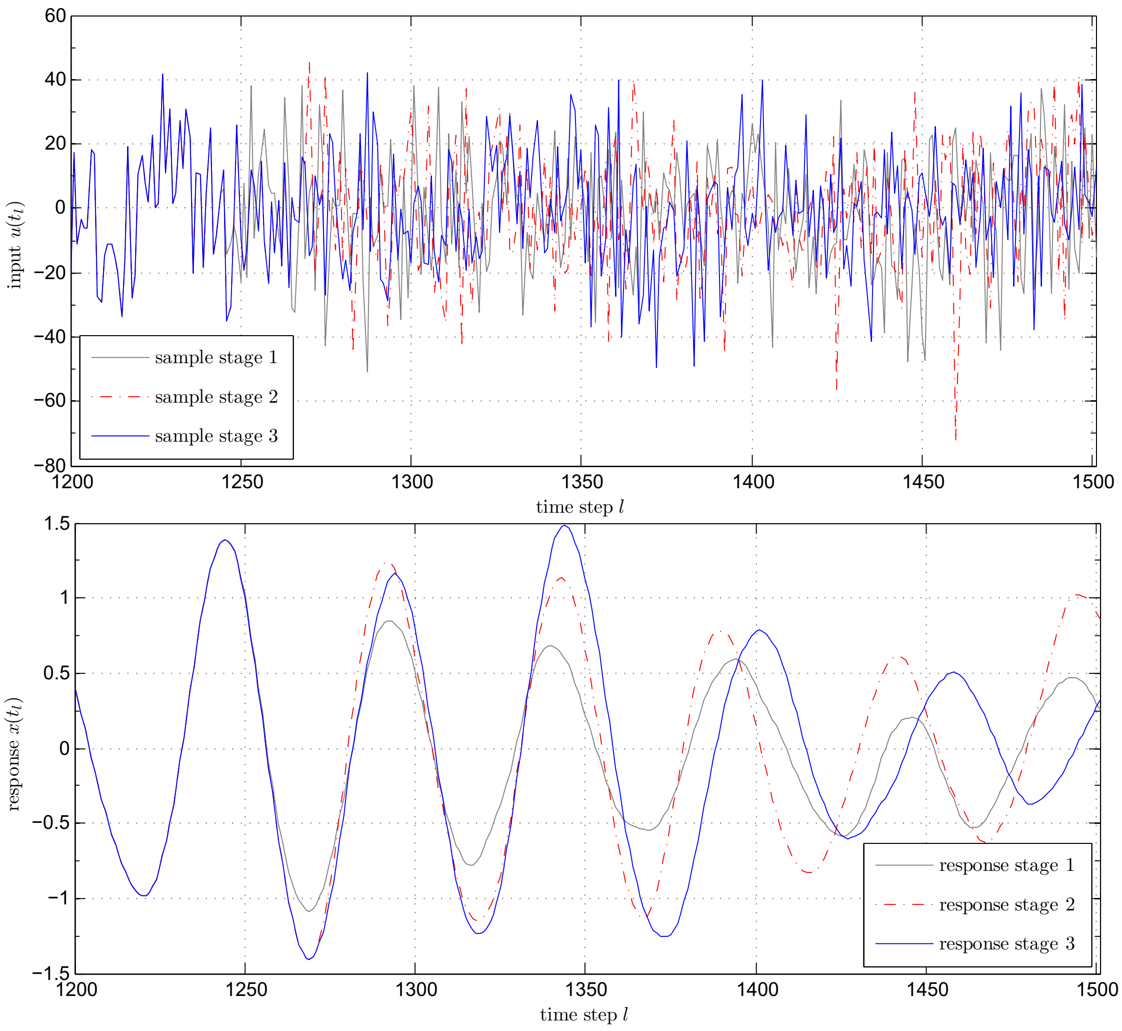


Figure 4.26.: Samples and responses generated by SS/S.

transformed design point in the u-space, because it is originally defined in z-space. According to its definition, the design point pushes the maximum response value at time step $l = 192$ exactly to the threshold level b. As a result of the transformation, the original sample is modified in a way that its maximum response will be at the same time and above threshold b. How much the threshold level will be exceeded depends on the (random) choice of α, which determines how 'far' the samples will be pushed into the failure region.

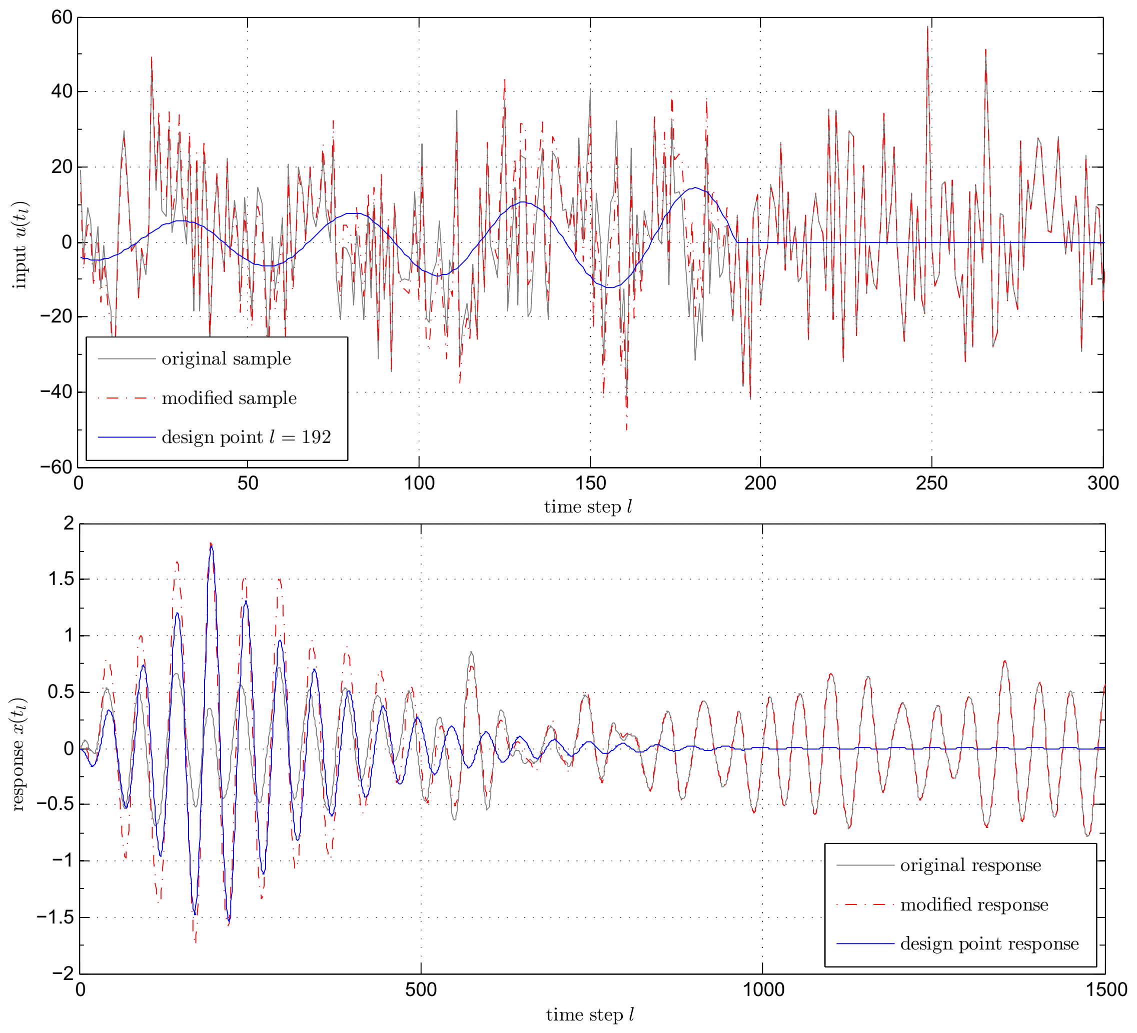


Figure 4.27.: Samples and their corresponding responses in the importance sampling framework with ISD B.

4.4. Properties of the Subset Sampling Methods

This section investigates properties of the subset simulation methods which may influence their estimation or computational efficiency. Moreover, the choices of parameters are investigated in several examples.

4.4.1. Parameter Selection for Subset Simulation

4.4.1.1. Choice of the Intermediate Thresholds

As mentioned before the simplest way to distribute the contribution of the intermediate failure probabilities equally over the sampling stages is to determine all (but the last) intermediate failure probabilities a priori with the same value p_0. The corresponding intermediate threshold levels b_i are then chosen adaptively during the simulation and moreover, all of them can be controlled by a single parameter. In the following, the influence of the parameter p_0 is shown on the SDOF oscillator example applied to SS/S.

Similar to the choice of the proposal function support region size, an appropriate value for p_0 is a trade-off between two contradicting influences. On the one hand, a small value of p_0 leads to better convergence of the algorithm toward the failure region of interest, that is, less sampling stages m may be required and the total number of samples N_T will be reduced. On the other hand, if the conditional probabilities are smaller their accurate estimation will need more samples N per sampling stages, which in turn increases the total number of samples. The choose of p_0 is therefore a trade-off between minimizing the number of samples per stage, while the number of stages increases, and minimizing the number of stages, while the number of necessary samples per stage increases.

However, this value should be chosen in a way that the estimation is efficient, which means that a small c.o.v. value should be favored. In order to investigate a good choice for this parameter, the SDOF example is studied with SS/S under fixed computational effort, that is, a fixed number of total samples N_T. The SS/S method has been selected, because it is free of other parameters as the proposal function in SS/MCMC and SS/H. To define the right proportion of the parameters m and N two scenarios are considered. For given failure probabilities $P_F = 10^{-3}$ and $P_F = 10^{-4}$ the number of necessary subset recursions m can be approximated by

$$m \approx \frac{\log P_F}{\log p_0} \tag{4.35}$$

Then the number of samples per sampling stage N for a fixed number of total samples can be obtained by Equation (2.69). Table 4.2 gives an overview over the considered parameters which have been used in the simulations. Due to the fact that the values N, R, m need to be integers, the actual numbers of N_T vary fractionally.

The results of 100 independent simulation runs are shown in Figures 4.28 and 4.29.

According to Figure 4.28, one can see that the changes of the value p_0 have no noticeable influence on the expectation value of the estimator. All estimations roughly yield the same probability values. However, in Figure 4.29 it can be seen that the c.o.v. values increase exponentially for very small values of $p_0 < 0.1$ and that the c.o.v. for values p_0 larger than 0.1 does not change significantly. This supports the approximate analytical equation for the estimator c.o.v. which has been derived in [1] under the assumption of independent first passage points. In sum, it can be seen that $p_0 = 0.1$ is a reasonable choice to achieve a good convergence to the target failure probability level while the estimator c.o.v. is sufficiently small.

Also note, that also in view of the parallelization of the subset simulation procedure a small value for m and a large value for N is favorable, because N effectively defines the limit for parallel computation during the simulation process. Contrarily, m defines the number of synchronization steps during the different stages. Hence, in a parallel

	$P_F = 10^{-3}$			$P_F = 10^{-4}$		
p_0	m	N	N_T	m	N	N_T
0.01	2	1407	2799	2	1859	3699
0.04	2	1428	2798	3	1267	3699
0.08	3	986	2800	4	984	3699
0.1	3	1000	2800	4	1000	3700
0.2	4	823	2797	6	740	3700
0.3	6	622	2797	8	628	3701
0.4	8	538	2800	10	579	3702
0.5	10	509	2795	13	529	3697
0.6	14	451	2792	18	475	3705
0.7	19	438	2796	26	435	3712
0.8	31	400	2770	41	411	3691
0.9	66	373	2778	87	390	3658

Table 4.2.: Number of samples used for the c.o.v. evaluation for different values of p_0

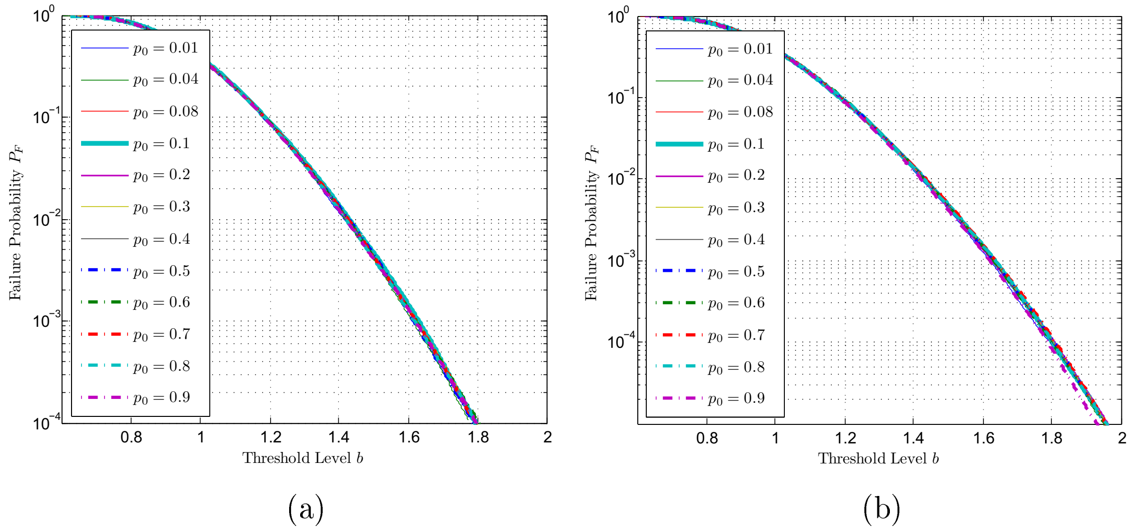
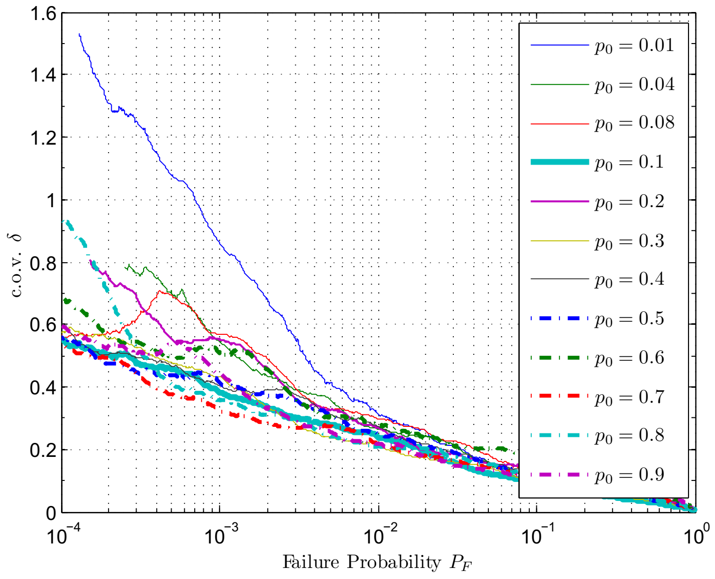


Figure 4.28.: Results of the failure probability estimation for different values p_0. with target failure probability (a) $P_F = 10^{-3}$ and (b) $P_F = 10^{-4}$.

computation framework, a value of $p_0 = 0.1$ is a good choice to maximize the number of possible parallel calculations and keep the estimator c.o.v. low at the same time. Therefore, for most of the experiments the value $p_0 = 0.1$ has been used.

4.4.1.2. Proposal Function for MCMC simulation

As already mentioned in subsection 2.4.3 the choice of the proposal function and their support region influences the efficiency of the MCMC algorithm. Too small or too broad support regions may affect the ergodicy of the MCMC simulation and may lead to a



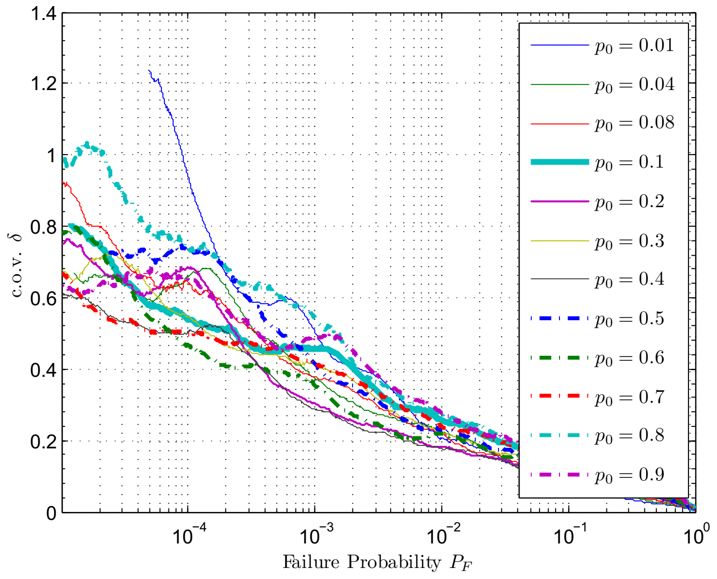


Figure 4.29.: Different values for p_0 and the resulting c.o.v. values for the SDOF example with target $P_F = 10^{-3}$ (top) and $P_F = 10^{-4}$ (bottom).

biased failure probability estimator and/or higher c.o.v. values.

If the proposal distribution is aligned at the current sample in the MH-algorithm, the proposal distribution is called adaptive. However it is also possible to choose a non-adaptive proposal PDF, i.e., it is independent of the current sample. In this case the shape of the PDF should be close to $p(x|F_i)$ [4]. This approach is then similar to the choice of the ISD in importance sampling, but this requires knowledge about the failure region, which is not available in the general case. Using an adaptive symmetric proposal distribution, that is, $q(x|x') = q(x'|x)$, the Metropolis-Hastings algorithm reduces to the original Metropolis algorithm. As described in subsection 2.4.2, the proposal function should be used independently for each dimension of x in order to avoid the zero-acceptance phenomenon. Two different proposal functions, that is, a local uniform proposal PDF and a Gaussian PDF have been tested.

Uniform proposal function

The uniform distribution is defined locally over the interval of size $2l_j$ and centered at the mother sample u_j^{i-1}:

$$q_j\left(u_j'|u_j^{i-1}\right) = U\left[u_j^{i-1} - l_j, u_j^{i-1} + l_j\right] = 2l_j \cdot U\left[0,1\right] + u_j^{i-1} - l_j \tag{4.36}$$

where $U\left[\cdot\right]$ is the uniform distribution over the specified interval. The parameter l_j should then be chosen appropriately, e.g. as a fraction of the standard deviation σ of the input PDF $p\left(u\right)$.

Gaussian proposal function

$$q_j\left(u_j'|u_j^{i-1}\right) = \frac{1}{\sqrt{2\pi}\sigma_j} \exp\left[-\frac{1}{2}\frac{(u_j' - u_j^{i-1})^2}{\sigma_j^2}\right] \tag{4.37}$$

Similar to the parameter l_j in the uniform proposal, the parameter σ_j controls the size of the support region of the proposal PDF.

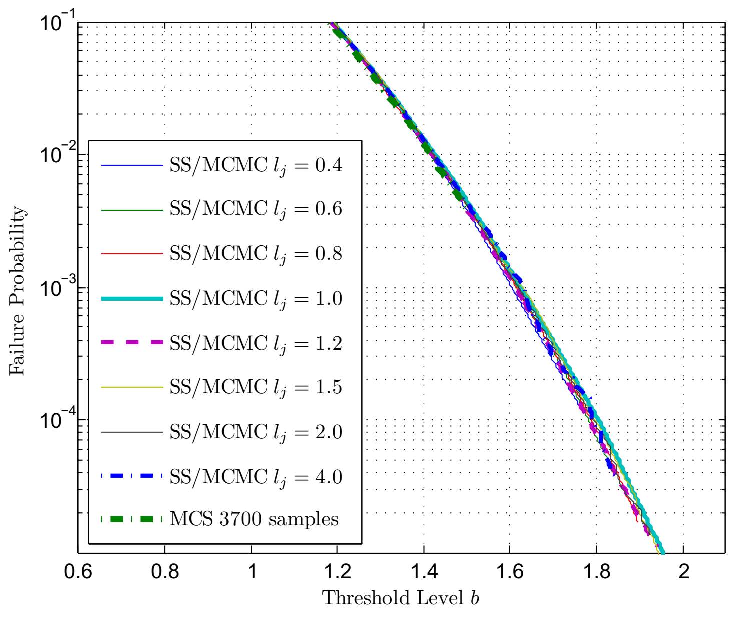
Note, that the input u for the dynamic examples is standard Gaussian stretched by the factor $g = \sqrt{2\pi S_0/\Delta t}$, i.e. $u = g \cdot z$. In this case, the Metropolis algorithm is actually applied to z_j and each new candidate sample z_j' is then back-transformed to u_j'. In this way the choice of the proposal function support by l_j or σ_j is independent of the spectral intensity of the chosen input u.

Figures 4.30 - 4.33 give an overview about the differences of the two proposal functions as well as their support region during the sampling process with SS/MCMC applied to the SDOF example in which $N = 1000$ samples have been used in each of the 4 stages and over 50 independent simulation runs.

Figure 4.30 shows that the values l_j should be between 0.8 and 1.2 and a value of $l_j = 1$ is a reasonable choice for the uniform distribution. While the bias of the estimator is not effected, this parameter has significant influence on the estimator variance. In comparison to the uniform proposal function, Figure 4.31 shows the results for the Gaussian proposal function. It can be seen that the results look very similar to the ones for the uniform distribution, with the difference that the values σ_j with the lowest c.o.v. are smaller. According to the figure a value $\sigma_j = 0.6$ seems to be a good choice.

By comparing the absolute c.o.v. values of both proposal functions in Figure 4.32, it can be seen that the Gaussian proposal function leads approximately to the same results. Consequently, it is natural to select the uniform proposal distribution, because it is simpler and more efficient in the implementation.

In the same experiment, the corresponding average rejection rates have been recorded



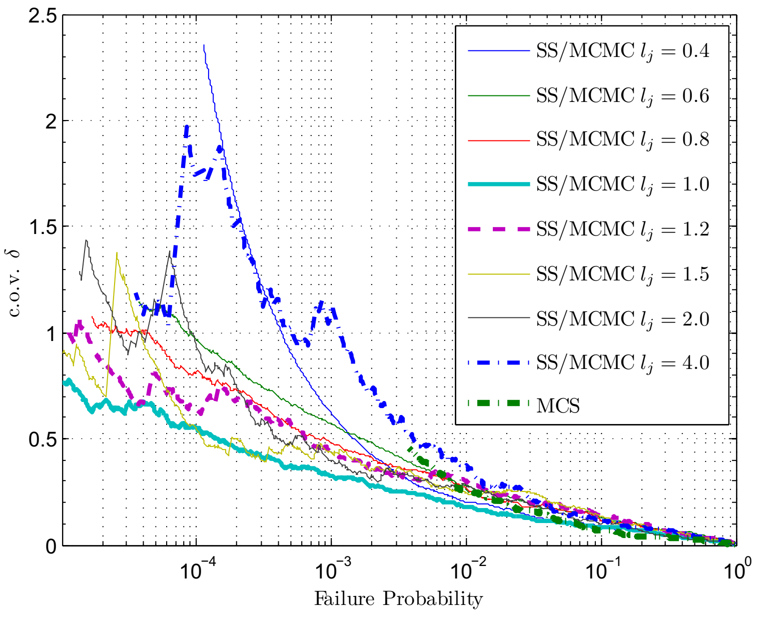
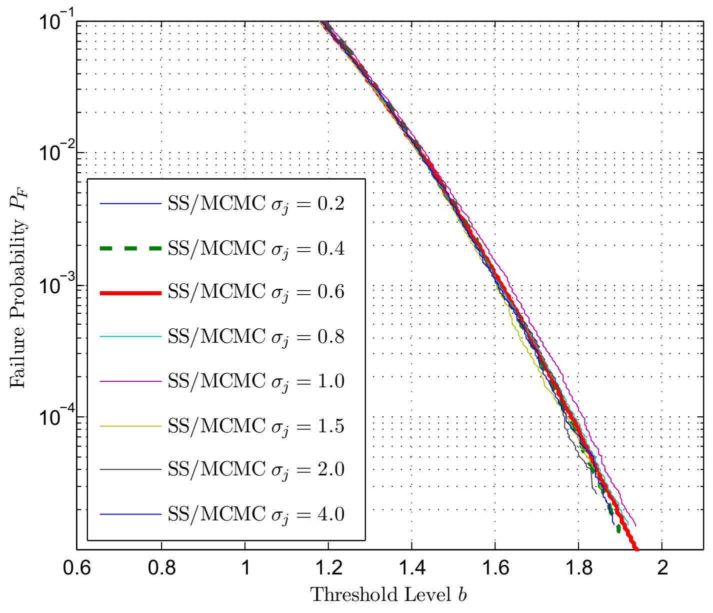


Figure 4.30.: Different values of l_j for the SDOF example with SS/MCMC, 1000 samples, 50 runs. Shown is the failure probability P_F and the estimation c.o.v. δ.

and are shown in Figure 4.33. More precisely, the figure shows the relative frequency of rejected candidate samples generated by MCMC simulation which had the desired distribution of the input, but did not lie in the desired intermediate failure region (the second step in algorithm 3). This rejection rate gives information about the efficiency advantage to generate samples with MCMC rather than with direct MCS, where the



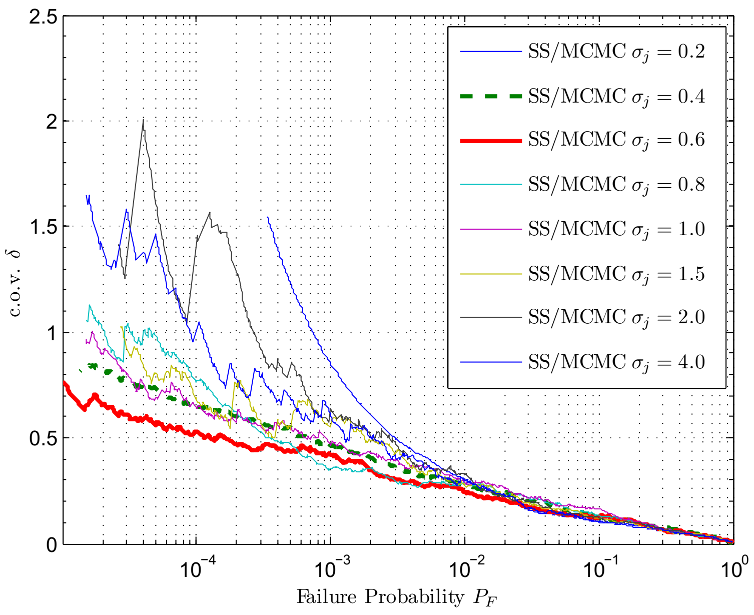


Figure 4.31.: Different values of σ_j for the SDOF example with SS/MCMC, 1000 samples, 50 runs. Shown is the failure probability P_F and the estimation c.o.v. δ.

expected rejection rate to generate conditional samples would be $1 - 1/P(F_i)$. It can be seen in the figure that the rejection rate is much better, but it is also unsatisfying to see that more than half of the generated samples are rejected together with their corresponding expensive system analysis. This is the contradiction which determines the best size of the support region of the proposal function, because every rejected

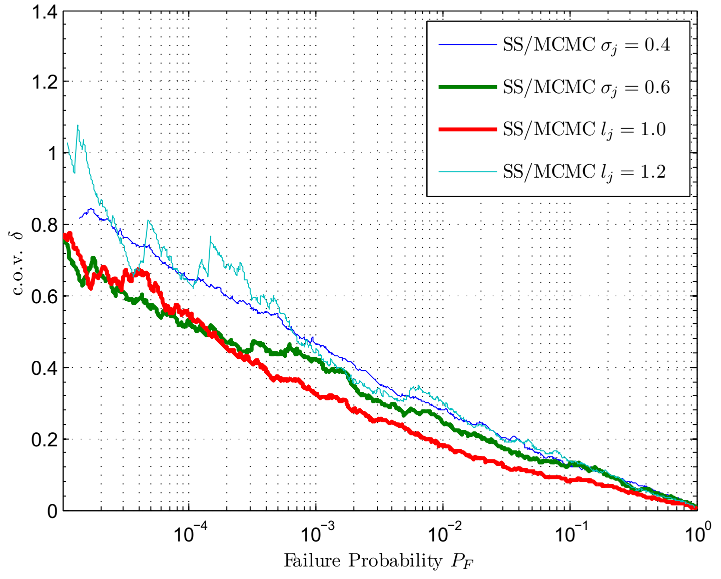


Figure 4.32.: Best c.o.v. results for the SDOF example of both proposal functions in comparison.

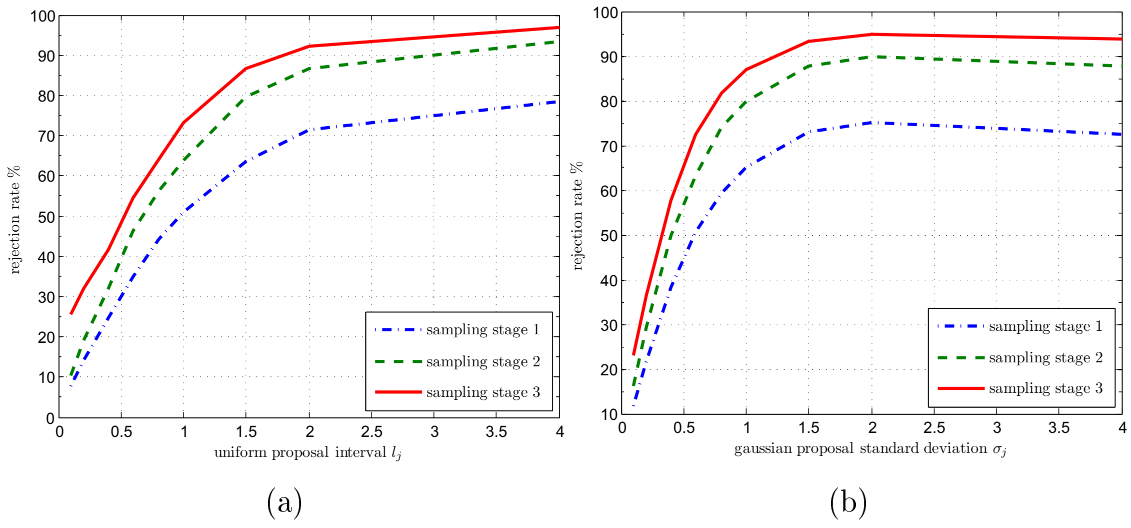


Figure 4.33.: Rejection rates for each sampling stage corresponding to (a) Figure 4.30 and (b) Figure 4.31.

sample introduces strong dependencies in the Markov chain due to the repetition of samples which reduces the efficiency of the simulation. On the other hand, if the rejection rate is reduced by a smaller support region, consecutive samples will lie closer to each other and will slow down the exploration of the failure region, which in turn also leads to an efficiency reduction.

In order to investigate the influence of the proposal function to SS/H and to compare the results with SS/MCMC the same experiments have been done with SS/H. Figure 4.34 shows robust estimation for both proposal functions except of a small bias for $\sigma_j = 4.0$ with the Gaussian proposal PDF.

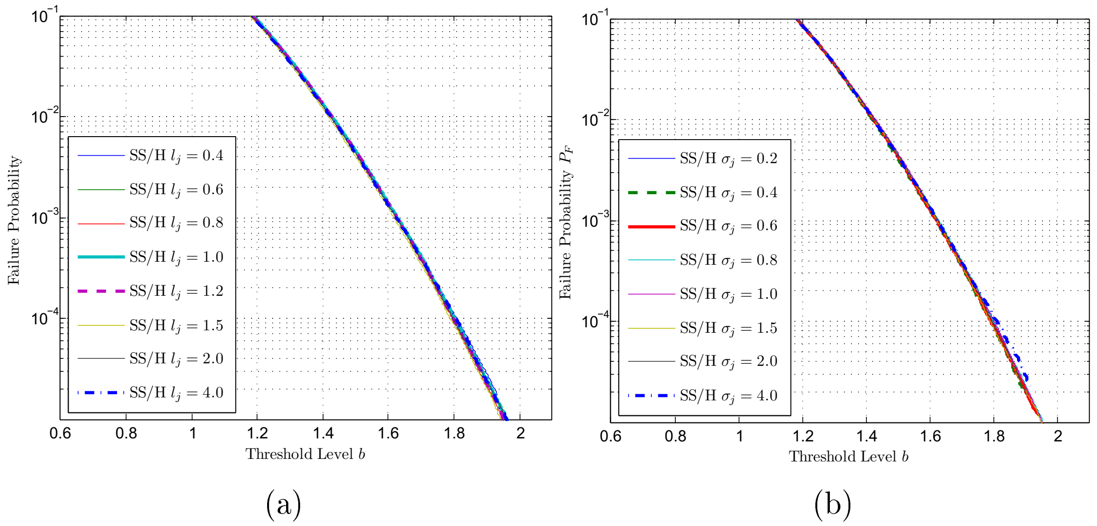
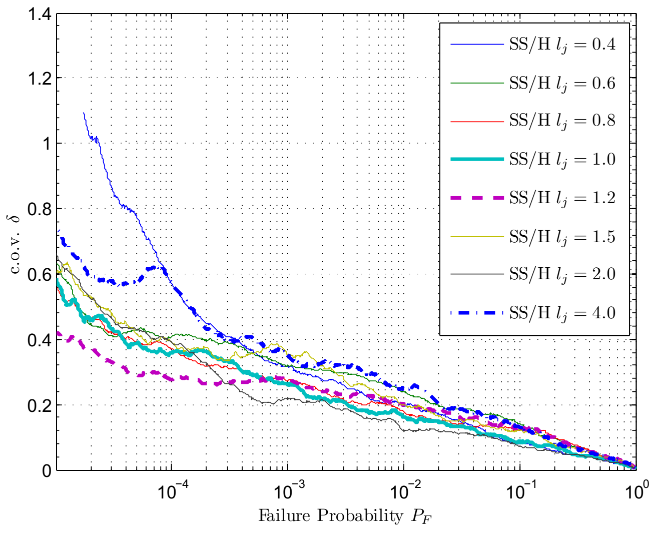


Figure 4.34.: Failure Probability for different values of (a) l_j (b) σ_j for the SDOF example with SS/H.

More interesting are the c.o.v. values of the estimation which are shown in Figure 4.35. It can be seen that the c.o.v. values vary less compared to values which have been obtained with SS/MCMC. This supports the assumption that SS/H is more robust to the choice of the proposal function due to the combination with the splitting method, which makes the method less dependent of MCMC simulation to explore the failure region. One can also see that the both proposal function again lead roughly to the same estimation variances and the shape of the proposal function therefore seems to be much less important than its support region.

The rejection rates of SS/H for both proposal functions are similar to the ones of SS/MCMC shown in Figure 4.33 since both methods work similar until the rejection step.



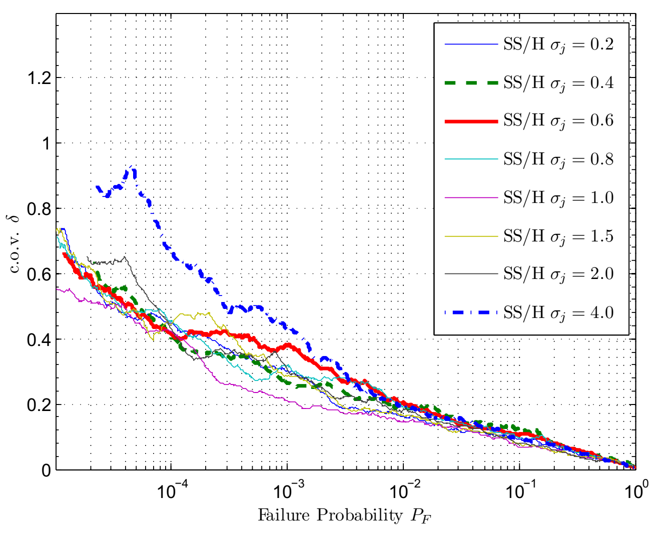


Figure 4.35.: Estimator c.o.v. for different values of (a) l_j (b) σ_j for the SDOF example with SS/H.

4.4.2. First Passage Point Times

In view of the subset simulation methods SS/S and SS/H, the first passage point time gives information about where direct Monte Carlo simulation starts and where the copy process or the MCMC simulation ends. This in turn influences both, the computational efficiency and c.o.v. of the estimator. Therefore, this subsection gives an overview of the average time for the first excursion of the system response over the intermediate limit values. This issue is highly related to the sampling problem and is thus investigated for all three dynamic system examples. For the examples the same parameters have been used ($p_0 = 0.1$, $N = 1000$, 50 runs) except of the number of sampling stages which has been set to $m = 6$. In order to illustrate the problem dependency and efficiency differences, the first passage point is compared with a similar SDOF oscillator which only has a damping ratio of 2% (instead of 5%). The lower damping ratio leads to higher system responses during the duration of study, which can be seen by comparing the response standard deviations as shown in Figure 4.36.

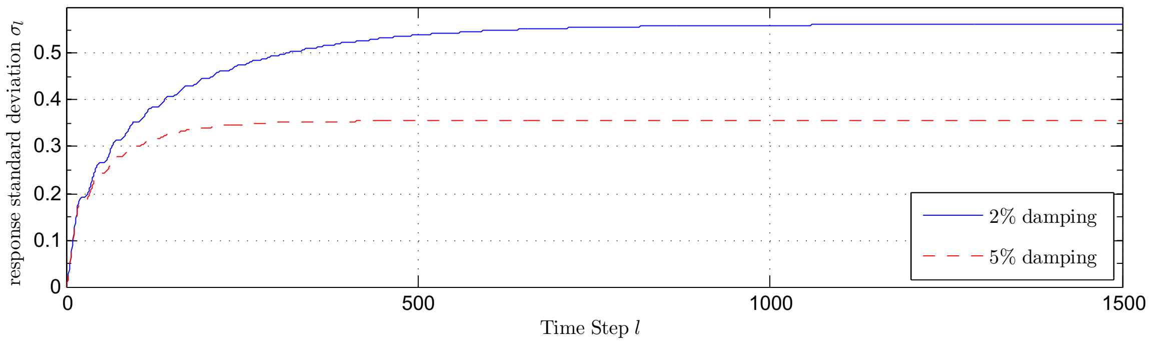


Figure 4.36.: Response standard deviation of the SDOF with 2% and 5% damping.

Figure 4.37 shows the minimum, average and maximum times for the first passage points for the SDOF oscillator with the two different damping ratios using both sampling methods, SS/S and SS/H.

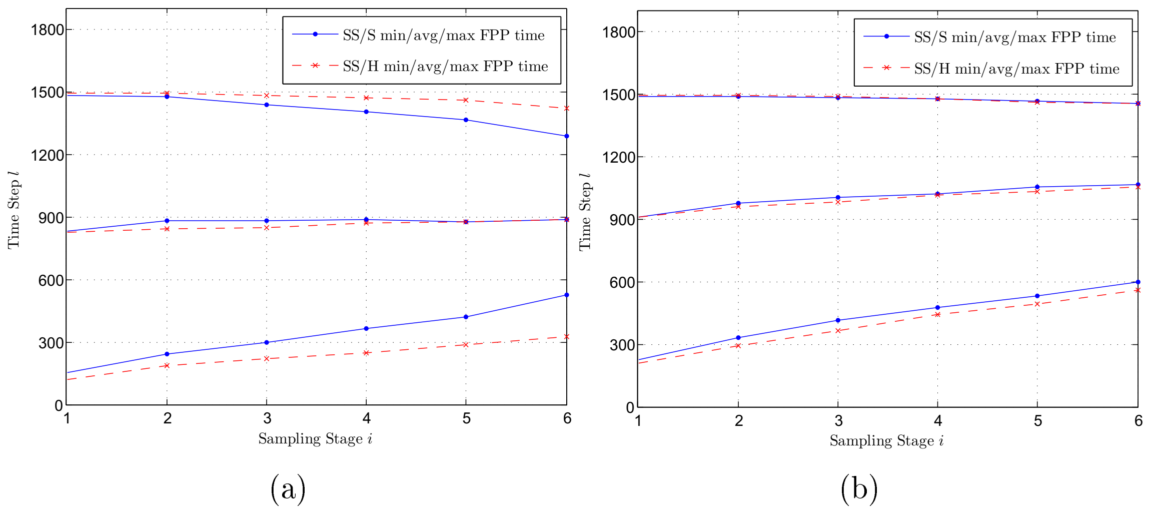


Figure 4.37.: Minimum, maximum and average FPP times for the SDOF oscillator with (a) 5% damping (b) 2% damping using SS/S and SS/H at different sampling stages

Figure 4.37 illustrates, that the first passage point times with SS/H are generally earlier than with SS/S. This is due to the additional use of MCMC simulation in each stage, which leads to a better exploration of the failure region before the first passage point. In comparison, the failure region before the the first passage point time in SS/S is only explored in the first stage by direct MCS.

Figure 4.37 also shows that the reduction of the damping ratio moved the average first point time into the last third of the duration of study. Due to the reduced damping, the system responses sum up over longer time periods and the highest system response is reached later in the duration of study.

The standard deviation of the output over time is therefore strongly related to the first passage point times that will occur during a sampling with the splitting method. This in turn highly influences the computational efficiency of SS/S, since the part of the trajectory and the corresponding response is larger the later the first passage point occurs. The change of the damping ratio from 5% to 2% had comparatively only a little effect on the first passage point time, but leads to a run time difference for SS/S of 12% over 50 independent simulation runs. This number can be expected to increase with the complexity of the problem under study, since the computational effort for the response calculation is little for SDOF. Clearly, the first passage point time has no effect on SS/MCMC and thus the run time difference for the hybrid method is also smaller (2% in this example), because the first part of trajectory is only copied in case of a rejection of the sample candidate generated with MCMC simulation.

Figure 4.38 shows the first passage point times for the linear and non-linear shear building examples. The first passage point times of both examples are mainly ruled by the influence of the envelope function. Therefore, the first passage point times are only distributed in the center part and limit excursions do not occur close to the end of the duration of study.

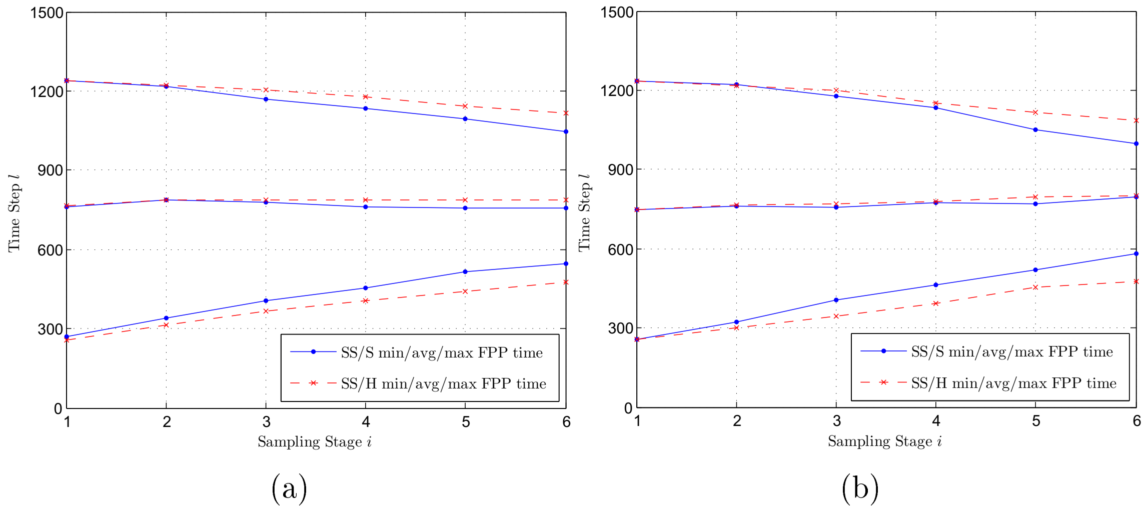


Figure 4.38.: Minimum, maximum and average first passage point times for the (a) linear shear building - case 2 (b) non-linear shear building

All diagrams have shown, that SS/H leads to a better distribution of the first passage point times over the duration of study. Therefore, the combination of SS/S with SS/MCMC leads to a better exploration of the failure region, because the minimum FPP times are always found earlier in comparison with SS/S.

The examples have shown, that the structure characteristics influence the performance simulation methods SS/S and SS/H. Especially for SS/S the computational effort can change drastically for systems with different first passage point characteristics.

4.4.3. Limits of the Splitting Method

As shown in subsection 4.4.1.2 the choice of the proposal function does not affect the bias, but the c.o.v. of the subset simulation estimator using MCMC. It has been shown that proposal function support needs to be chosen appropriately in order to obtain estimations with small c.o.v. values. With the invention of the splitting method to generate conditional samples this problem can be avoided for causal dynamic systems, because SS/S does not have any additional parameters. However, the way in which the conditional samples are generated in the SS/S method introduces problems with respect to the robustness of the method and actually reduces its generality, because in some way assumptions about the shape response function over time are introduced by this method. In subsection 4.3.3 the differences of the conditional sample generation process for SS/MCMC and SS/S have been compared in Figures 4.25 and 4.26. The MCMC method explores the failure region over the whole time space in every sampling stage. In contrast, SS/S explores the failure region only after the first passage point. Over consecutive sampling stages, SS/S always needs to find new first passage points with higher response values in later time steps. Consequently, SS/S can explore a particular elementary failure region only once within a chain of consecutive samples with a mother-child relationship. A single elementary failure region can hence only be explored by independent samples which are not ancestors and a better exploration can only be achieved by using a larger number of samples.

This property indirectly introduces the assumption that the failure region is somehow distributed over the duration of study, which means that SS/S needs some 'time space' to efficiently explore the failure region over several sampling stages. In the case, that the failure region is concentrated within a small time interval, the SS/S method will have performance problems in comparison with SS/MCMC and poor ability to explore the failure region may lead to a biased estimation.

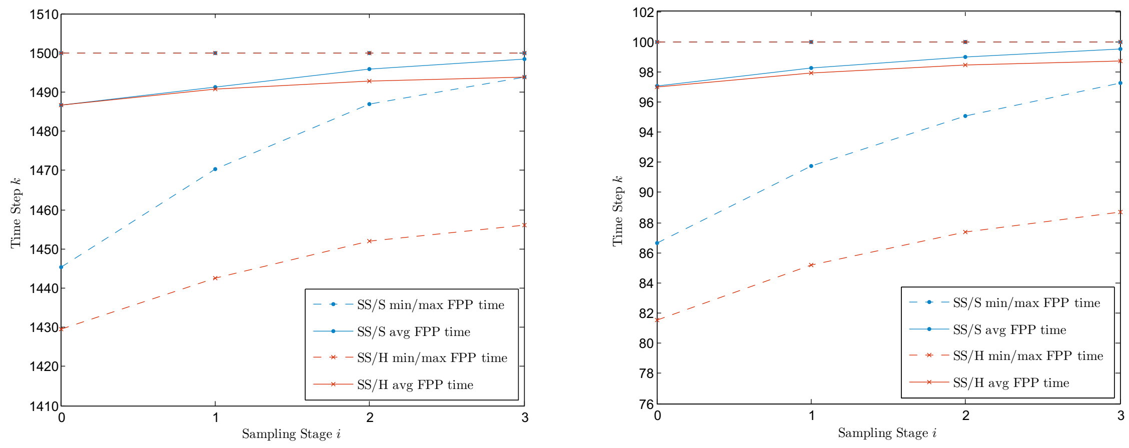
To illustrate this problem an artificial dynamic sampling problem is constructed in which the average first passage point times are concentrated over a small time interval. For example, by using a strictly monotonic increasing response function $r(\cdot)$ the average first passage point times are moved close to the end of the time series. This is accomplished, by simply summing up the absolute values of the Gaussian white noise input function. Using the iterative description for causal dynamic systems (Equation 2.4) the response function is then defined by

$$t_{l+1} = h\big(x(t_l), u(t_l), t_l\big) = x(t_l) + |u(t_l)| \tag{4.38}$$

Two different different lengths of input excitations $u(t)$ are investigated in the following, a long term study: $T = 30\text{s}$, $n_t = 1501$ and a short term study: $T = 2\text{s}$, $n_t = 101$ with a system input $u(t)$ similarly defined as in section 4.2 with $\Delta t = 0.02$ s. The spectral intensity is chosen to be $S_0 = \Delta t/2\pi$, which means that simply the realizations of a standard Gaussian are summarized. For the experiments with SS/MCMC, SS/S, SS/H $N = 2000$ samples have been used to estimate each partial failure probability. Together with a fixed total number of $m = 5$ stages, the total number of samples is $N_T = 9200$.

The results are compared with MCS using 1 000 000 samples.

The following Figure 4.39 shows minimum, average and maximum FPP times for SS/S compared to the ones obtained with SS/H. It can be seen that the first passage point times are on average within the last 5% of the duration of study for both excitation lengths, which means that the failure region is concentrated in a small period of time.



(a) (b)

Figure 4.39.: Average, minimum and maximum FPP times for the intermediate thresholds during SS/S and SS/H estimation for the (a) long term study (b) short term study.

The curve for the minimum FPP time again shows that the hybrid method finds samples with earlier FPPs, because SS/MCMC is also able to find higher response values in the same or at earlier time points over consecutive simulation stages. Due to the concentration of the failure region in time, the SS/S estimator cannot properly explore it, which finally leads to a bias of the SS/S estimator. The bias is shown in the graphs of Figure 4.40, which compares the average failure probability estimates of 50 independent runs of SS/S with MCS, SS/MCMC and SS/H. While the estimations from SS/MCMC, SS/H and MCS show almost identical results, the failure probability graph for SS/S shows a bias which increases as the failure probability decreases.

Also, the failure probability estimation of SS/S for the shorter duration of study shows that the bias is larger and starts with larger failure probabilities compared to the long excitation, because the failure region is even more concentrated in fewer time steps. Although, the effect is comparably little with respect to the length difference of the two different excitations. Due to the fact that SS/S lacks to efficiently explore single elementary failure regions, it is not able to generate new offspring samples with responses that exceed the maximum response value from the previous sampling stage. Consequently, a large number of samples generated in the next sampling stage have the same maximum response value. The estimation of the corresponding conditional failure probability is then based on a set of samples that contains a large number of samples

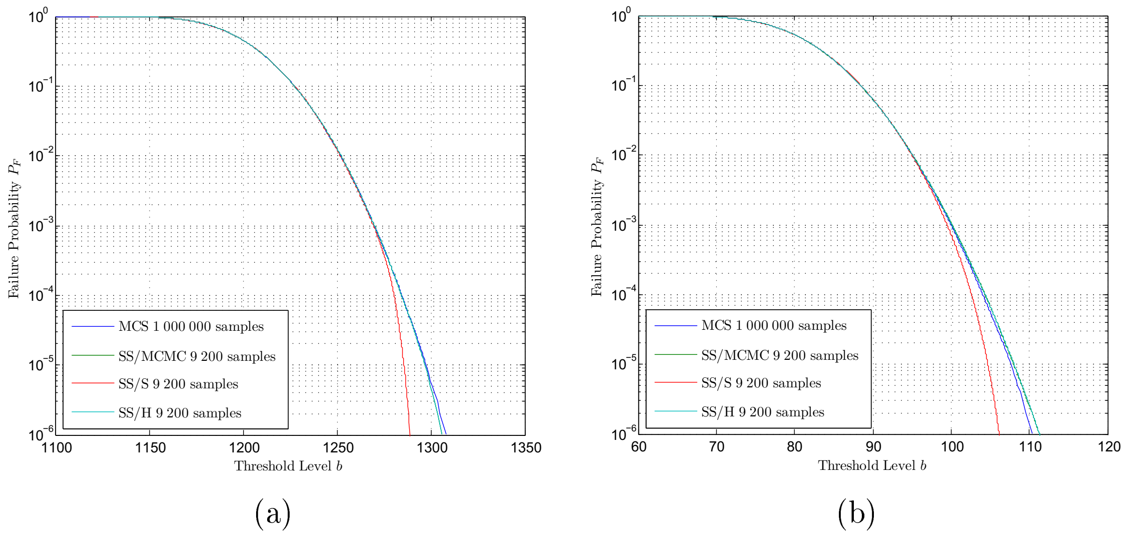


Figure 4.40.: Average failure probabilities showing the biased SS/S estimator compared with SS/MCMC, SS/H and MCS for the (a) long term study (b) short term study.

with equal maximum response values. This finally results in a biased estimation of the conditional failure probability.

The subset simulation framework amplifies this effect extremely, because most of the samples with an equivalent maximum response will have the highest response values among all generated responses. These samples in turn are then selected as mother samples for the next stage. As a result, the number of samples with an equivalent maximum response value grows exponentially over the sampling stages, which leads to stronger biases of the estimated conditional failure probabilities of higher sampling stages. Figure 4.41 shows the stepwise failure probability graph of the SS/S method compared with the other simulation methods. The steps in the failure probability graph are due to large number of samples with similar maximum response values.

This example has shown, that the SS/S method may be biased if the failure region is concentrated in a short time period. This may not be a common scenario in structural engineering practice, but the same problem also occurs in a different scenario which has more practical relevance. For this purpose the non-linear shear building example is considered for further investigation. The response standard deviation over time is very broad and flat as the maximum response values are quickly approached in the first part of the duration of study. The shape of the response standard standard deviation is mainly ruled by the envelope function and the failure region of this example almost has the opposite properties compared to the last example, because the failure region is as broad as the envelope function is equal to one. The failure region is hence not concentrated in time and one should expect the SS/S method to work properly.

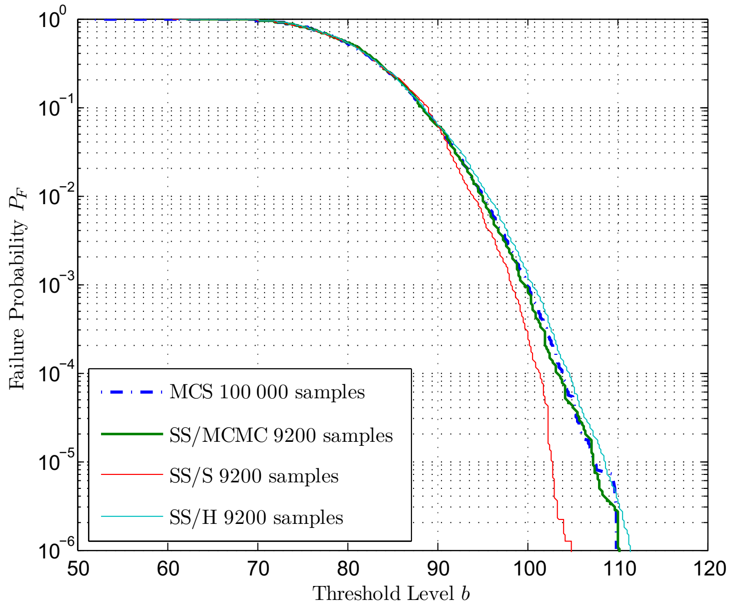


Figure 4.41.: Single estimation run for the short term study.

However, the same problem occurs if the failure probability graph gets steeper, which means that subset simulation procedure reaches small failure probabilities faster. This can be achieved by increasing the stiffness of the structure. Therefore, the non-linear shear building example has been considered with different values of the device stiffness k_d.

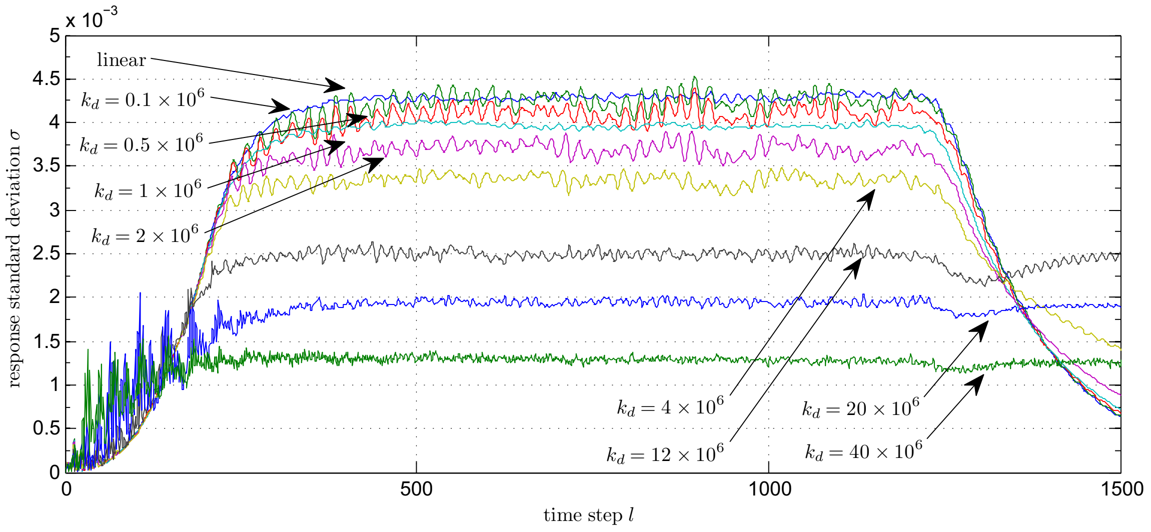
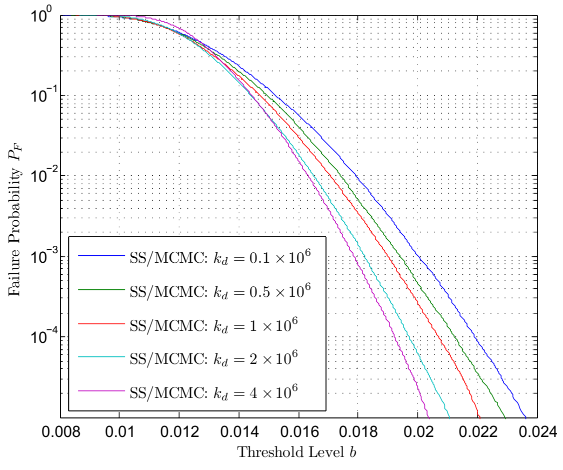


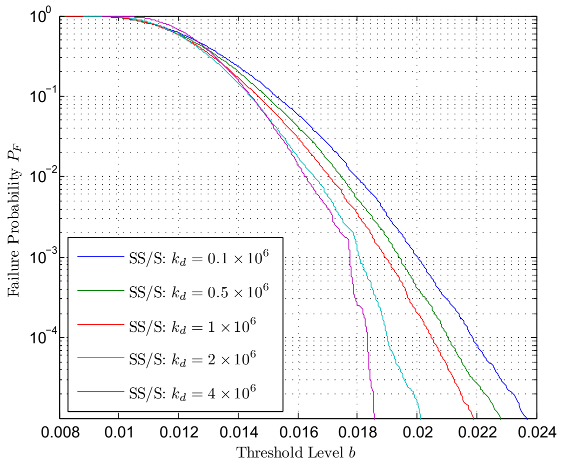
Figure 4.42.: Response standard deviation for the first story of the non-linear shear building.

Figure 4.42 shows estimations of the response standard deviation σ_{1l} for the first story over time. The functions have been estimated with 1000 samples and the standard deviations for the other stories are very similar. Due to the estimation the standard deviation consists of oscillations which would vanish, if more samples are being used. However, the oscillations also illustrate the effect of the change of the parameter k_d, that is, for larger values of k_d the response amplitudes are smaller, but the response frequencies are larger.

The following two Figures 4.43 and 4.44 show the average of 50 independent simulation runs for all three subset simulation methods for different values of k_d. Furthermore, Figure 4.44 (b) shows the results of a single simulation run for all three methods. The same structure and sampling parameters as above have been used for the simulation ($N_T = 3700$ samples).

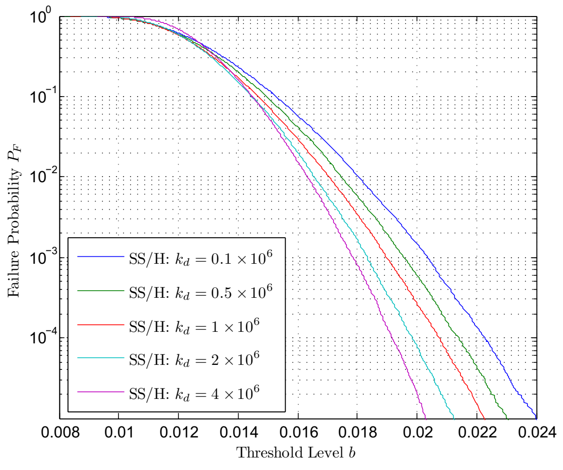


(a)

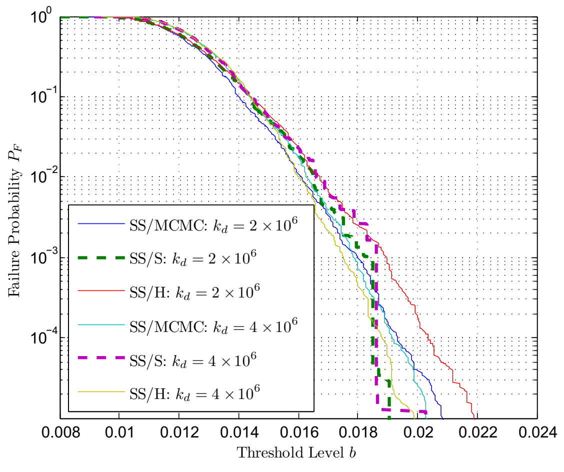


(b)

Figure 4.43.: Different values of k_d: (a) SS/MCMC (b) SS/S.



(a)



(b)

Figure 4.44.: Different values of k_d: (a) SS/H (average) (b) single simulation run.

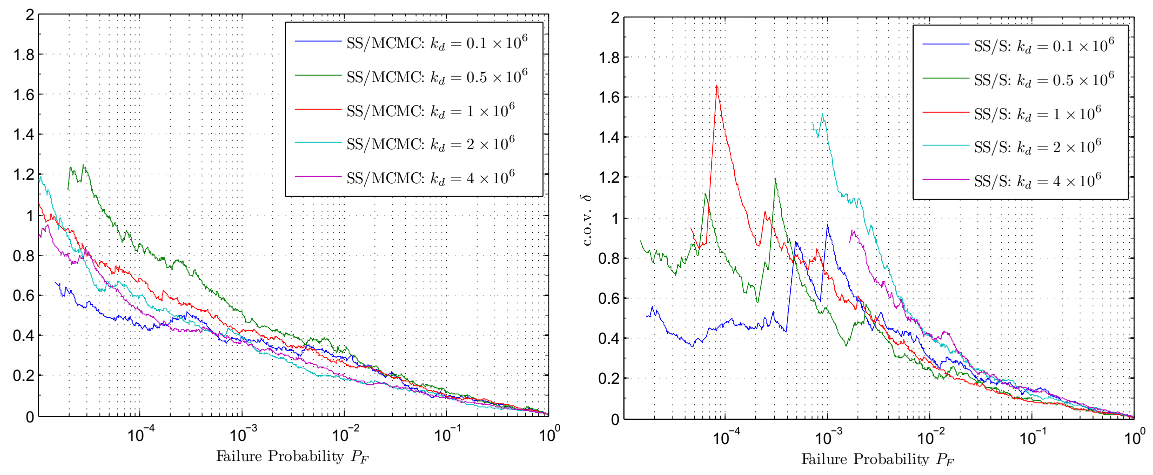
In Figure 4.43 (b) one can observe a similar biased estimation as in the previous example and Figure 4.44 (b) clarifies that the bias is again due to occurrence of many samples with similar response values. This in turn is due to the steepness of the failure probability graph which can also be seen in the figures. For larger values of k_d the failure probability graph gets steeper and the subset simulation method approach the areas of smaller failure probabilities faster. This means that samples with high maximum response values are sampled with higher probability and the probability to generate a sample with a higher response value is very small. Therefore, all subset simulation methods will have difficulties to generate samples with higher response values in the next sampling stage. However, the lack of SS/S to efficiently explore a single elementary failure event increases the problem significantly. Remember, that SS/S always needs to find higher response values at later time steps, which is achieved by direct MCS. This means, if a sample with a high response value is selected as mother sample, SS/S actually needs to start from scratch to find a higher response value in a later time step. In contrast, SS/MCMC can easily increase the maximum response value in a particular time step due to MCMC simulation. Compare also Figures 4.25 and 4.25 in subsection 4.3.3, which show the principles of offspring generation with SS/MCMC and SS/S, respectively. Consequently, SS/MCMC (and so SS/H) has much better abilities to push a sample further into the failure region, in case that the response standard deviation is broad and flat and the corresponding failure probability graph steep.

The two examples have shown that the SS/S method has drawbacks which should be considered, if the method is used to estimate the reliability of structures. Especially, the last example has shown that the problem can occur in engineering practice. It has been shown that the SS/S method is biased if one of the two cases occur. However, it has turned out in experiments that the efficiency of the SS/S method decreases if one of the two cases is approached, that is, the estimator c.o.v. increases. Therefore, the c.o.v. values of the estimators for different values of k_d is studied in the following.

In order to understand the following figures, it is noted that the c.o.v. values for the subset simulation methods cannot be directly calculated, because all subset simulation methods actually sample structure response values. In a single simulation run N_T (likely different) response values are generated. Bringing them into an order, each of the response values can be assigned a probability between 0 and 1 in steps of $1/N_T$. For the next independent simulation run the same number of samples are generated which can be assigned to the same values of failure probabilities, but the response values will be different. Actually, it is likely to end up with $n \times N_T$ different system response values after n independent simulation runs. Thus, the mean and variance of the responses for each of the N_T probability values can be readily calculated, but the mean and variance of the failure probabilities for a given response threshold value cannot be calculated directly. With the definition of particular response values the failure probability mean and variance can be calculated by means of interpolation methods. In order to do this a special case at the borders of the response-range intervals needs to be considered. Each independent simulation run will have a minimum and maximum response value that occurred during the simulation. In the range of the smallest minimum and the largest maximum response values for all independent simulation runs, the mean value can still be calculated, but the calculation of the variance in this range is not meaningful. The same problem occurs in the interval of the maximum response values. Therefore, the response values smaller than the largest minimum response and the ones larger than the smallest maximum response are not accounted for the variance calculation. In the implementation, N_T equidistant response values in the interval between the largest minimum response and smallest maximum response are accounted for the variance calculation and their corresponding probabilities are found by linear interpolation.

This leads to the problem that large ranges of threshold values cannot be accounted for the failure probability c.o.v. estimation, if the failure probability graph varies significantly over the threshold domain. In the considered example, this happens in regions where the failure probability graph consists of large steps which highly vary in independent simulation runs. Consequently, the higher the variation of the failure probability graph is, the smaller will be the accounted threshold interval for the c.o.v. estimation and the shorter will be the c.o.v. graph over the failure probability domain as shown in the following.

Figures 4.45 and 4.46 show the c.o.v. values of the failure probability estimations of 50 independent simulation runs. It can be seen that c.o.v. values for SS/S are also higher for the values of k_d, where the estimator is still unbiased. Furthermore, the c.o.v. graphs are cut as described above, because for larger values of k_d, the estimator gets biased for smaller failure probabilities. This demonstrates, that the drawback of the splitting method also influences the estimation with smaller stiffness values, which means that the c.o.v. values for SS/S in the experiments for the non-linear and even for the linear shear building may be increased due to this problem.



(a) (b)

Figure 4.45.: c.o.v. for different values of k_d: (a) SS/MCMC (b) SS/S.

Furthermore, Figure 4.46 depicts that SS/H is again the most robust method, because it obtained the lowest c.o.v. values among all subset simulation methods. Finally, note that SS/MCMC and SS/H also deliver biased estimation for larger values of k_d (e.g. $k_d = 20 \times 10^6$ N/m) and only for very high stiffness values of (e.g. $k_d = 40 \times 10^6$ N/m) the probability graphs have a stepwise shape.

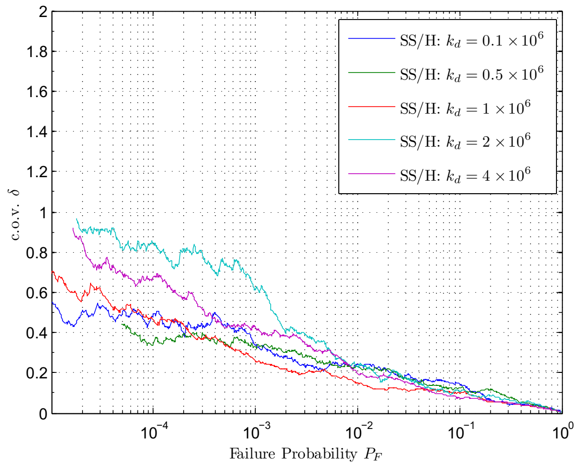


Figure 4.46.: c.o.v. for different values of k_d: SS/H.

4.5. Efficiency of the Sampling Methods

4.5.1. Computational Efficiency vs. Estimation Efficiency

The subset simulation methods are found to have different run times for the same number of samples. As already mentioned, this is due to the fact that SS/MCMC performs exactly one system analysis per sample, SS/S performs less and SS/H performs more than one system analysis per sample on average. The actual differences depend on the sampling problem and are shown in Table 4.3. Note that the run times are also different for sequential and parallel computing.

Method	SDOF		linear shear building - case 2 -		non linear shear building	
	sequential	parallel	sequential	parallel	sequential	parallel
SS/MCMC	1.277	1.360	1.382	1.257	1.631	1.642
SS/S	1.838	1.597	1.530	1.604	1.815	1.840
SS/H	1	1	1	1	1	1

Table 4.3.: Proportional computational effort for different variants of the subset simulation methods.

Due to the runtime differences, a faster method could use more samples in the same run time to reduce the estimator c.o.v.. As shown in chapter 2 the relation between the number of samples and the estimator c.o.v. is approximately quadratically for all subset simulation methods. For instance, to reduce the c.o.v. by half, four times more samples are needed. Figure 4.47 shows the c.o.v. values of the methods weighted according to their computational effort. Since the majority of the experiments have been made in parallel computing mode, the c.o.v. in Figure 4.47 have been divided by the square root of the parallel proportion values in table 4.3.

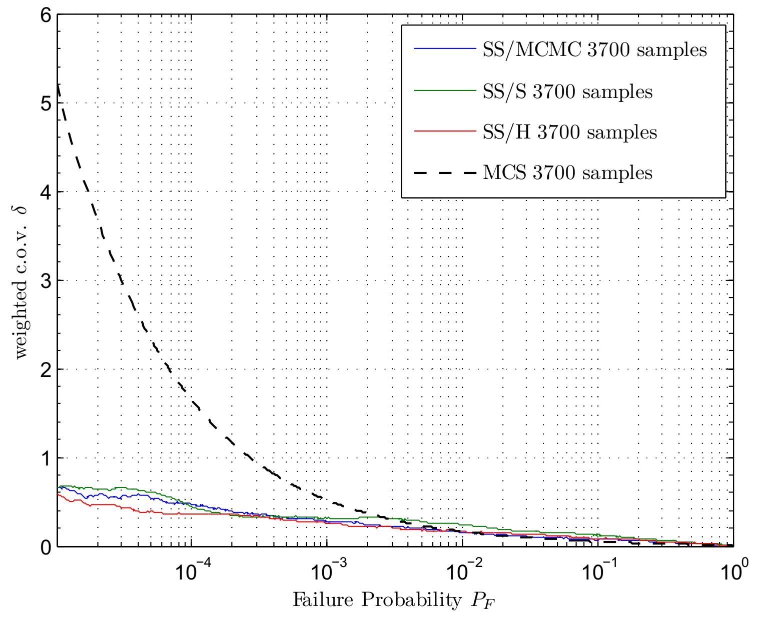


Figure 4.47.: Weighted c.o.v. for the SDOF example.

It can be seen that the estimation results for all three variants of the subset simulation method are similar and show only little differences. Correspondingly, Figure 4.48 shows the weighted c.o.v. for the linear shear building example (case 2). The results for case 1 have been found to be very similar and are not shown here.

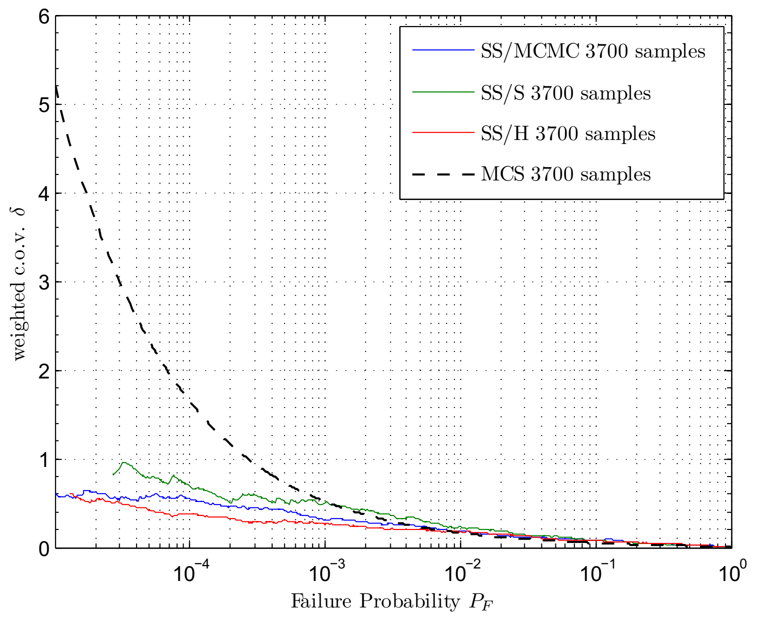


Figure 4.48.: Weighted c.o.v. for the linear shear building example - case 2.

The weighted c.o.v. for the non-linear shear building example is shown in Figure 4.49.

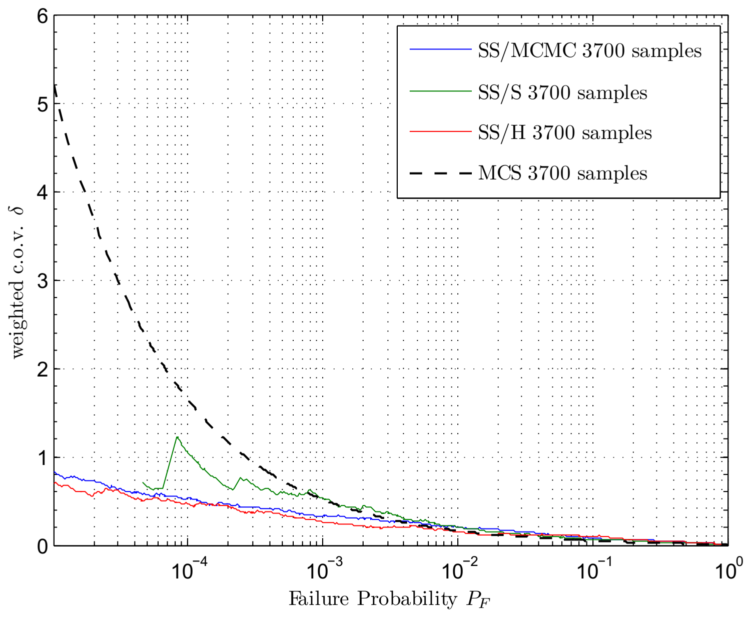


Figure 4.49.: Weighted c.o.v. for the non-linear shear building example.

All figure show that c.o.v. values weighted with the method run time lead to very similar results. The bad performance of the splitting method on the non-linear shear building in Figure 4.49 may be due to the problem described in subsection 4.4.3. Therefore, no clear winner can be identified for the examples studied in this thesis.

4.5.2. Parallelization

This section gives an overview of the performance gain due to symmetric multiprocessing. The test were run on a notebook with an Intel Core2 Duo T7500 processor with 2.2 GHz frequency, 2 GB RAM and OS: Windows Vista Business. The computer has two processor cores which can work independently on different tasks.

In the following diagrams the different sampling methods are identified by the numbers shown in Table 4.4.

Number	sampling method
1	MCS
2	SS/MCMC
3	SS/S
4	SS/H
5	IS / ISD A
6	IS / ISD B

Table 4.4.: Short terms for the sampling methods.

Figure 4.50 shows the time percentage which has been spent in the parallel program section for the SDOF oscillator example and for the linear shear building example - case 1. Case 2 is shown in Figure 4.51 (a), together with the results for the non-linear shear building in Figure 4.51 (b). The parallel program proportion describes indirectly the maximum speedup that can be achieved with parallelization techniques. In comparison, the figure also shows the percental time being spent in the parallel program proportion when the program was actually running in parallel. This percentage in comparison with the sequential result provides information about how the proportions between sequential and parallel program proportions have changed. For large parallel program proportions a small change in this relation gives information about the potential for further parallelization.

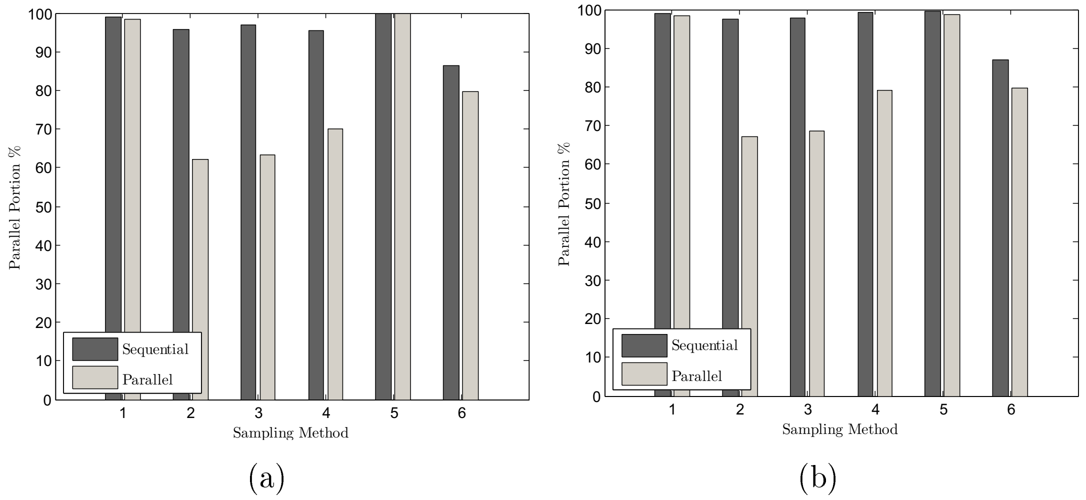
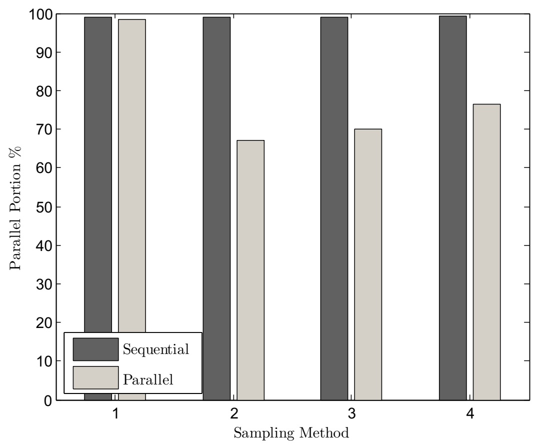


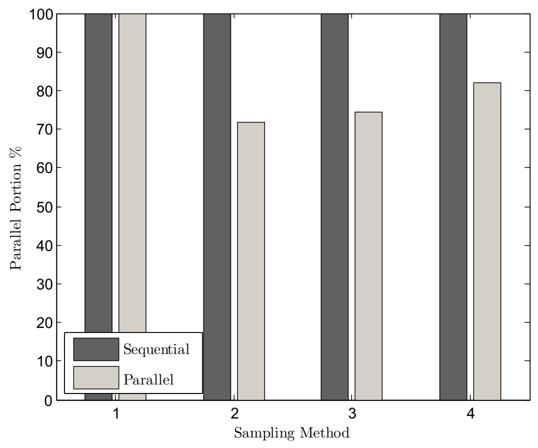
Figure 4.50.: Percentual time spent in parallel program proportions in sequential and parallel simulation runs. (a) SDOF example (b) linear shear building example - case 1.

One can see in both Figures 4.50, 4.51 that the proportion between sequential and parallel does only slightly change for MCS. This is natural since MCS is fully parallelizable and the maximum rate for parallelization can be expected. Also, the importance sampling method with ISD A led to similar results. This may be due to several reasons. Generally, one can say that the importance sampling methods have better sequential performance, because the subset sampling methods deal with much more data and some of the statistic data is written to the hard disk during the sampling process. This is not necessary for the importance sampling methods. The influence of this issue on the parallelization is, however, difficult to predict. On the one hand parallel physical writing to the disk is not possible and on the other hand all input/output operations are cached by the operation system.

The difference between the the importance sampling methods is due to the different complexity of the estimator. While the importance sampling with ISD A spends nearly all the time in the sampling process (step 3), the importance sampling procedure with ISD B usually has the largest program proportion within the precalculation (step 2).



(a)



(b)

Figure 4.51.: Parallel program proportions for (a) the linear shear building example - case 2 (b) non-linear shear building.

This step has in turn some sequential synchronization steps in between, because the precalculated values partially depend on each other. In general one can say, that the sequential code parts of the importance sampling method have higher proportions due to the high efficiency of the the estimator using ISD B. Therefore, the parallel program proportion is less. This is also true for the subset methods shown in the figures. For instance, by comparing Figure 4.50 (a) and Figure 4.51, one can see that the parallel program proportions increase with the complexity level of the sampling problem.

However, the most important measure is the speedup, which simply tells how much faster the parallel processing in comparison to the pure sequential program is. The obtained speedup values for the SDOF example and the linear shear building are shown in Figure 4.52 (a) and (b), respectively.

The speedup rates are between 1.4 and 1.7 which are under the limits that have been obtained by Amdahl's law (Equation 3.2). As stated above, this may be due to the program sections which cannot run in parallel. However, the parallel program proportion of all subset simulation methods is high and a better parallel performance can be expected for more complex response functions. This can also be seen in the figures as the the parallel portion increases with the complexity of the sampling problem. The reliability analysis of industry size structures can therefore be expected to be highly parallelizable.

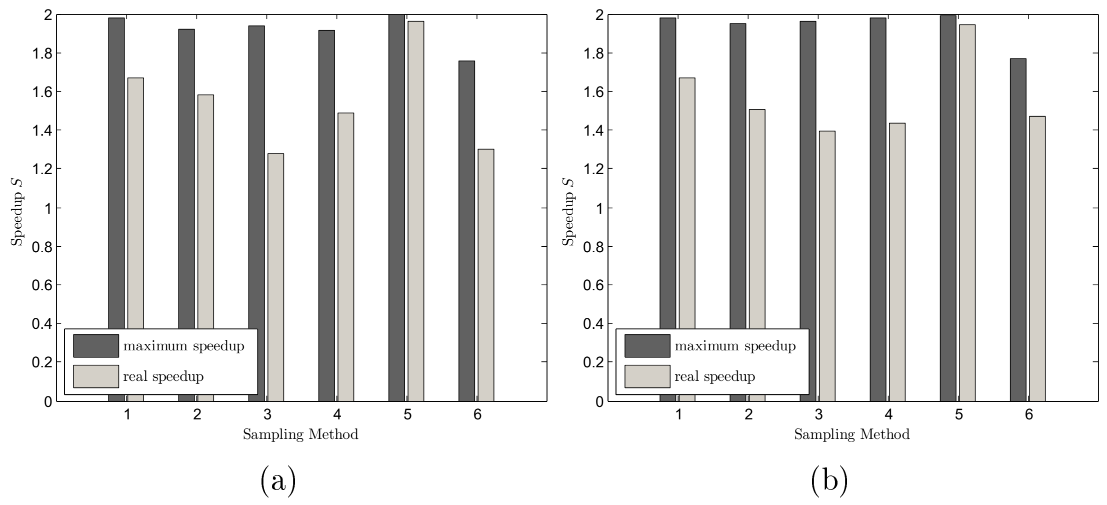


Figure 4.52.: Maximum speedup by Amdahl's law and real speedup. (a) SDOF example (b) linear shear building example - case 1.

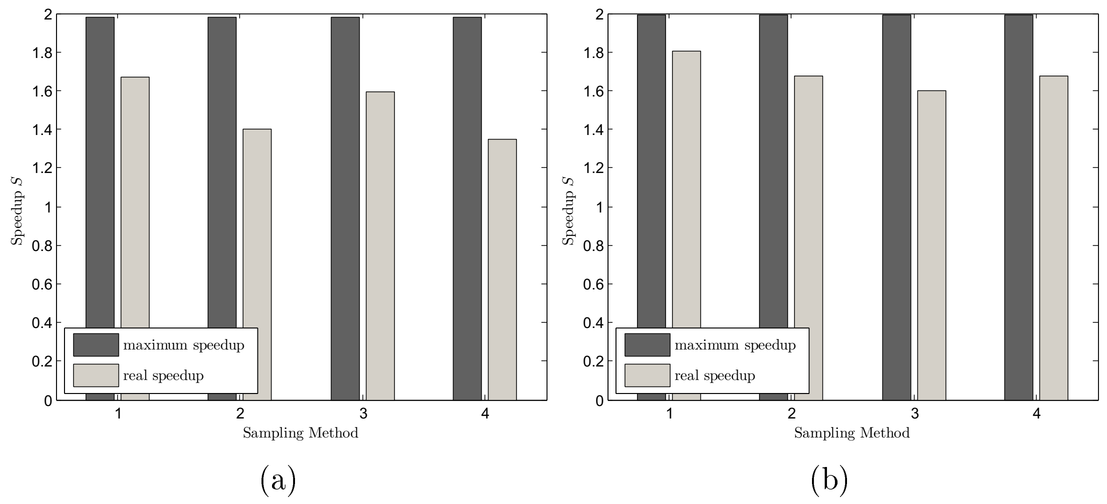


Figure 4.53.: Maximum and real speedup - (a) linear shear building example - case 2 (b) non-linear shear building.

4.6. Summary

All simulation methods have been studied with respect to their estimator bias and their estimation as well as their computational efficiency. All of them have proved to be much more efficient than direct Monte Carlo simulation in order to estimate small failure probabilities.

Subset simulation methods

The choice of the parameters for subset simulation has been studied empirically. It has been found that subset simulation methods are widely independent of the a priori choice p_0 of the conditional failure probabilities. However, in order to avoid high c.o.v. values, p_0 should be chosen to be greater than 0.1.

Furthermore, the subset simulation methods which make use of MCMC simulation are also robust to the choice of reasonable proposal functions. The shape of the proposal function seems to have little influence on the estimation procedure and therefore the most simply proposal function can be readily used. For the examples considered in this thesis, the size of the support region does also not influence the estimator bias, but the estimator c.o.v. has been found to reach its minimum for values around 1 for the uniform proposal and 0.6 for the Gaussian proposal function. The hybrid method of subset simulation has shown to be more robust to the choice of any parameter and usually delivers lower c.o.v. values in comparison to SS/MCMC and SS/S. The SS/S method has the best computational efficiency, but it has drawbacks to explore particular elementary failure regions, which may effect the estimator c.o.v. or bias in some scenarios.

For the reliability investigation of general systems subset simulation is a powerful tool to decrease the c.o.v. of the estimator significantly for small probabilities. For the failure probability estimation of large systems the computational efficiency can be improved by magnitudes in comparison to direct MCS estimation.

In [14] the three different subset simulation methods have already been discussed and compared in an analytical way. It is stated, that the c.o.v. values for the estimation using either SS/MCMC or SS/S are problem dependent and each method may outperform the other. The hybrid method, SS/H, combines the advantages of both methods and ensures that the estimator c.o.v. will always be lower or equal than that of SS/MCMC or SS/S. These statements are mathematically grounded and have also been verified empirically in several experiments in this work.

Importance Sampling

The advantages of the described importance sampling method are obvious. The method is very fast and extremely efficient to estimate small failure probabilities for linear systems. The importance sampling method highly exploits the knowledge about the shape of the failure region and is free of any parameters. It has been shown that the construction of an ISD from conditional elementary failure probabilities (ISD B) leads to a major advantage in efficiency, because the choice is close to the optimal sampling density. Therefore, this ISD should be preferred, because it is much more efficient in both ways, estimation efficiency and computational efficiency. However, for larger failure probabilities (> 0.5) and especially for failure probabilities close to one, the importance sampling method with both ISDs have bad performance, that is, the c.o.v. values are higher compared to direct MCS. However, the methods are made to estimate small failure probabilities, but one should consider the limited generality.

Therefore, the importance sampling methods are perfectly suited within a framework where small failure probabilities need to be calculated rapidly, e.g. reliability-based optimization [5, 15]. However, note that the described importance sampling method is only applicable to linear systems. Moreover, the method is less suitable to get broad and accurate information about the whole failure probability range for a given structure. That is, in order to construct the whole failure probability graph the estimation must be repeated at predefined steps, which also need to be found before. Considering this aspect, the subset simulation methods are much better suited for this task and they also have good performance on large failure probability estimates.

Efficiency and Parallelization

It has been shown that the three variants of the subset simulation methods have different efficiency with respect to the estimator c.o.v. as well as with respect to the computational effort. Based on the examples studied in this thesis no clear winner could be identified, because all three methods roughly led to similar results if both criteria are considered. In comparison with the importance sampling methods the subset simulation methods are clearly outperformed, but their efficiency is bought by a much lower level of generality.

It has been shown that the subset simulation procedure has little influence on the parallelizable program proportion and thus on the scalability of the parallelization. Hence, all subset simulation methods are well suited to solve failure probability estimations in a parallel way. The importance sampling methods are also parallelizable, but the speedup factor for the efficient ISD B is moderate due to sequential code parts and numerous data synchronization steps in the precalculation step. This is because of the efficient sampling process and the sufficiency of a small number of samples for a good estimation and therefore most of the processing time is spent in the preprocessing step.

5. Summary and Outlook

In this thesis some recent failure probability estimation techniques have been reviewed. According to literature, these methods are currently one of the most efficient for their corresponding level of generality. However, research on this topic is in progress and several extensions (e.g.[29]) and combinations of methods exist. For instance, a combination of Markov chain Monte Carlo simulation and importance sampling has been studied in [10].

All described subset simulation methods significantly reduce the estimator c.o.v. for small failure probabilities. Observing the c.o.v. over the logarithm of decreasing failure probabilities, the c.o.v. of the subset simulation methods approximately grows linearly, while in contrast, the c.o.v. for MCS grows exponentially on the same scale.

It has been shown that the usage of memory resources of all subset simulation methods can be reduced considerably (by factor 10) without noticeable loss of performance. This is an important improvement, since most of the memory resources are usually needed to store the finite element model of the structure under study. Moreover, several possibilities for runtime improvements of the simulation methods have been discussed and incorporated in the proposed implementation.

In this work a general purpose software framework for failure probability estimation has been developed which is widely problem independent and expendable for other methods. Furthermore, data structures and methods have been integrated to facilitate a parallel processing scheme. The presented methods have been implemented in a sequential as well as in a parallel manner. The sampling methods have been implemented in an object orientated way according to their generality. This software structure facilitates the application of the methods to new sampling problems and provides a high rate of source code re-usability. Due to the parallel processing integration the definition of new sampling problems is widely independent of complex parallelization techniques. That is, for example, if a sampling problem does not make use of thread-unsafe methods, the user only has to move the algorithm state data into a predefined data structure for sensitive data and the sampling problem can be readily analyzed by parallel computation.

The proposed software implementation has been tested in order to investigate the applicability of the sampling methods to parallel computation. Furthermore, the computational efficiency as well as the estimation efficiency has been investigated on several examples and their influences are discussed. Moreover, the choice of the algorithm parameters has been investigated empirically and the corresponding results agree with other empirical studies or support theoretical derivations. During the investigation of the implemented simulation methods a drawback of the subset simulation method with splitting has been identified and studied empirically. A more fundamental investigation of the identified problems is left for future work. Alternatives, in order to prevent incorrect failure probability estimations are already given by the other two variants of subset simulation.

In short, the advantages, disadvantages and limits of the described methods have been identified and verified by empirical studies.

This study has shown that the subset simulation methods are generally well suitable for parallelization. Their application to distributed computing over several computers, however, is more complicated. The higher complexity of synchronization and workload distribution will increase the sequential program proportion and increase the parallel overhead, but with an increasing complexity of the problem under study this will be worth the effort and is left for future work.

It has been shown that knowledge about the system under study - if used in a proper way - can enormously increase the efficiency to estimate small failure probabilities. However, the efficiency is bought by less generality of the method and therefore especially better estimation methods for non-linear systems need to be developed in order to increase the efficiency of the reliability estimation. Therefore, the combination of the estimation methods with approximation concepts is under investigation (e.g. [5, 15]) and promising to further improve the efficiency of reliability methods with little loss of accuracy. In view of this work, it then needs to be investigated how approximation methods or other improvement concepts can be incorporated with the parallelization.

A. Appendix

A.1. Numerical Integration

A.1.1. Newmark Method

The Newmark method or Newmark-beta method describes actually a class of different numerical integration methods to solve differential equations. The parameters β, γ eventually define which numerical integration method is applied.

Given are the parameters

Δt	size of the time fraction
β, γ	method parameters

and the differential equation:

$$\ddot{x}(t) + 2\zeta\omega\dot{x}(t) + \omega^2 x(t) = u(t) \tag{A.1}$$

The base equations of the Newmark method are given by

$$\dot{x}(t+\Delta t) = \dot{x}(t) + \left[(1-\gamma)\ddot{x}(t) + \gamma\ddot{x}(t+\Delta t)\right]\Delta t \tag{A.2}$$

$$x(t+\Delta t) = x(t) + \dot{x}(t)\Delta t + \left[\left(\frac{1}{2} - \beta\right)\ddot{x}(t) + \beta\ddot{x}(t+\Delta t)\right]\Delta t^2 \tag{A.3}$$

With these equations the following iterative procedure can be derived to numerically solve Equation (A.1).

Initialization of the iteration:

$$x(0) = x_0 \tag{A.4}$$

$$\dot{x}(0) = \dot{x}_0 \tag{A.5}$$

$$\ddot{x}(0) = u(0) - 2\zeta\omega\dot{x}(0) - \omega^2 x(0) \tag{A.6}$$

Iteration:

$$\begin{aligned} x(t+\Delta t) =& \frac{\beta u(t)\Delta t^2 + (1+2\gamma\zeta\omega)\,x(t) + \big(\Delta t + 2\,(\gamma-\beta)\,\zeta\omega\Delta t^2\big)\dot{x}(t)}{1+2\gamma\zeta\omega\Delta t+\beta\omega^2\Delta t^2} \\ &+ \frac{\big((0.5-\beta)\Delta t^2 + \Delta t^3(\gamma-2\beta)\zeta\omega\big)\ddot{x}(t)}{1+2\gamma\zeta\omega\Delta t+\beta\omega^2\Delta t^2} \end{aligned} \tag{A.7}$$

$$\ddot{x}(t+\Delta t) = \frac{x(t+\Delta t)-x(t)}{\beta\Delta t^2} - \frac{\dot{x}(t)}{\beta\Delta t} - \left(\frac{1}{2\beta}-1\right)\ddot{x}(t) \tag{A.8}$$

$$\dot{x}(t+\Delta t) = \dot{x}(t) + \big((1-\gamma)\ddot{x}(t)+\gamma\ddot{x}(t+1)\big)\Delta t \tag{A.9}$$

Popular choices of the method parameters are

- $\beta = 1/6 \quad \gamma = 1/2$ - assumes linearity of accelerations and is conditionally stable
- $\beta = 1/4 \quad \gamma = 1/2$ - assumes constant accelerations and is unconditionally stable

For the examples in this thesis, the latter choice has been used.

A.1.2. Numerical Integration of Non-Linear Systems

In section 4.2.3 the solution of two coupled differential equations is necessary in order to obtain the response of the non-linear structure. An iterative numerical integration scheme has been adopted from [6], which is explained in the following. After the modal analysis of the differential equation, the solution for the linear case is given by means of the modal responses due to the substitution

$$\{x(t)\} = [\phi]\,\{\eta(t)\} \tag{A.10}$$

However, for non-linear systems, the behavior of the non-linear restoring force $\{q\,(t)\}$ is also described by a differential equation, that is,

$$\{\dot{q}\,(t)\} = \{g\,(\{x(t)\}\,,\{\dot{x}(t)\}\,,\{q\,(t)\})\} \tag{A.11}$$

It can be seen that the solution of Equation (A.11) depends on $\{x(t)\}$ and $\{\dot{x}(t)\}$, which in turn can be obtained from the solution of the equation of motion

$$\begin{aligned} &\ddot{\eta}_j(t) + 2\zeta_j\omega_j\dot{\eta}_j(t) + \omega_j^2\eta_j(t) = -\frac{\{\phi_j\}^T[M]\{1\}}{M_j}\ddot{a}(t) - \frac{\{\phi_j\}^T[R]}{M_j}\{q\,(\{x(t)\})\} \\ &j = 1,2,\ldots,n \end{aligned} \tag{A.12}$$

With the Newmark method the solution of the equation of motion at time $t + \Delta t$ can be computed, given the solution at time t. In order to do that the value of $\{q(\cdot)\}$ is required at time $t + \Delta t$. Therefore, the two differential equations have to be solved in an iterative solution scheme. First, the iteration starts ($k = 0$) with the assumption that the value of $\{q(\cdot)\}$ at the next time step is close to the value at the previous time step, that is

$$\left\{q^{(0)}\left(\{x(t+\Delta t)\}, \{\dot{x}(t+\Delta t)\}\right)\right\} = \{q\left(\{x(t)\}, \{\dot{x}(t)\}\right)\} \tag{A.13}$$

Using this initial value for $\{q(\cdot)\}$ a solution for $\left\{x^{(k+1)}(t+\Delta t)\right\}$ and $\left\{\dot{x}^{(k+1)}(t+\Delta t)\right\}$ can be obtained by solving Equation (A.12) and using Equation (A.10). With these results Equation (A.11) can be integrated using the Crank-Nicolson method [26], given by

$$\left\{q^{(k+1)}(t+\Delta t)\right\} = \{q(t)\} + \frac{\Delta t}{2}\Big(\{\dot{q}(t)\} + \left\{\dot{q}^{(k+1)}(t+\Delta t)\right\}\Big) \tag{A.14}$$

where

$$\left\{\dot{q}^{(k+1)}(t+\Delta t)\right\} = \left\{g\left(\left\{x^{(k+1)}(t+\Delta t)\right\}, \left\{\dot{x}^{(k+1)}(t+\Delta t)\right\}, \left\{q^{(k)}(t+\Delta t)\right\}\right)\right\} \tag{A.15}$$

The new estimate of $\{q(\cdot)\}$ can in turn be used to solve Equation (A.12) again and the iteration starts over again ($k := k+1$). Finally, the iteration is stopped, if two consecutive estimations for the vector $\{q(\cdot)\}$ are sufficiently close to each other, that is,

$$\frac{\left\|\left\{q^{(k+1)}(t+\Delta t)\right\} - \left\{q^{(k)}(t+\Delta t)\right\}\right\|}{\left\|\left\{q^{(k+1)}(t+\Delta t)\right\}\right\|} \leq \epsilon \tag{A.16}$$

where ϵ is a predefined tolerance. For the example in section 4.2.3 a tolerance value of $\epsilon = 0.001$ has been used. The applied Crank-Nicolson method is unconditionally stable and works perfectly with the constant acceleration Newmark method.

A.2. Derivations

A.2.1. Efficient Evaluation of the Importance Sampling Estimator

The failure probability estimator for importance sampling using ISD A can be transformed to be computational more efficient. The estimator has been derived as

$$\hat{P}_F = \frac{1}{N} \sum_{k=1}^{N} \frac{\mathbb{I}\left(r\left(\mathbf{z}^k\right) \in F\right) \phi\left(\mathbf{z}^k\right)}{\sum_{i=1}^{n} \sum_{l=1}^{n_t} w_{il} \cdot \phi\left(\mathbf{z}^k - \mathbf{z}^*_{il}\right)} \tag{A.17}$$

with the joint probability function

$$\phi\left(\mathbf{z}\right) = (2\pi)^{-q/2} \exp\left[-\frac{1}{2} \sum_{j=1}^{q} z_j^2\right] \tag{A.18}$$

The evaluation of the exponential function is computationally very expensive and should thus be avoided if possible. Their number of evaluations can be reduced by combining the exponential functions of the numerator and the denominator. Applying Equation (A.18) to (A.17) leads to

$$\begin{aligned}
\hat{P}_F &= \frac{1}{N} \sum_{k=1}^{N} \frac{\mathbb{I}\left(r\left(\mathbf{z}^k\right) \in F\right) \exp\left[-\frac{1}{2}\sum_{j=1}^{q}(\mathbf{z}_j^k)^2\right] \cdot \exp\left[\frac{1}{2}\sum_{j=1}^{q}(\mathbf{z}_j^k)^2\right]}{\left(\sum_{i=1}^{n}\sum_{l=1}^{n_t} w_{il} \cdot \exp\left[-\frac{1}{2}\sum_{j=1}^{q}(\mathbf{z}_j^k - \mathbf{z}^*_{ilj})^2\right]\right) \exp\left[\frac{1}{2}\sum_{j=1}^{q}(\mathbf{z}_j^k)^2\right]} && \text{(A.19)} \\
&= \frac{1}{N} \sum_{k=1}^{N} \frac{\mathbb{I}\left(r\left(\mathbf{z}^k\right) \in F\right)}{\sum_{i=1}^{n}\sum_{l=1}^{n_t} w_{il} \cdot \exp\left[-\frac{1}{2}\sum_{j=1}^{q}(\mathbf{z}_j^k - \mathbf{z}^*_{ilj})^2 + \frac{1}{2}\sum_{j=1}^{q}(\mathbf{z}_j^k)^2\right]} && \text{(A.20)} \\
&= \frac{1}{N} \sum_{k=1}^{N} \frac{\mathbb{I}\left(r\left(\mathbf{z}^k\right) \in F\right)}{\sum_{i=1}^{n}\sum_{l=1}^{n_t} w_{il} \cdot \exp\left[\frac{1}{2}\left(\sum_{j=1}^{q}(\mathbf{z}_j^k)^2 - (\mathbf{z}_j^k - \mathbf{z}^*_{ilj})^2\right)\right]} && \text{(A.21)}
\end{aligned}$$

The derived form of the estimator is much more efficient. However, further improvements are possible. Remember that $q = p \cdot n_t$ is the dimension of each design point. With the definition $\bar{q} = p \cdot l$ Equation (A.21) can be written as

$$\hat{P}_F = \frac{1}{N}\sum_{k=1}^{N}\frac{\mathbb{I}\left(r\left(\mathbf{z}^k\right)\in F\right)}{\sum_{i=1}^{n}\sum_{l=1}^{n_t} w_{il}\exp\left[\frac{1}{2}\Big(\sum_{j=1}^{\bar{q}}(\mathbf{z}_j^k)^2-(\mathbf{z}_j^k-\mathbf{z}_{ilj}^*)^2+\underbrace{\sum_{j=\bar{q}+1}^{q}(\mathbf{z}_j^k)^2-(\mathbf{z}_j^k-\mathbf{z}_{ilj}^*)^2}_{0}\Big)\right]} \tag{A.22}$$

$$= \frac{1}{N}\sum_{k=1}^{N}\frac{\mathbb{I}\left(r\left(\mathbf{z}^k\right)\in F\right)}{\sum_{i=1}^{n}\sum_{l=1}^{n_t} w_{il}\exp\left[\frac{1}{2}\Big(\sum_{j=1}^{\bar{q}}(\mathbf{z}_j^k)^2-(\mathbf{z}_j^k-\mathbf{z}_{ilj}^*)^2\Big)\right]} \tag{A.23}$$

This means that it is sufficient to sum up the design points only until their corresponding design point index l, because all values of the design point with larger index values are zero. Only this transformation easily doubles the computational efficiency of the estimator. Moreover, many transformations in the importance sampling procedure for both ISDs can be simplified in the same manner to exploit the special shape of the design points in order to improve the computational efficiency. For instance, the reduction to do all design point calculations only until the corresponding design point index yield a speedup of factor 16 for the SDOF example with ISD A and 1000 samples. For importance sampling with ISD B the sampling process has little proportion to the overall runtime and the transformation has less impact, but still led to a speedup of factor 2 for the same number of samples. This impressively illustrates that such details can be more important than the parallelization of the algorithm which costs more effort and special hardware.

Notation

b	limit value to define failure region F
$[C]$	damping matrix
$e(t)$	envelope function to model non-stationary random processes for seismic system input
$E[\cdot]$	expected value of the argument
F	failure region $F \subseteq \mathcal{X}$, which defines undesired system states
$g(\cdot)$	maps a system state to an engineering demand parameter
$g_{ij}(\cdot,\cdot)$	discrete unit impulse response function for input j and output i
$h_{ij}(\cdot,\cdot)$	continuous unit impulse response function for input j and output i
$h(\cdot,\cdot,\cdot)$	function which describes the system dynamics of a causal system
$H(\cdot)$	Heaviside step function (unit step function)
$\mathbb{I}(\cdot)$	indicator function, is 1 if the argument is true,otherwise 0
$[K]$	stiffness matrix
k_d	stiffness of the non-linear device
l_j	half interval size for the uniform proposal function for dimension j
m	number of subset simulation stages
M	number of modes used in modal analysis of the structure
$[M]$	mass matrix
n	dimension of the system state vector x
n_t	number of discrete time steps within the duration of study
N	number of samples
N_i	number of samples in subset simulation stage i
N_T	total number of samples used/needed to estimate P_F
$O(\cdot)$	big o-notation: describes a class of functions/algorithms with their order which is given by the function argument
p	dimension of the input vector u
p_0	a priori value for partial failure probabilities (stages $i = 1, \ldots, m-1$)
$p(\cdot)$	probability density function evaluated at its argument
P	percentage of the parallelizable program proportion
P_F	failure probability
$P(\cdot)$	probability or probability content of a region given by the argument
$q(\cdot)$	conditional proposal PDF for the Metropolis-Hastings algorithm
q^1, q^2	plastic elongation of the non-linear device
r	exponent describing the sample correlation in MCMC simulation
$r(\cdot)$	response function
R_i	number of samples lying in failure region i
$r_N(t)$	vector of the non-linear restoring force
$[R]$	transformation matrix for the non-linear response
S	first passage point

S_j	white noise signal spectral intensity for input j
T	duration of study for dynamical problems
$[T]$	local - global coordinate transformation matrix
u	input vector / excitation (realization of random vector U)
u_p, u_y	model parameters of the non-linear device
U	random input vector
$\mathcal{U}$	input domain: set of all input vectors
$Var\left[\cdot\right]$	variance of the argument
$Z\left(t_l\right)$	series of zero-mean, unit-variance Gaussian variables
x	system state vector / trajectory (realization of random vector X)
X	random state vector
$\mathcal{X}$	system state domain: set of all state vectors
α	conditional random Gaussian vector in importance sampling
δ	coefficient of variance
$\phi\left(\cdot\right)$	q-dimensional joint Gaussian PDF
$\Phi\left(\cdot\right), \Phi^{-1}\left(\cdot\right)$	cumulative standard Gaussian distribution function and its inverse
γ	correlation factor of samples generated by MCMC
σ, σ^2	standard deviation and variance, respectively
ω	frequency
ζ	damping factor

List of Abbreviations

c.o.v.	coefficient of variation
FPP	first passage point
GUI	graphical user interface
i.i.d.	independent and identically distributed
IS	importance sampling
ISD	importance sampling density
MDOF	multi-degree of freedom
MCMC	Markov chain Monte Carlo
MCS	Monte Carlo simulation
MH	Metropolis-Hastings
PDF	probability density function
UML	unified modeling language
SDOF	single-degree of freedom
SMP	symmetric multiprocessing
SS	subset simulation
wrt.	with respect to

Bibliography

[1] J.Ching; S.K. Au; James L. Beck. Reliability estimation for dynamical systems subject to stochastic excitation using subset sampling with splitting. In *Computational Methods in Applied Mechanics and Engineering 194 (2005)*, pages 1557–1579, 2004.

[2] Bruce Ellingwood; Jun Kanda. *Structural Safety and its Quality Assurance.* ASCE Publications, 2005.

[3] R. E. Melchers. *Structural reliability analysis and prediction.* John Wiley and Sons, New York, 1999. 2nd Edition.

[4] Siu-Kui Au. *On the Solution of First Excursion Problems by Simulation with Applications to Probabilistic Seismic Performance Assessment.* PhD Thesis, California Institute of Technology, Pasadena, 2001.

[5] H.A. Jensen. Design and sensitivity analysis of dynamical systems subjected to stochastic loading. In *Computers and Structures*, 2005. 83(2005):1062-1075.

[6] H.A. Jensen; Michel A. Catalan. On the effects of non-linear elements in reliability-based optimal design of stochastic dynamic systems. In *Journal of Non-Linear Mechanics*, 2007. 42(2007):802-816.

[7] J.M.Hammersley; D.C.Handscomp. *Monte Carlo methods.* London: Methuen, 1964.

[8] R. Rubenstein. *Simulation and the Monte Carlo Method.* Wilay, New York, 1981.

[9] R.E. Melchers. Importance sampling in structural systems. In *Structural Safety*, 1989. 6:3-10.

[10] S.K. Au; James L. Beck. A new adaptive importance sampling scheme for reliability calculations. In *Structural Safety Volume 21, Issue 2, June 1999, Pages 135-158*, 1999. Elsevier, 1999.

[11] S.K. Au; James L. Beck. First excursion probabilities for linear systems by very efficient importance sampling. In *Probabilistic Engineering Mechanics 16 (2001) 193-207*, 2001. Elsevier, 2001.

[12] P.S. Koutsourelakis; H.J. Pradlwarter; G.I. Schuëller. Reliability of structures in high dimensions, part i: algorithms and applications. In *Probabilistic Engineering Mechanics*, 2004. 19(4):409(417).

[13] Siu-Kui Au; James L. Beck. Estimation of small failure probabilities in high dimensions by subset simulation. In *Probabilistic Engineering Mechanics 16 (2001):263-277*, 2001.

[14] J.Ching; James L. Beck; S.K. Au. Hybrid subset simulation method for reliability estimation of dynamical systems subject to stochastic excitation. In *Probabilistic Engineering Mechanics 20 (2005)*, pages 199–214, 2004.

[15] G.I.Schuëller H.A. Jensen; M.A. Valdebenito. An efficient reliability-based optimization scheme for uncertain linear systems subject to general gaussian excitation. In *Computational Methods in Applied Mechanics and Engineering*, 2008. doi:10.1016/j.cma.2008.01.003.

[16] Zdeněk Bittnar; Jiří Šejnoha. *Numerical Methods in Structural Mechanics*. Published by ASCE Publications, 1996.

[17] T. Hughes. *The finite element method: linear static and dynamic finite element analysis*. Englewood Cliffs, NJ: Prentice-Hall, 1987.

[18] R.F. Drenick. Model-free design of aseimic structures. In *Journal of Engineering Mechanics*, 1970. ASCE; 96:483-93.

[19] N. Metropolis; A.W. Rosenbluth; M.N. Rosenbluth; A.H. Teller; and E. Teller. Equations of state calculations by fast computing machines. In *Journal of Chemical Physics*, 1953. 21(6):1087-1092.

[20] W.K. Hastings. Monte carlo sampling methods using markov chains and their applications. In *Biometrika*, 1970. 57(1):97-109.

[21] J. S. Liu; F. Liang; W. H. Wong. The use of multiple-try method and local optimization in metropolis sampling. In *Journal of the American Statistical Association*, 2000. 96(454):561-573.

[22] Martin Liebscher; Stephan Pannier; Jan-Uwe Sickert; Wolfgang Graf. Efficiency improvement of stochastic simulation by means of subset sampling. In *LS-DYNA Anwenderforum, Ulm 2006*, pages K–I– 27 – 40, 2006.

[23] P. van Emde Boas. Preserving order in a forest in less than logarithmic time. In *Proceedings of the 16th Annual Symposium on Foundations of Computer Science, IEEE Computer Society*, pages 75–84, 1975.

[24] Thomas H. Cormen; Charles E. Leiserson; Ronald L. Rivest; and Clifford Stein. *Introduction to Algorithms*. MIT Press and McGraw-Hill, 2001. Second Edition.

[25] Gene Amdahl. Validity of the single processor approach to achieving large-scale computing capabilities. In *AFIPS Conference Proceedings (30)*, pages 483–485, 1967. http://www-inst.eecs.berkeley.edu/~n252/paper/Amdahl.pdf (visited: 15/05/2008).

[26] K. Subbaraj; M.A. Dokainish. A survey of direct time-integration methods in computational structural dynamics. In *II. Implicit methods. Computers and Structures*, 1989. 32(6):1387-401.

[27] Farzad Naeim. *The seismic design handbook*. Published by Springer, 2001. International Conference of Building Officials, Structural Engineers Association National Council.

[28] R.W. Clough; J. Penzien. *Structural Dynamics*. McGraw-Hill, Inc., New York,

NY, 1975.

[29] S.K Au; J.L. Beck. Subset simulation and its application to seismic risk based on dynamic analysis. In *Journal of Engineering and Mechanics, ASCE Aug. 2003*, 2003. DOI: 10.1061/(ASCE)0733-9399(2003)129:8(901).

[30] S.K Au; J.Ching; J.L. Beck. Application of subset simulation methods to reliability benchmark problems. In *Structural Safety (accepted for publication)*, 2006.

[31] S.K. Au; James L. Beck. On the solution of first-excursion failure problem for linear systems by efficient simulation. In *Technical Report: CaltechEERL:2000.EERL-2000-01, California Institute of Technology*, 2000. `http://nsdl.org/resource/2200/20061003163119647T` (visited: 10/07/2008).