# Subset Simulation with Multivariate Draw

Carl Ehrett

**Abstract**

Subset simulation is a specialized Markov chain Monte Carlo method for estimating low-probability failure events. Previous versions of subset simulation algorithms have used a component-wise proposal distribution, which can cause the algorithm to be less efficient when the parameters are correlated with respect to the failure event of interest. Here, a version of subset simulation is proposed in which an adaptive multivariate proposal distribution is used in order to facilitate convergence of the MCMC chains to the failure region. The method is illustrated with a simple example of finding the volume of a predefined high-dimensional shape, and then by application to a problem of estimating the downtime of a power plant following an earthquake.

## 1 Introduction

Suppose that $f$ is a function taking as input a parameter $\boldsymbol{\theta} \in \mathbb{R}^n$ which contains information about the state of components of a power plant prior to an earthquake of a certain magnitude, and returns the estimated downtime of the power plant following that earthquake. We might then define $F$ might to be the event that downtime is under, say, $g_t = 62.5$ days; i.e., $F = \{\boldsymbol{\theta} : f(\boldsymbol{\theta}) \leq g_t\}$. Even if we already fully understand the probability distribution of $\boldsymbol{\theta}$, we may not yet know the distribution of $\boldsymbol{\theta}|F$. For example, if we are expending finite resources to improve the resiliency of the plant against extended downtime following seismic events, we will want to have a sense of which parameter values $\boldsymbol{\theta}$ are in $F$; i.e., which states of the plant's components will lead to downtime under 62.5 days if an earthquake (of a given magnitude) occurs. These are some of the reasons why we may wish to be able to draw samples from $\boldsymbol{\theta}|F$.

Direct Monte Carlo simulation is the most straightforward way to approach the problem of drawing samples from $\boldsymbol{\theta}|F$ (and thereby estimating $P(F)$). That is, for a large integer $M$, one could simply draw $M$ samples $\boldsymbol{\theta}^{(1)}, \cdots, \boldsymbol{\theta}^{(M)}$ from the parameter space $\Theta$, then find their performance values: $f(\boldsymbol{\theta}^{(1)}), \cdots, f(\boldsymbol{\theta}^{(M)})$. Then one could use these performance values to find

$$\Theta_F := \{\boldsymbol{\theta}^{(j)} : 1 \leq j \leq M,\ f(\theta_j) < 62.5\}$$

This would easily allow one to estimate the probability of $F$ (as $|\Theta_F|/M$), and one could also treat $\Theta_F$ as a sample from $\boldsymbol{\theta}|F$, thereby learning about its distribution.

Several features of a problem, however, may make the direct Monte Carlo simulation strategy unworkable.

*Low failure probability.* It may be that the failure region of interest is highly improbable. For example, suppose that the probability of $F$ is $10^{-4}$. Then we expect to have to draw one million samples of $\boldsymbol{\theta}$ in order to produce just one hundred samples from our target failure region $F$. This is inefficient, and can be especially problematic when combined with the difficulties listed below.

*High cost of draw.* The probability distribution of $\boldsymbol{\theta}$ may be such that it is computationally expensive to draw samples from it. This will mean that the inefficiency of a direct Monte Carlo simulation has a high cost.

*High cost of performance function.* Even in those cases where we may easily and cheaply draw samples of $\boldsymbol{\theta}$, the performance function may be computationally expensive. Since, under the direct Monte Carlo method, we must evaluate each sample of $\boldsymbol{\theta}$ under the performance function, this can involve great inefficiency. Indeed, in a case where $P(F) = 10^{-4}$, if the performance function requires one second to run per sample, then we expect our 100 samples from $\boldsymbol{\theta}|F$ to require almost twelve days.

*High dimensionality.* If $\theta$ is high-dimensional—say, $\boldsymbol{\theta} \in \mathbb{R}^{1000}$—then due to the curse of dimensionality, a small sample from $F$ may not give us as much information as we require about the distribution of $\boldsymbol{\theta}|F$.

Subset simulation is a technique that may be used in such situations to more efficiently sample from a low-probability event. Originally developed by Au and Beck [2], subset simulation has seen numerous refinements and broad application in reliability engineering. The central idea of subset simulation is as follows. Define the failure region of interest as $F = \{\boldsymbol{\theta} : g(\boldsymbol{\theta}) > g_t\}$, where $g$ is a performance function, $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_n)$, and $g_t$ is a scalar threshold value defining the failure region of interest[1]. We may represent $F = F_m$ as the intersection of a collection of $m$ monotone decreasing intermediate failure regions which include $F_m$:

$$F = \cap_{i=1}^{m} F_i, \quad F_{i+1} \subseteq F_i \ \forall i = 1, \cdots, m-1$$

This is illustrated in Figure 1. With these intermediate failure regions, we may represent the probability of failure as a product of the conditional probabilities of intermediate failures:

$$\begin{aligned} P(\boldsymbol{\theta} \in F) = P(\boldsymbol{\theta} \in F_m) &= P(\boldsymbol{\theta} \in F_{m-1}) \cdot P(\boldsymbol{\theta} \in F_m | \boldsymbol{\theta} \in F_{m-1}) \\ &= P(\boldsymbol{\theta} \in F_1) \cdot \prod_{i=i}^{m-1} P(\boldsymbol{\theta} \in F_{i+1} | \boldsymbol{\theta} \in F_i) \end{aligned} \tag{1}$$

In order to estimate the conditional probabilities $P(\boldsymbol{\theta} \in F_{i+1} | \boldsymbol{\theta} \in F_i)$, Au and Beck offer a modification of the Metropolis algorithm [6], and show that the stationary distribution of the resulting Markov chain is indeed that of $\boldsymbol{\theta}|F_i$.

Various modifications to the basic subset simulation methodology have been made. Ching, Au and Beck [3] describe an alternative method, applicable to first-passage problems involving deterministic dynamical systems, of estimating

[1] E.g., in the earthquake example, we may set $g_t = 62.5$ in order to define the failure region of interest as being the event that total downtime is less than 62.5 days.

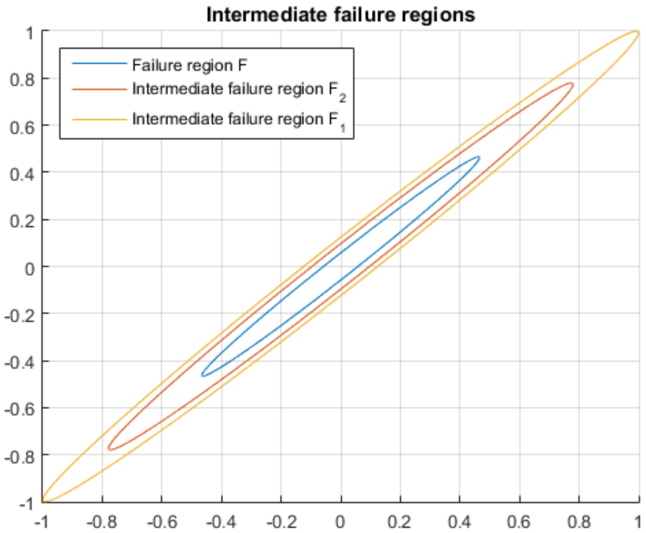


Figure 1: Each ellipse is the perimeter of an intermediate failure region. The target failure region $F$ is the intersection of all such intermediate failure regions.

the conditional probabilities $P(\boldsymbol{\theta} \in F_{i+1}|\boldsymbol{\theta} \in F_i)$. This method, which the authors refer to as subset simulation with splitting, utilizes direct Monte Carlo simulation rather than MCMC. Zuev et al. [9] describe a technique for scaling the MCMC algorithm for optimally efficient exploration of the parameter space, and find a theoretical basis for selecting intermediate failure regions so that each conditional probability $p_0 = P(\boldsymbol{\theta} \in F_{i+1}|\boldsymbol{\theta} \in F_i) \in [0.1, 0.3]$. They also develop a method of Bayesian post-processing so that subset simulation can be used to generate a probability distribution on $P(F)$, rather than a point estimate. Papaioannou et al. [7] propose a MCMC approach that uses adaptive scaling of the proposal distribution variance.

In this paper I propose a version of subset simulation which is designed for cases in which it is known or suspected that the components of $\boldsymbol{\theta}$ have nonzero correlation with respect to the failure region[2]. The central idea is to use a Gaussian multivariate proposal density to sample from each intermediate failure region $F_i$. The covariance matrix $\boldsymbol{\Sigma}_i$ for this distribution is set adaptively using the samples drawn from the previous intermediate failure region $F_{i-1}$. It follows from the way that intermediate failure regions are determined in the Au/Beck algorithm that some predetermined number of samples from $F_{i-1}$ will fall in $F_i$. Under the methodology suggested here, the covariance of the samples in $F_{i-1} \cap F_i$ is used to define $\boldsymbol{\Sigma}_i$, which is then used to draw samples from $F_i$.

Before elucidating the current proposed modification to the MCMC algorithm described by Au and Beck, I explore in detail their original proposal. In Section 2, Au and Beck's modified Metropolis algorithm (MMA) is described in detail. In Section 3, I present my own alternative multivariate proposal distribution algorithm (MVD). In Section 4, the two versions of subset simulation

[2] Note that we must still have that the components of $\boldsymbol{\theta}$ are independent; the correlation to be accommodated by the proposal here is of $\boldsymbol{\theta}|F$.

are illustrated by using them to estimate the volume of a predefined region of the parameter space. The region is defined in a way that exemplifies a case of parameters correlated with respect to the target failure region. In Section 5, the two techniques are applied to the case of estimating the probability that the downtime of a given power plant following an earthquake will not exceed 62.5 days.

## 2 Modified Metropolis Algorithm (MMA)

In their seminal 2001 paper, Au and Beck [2] describe an MCMC method for sampling in turn from intermediate failure regions $F_0, F_1, \cdots, F_m$, where $F_0$ is the entire parameter space and $F_m$ is the target failure region of interest. For some predetermined $N \in \mathbb{N}$, $N$ samples of $\boldsymbol{\theta}$ are drawn from $F_0$. For some predetermined $p_0 \in (0,1)$, the $N \cdot p_0$ samples with the highest performance function values are identified[3]; call these the *seeds* for $F_1$. That is, the seeds of $F_1$ are $N \cdot p_0$ samples from $F_0$ such that for any seed $\boldsymbol{\theta}$ and any sample $\boldsymbol{\tau}$ from $F_0$ which is not a seed of $F_1$, we have $g(\boldsymbol{\theta}) \geq g(\boldsymbol{\tau})$ (where $g$ is the performance function). Let $g_{t_1}$ be the mean of two values: the lowest performance value of the seeds of $F_1$, and the highest performance value from the samples of $F_0$ which are not seeds of $F_1$. Then for $i \geq 1$, $F_i$ is defined as $\{\boldsymbol{\theta} : g(\boldsymbol{\theta}) \geq g_{t_i}\}$.

To sample from $F_i$, the seeds for that level are each used as the starting position in an MCMC chain of length $1/p_0$. The resulting $N$ total samples are all in $F_i$, and the $N \cdot p_0$ samples with the highest performance values become the seeds of $F_{i+1}$. The samples are used to define $g_{t_{i+1}}$ (and thereby to define $F_{i+1}$). This process continues until $g_{t_m} \geq g_f$, where $g_f$ is the threshold of the failure region of interest.

Au and Beck's algorithm for this procedure, which they call the modified Metropolis algorithm (MMA), may be summarized as follows.

**Definitions:**

$n$ = the dimension of the parameter space

$N$ = the Monte Carlo sample size in each chain

$F$ = the failure region of interest

$F_i$ = the $i^{\text{th}}$ intermediate failure region, where $F_m = F$

$\boldsymbol{\theta}^{(i,j,k)}$ = $k^{\text{th}}$ sample drawn in the $j^{\text{th}}$ Markov chain in the failure region $F_i$, where $\boldsymbol{\theta}^{(i,j,k)} = [\theta_1^{(i,j,k)}, \theta_2^{(i,j,k)}, \cdots, \theta_n^{(i,j,k)}]$

$p_k(\cdot|\cdot)$ = conditional proposal distribution for drawing from $F_i$, with the symmetry property $p_k(a|b) = p_k(b|a)$

$\pi_k(\cdot)$ = target distribution of parameter $\theta_k$

$p_0$ = proportion of samples in each step used to define next intermediate failure region; $p_0 \in (0,1)$

$g_f$ = threshold of the performance function defining the failure region; i.e., $\boldsymbol{\theta} \in F \iff g(\boldsymbol{\theta}) \geq g_f$

[3] $N$ and $p_0$ should be chosen so that both $p_0^{-1}$ and $N \cdot p_0 \in \mathbb{N}$.

**MMA:**

```
Generate initial sample set:
    Generate samples $\theta_i^{(0,j,k)}$, $i = 1, \cdots, n$, $j = 1, \cdots, p_0^{-1}$, $k = 1, \cdots, N \cdot p_0$, using $\pi_k(\cdot)$.
    Set first intermediate threshold $g_{t_1}$ equal to the $100 \cdot p_0$ percentile of the values $\{g(\boldsymbol{\theta}^{(0,j,k)})\}$
    Set the seeds of $F_1$ as those $\boldsymbol{\theta} = \boldsymbol{\theta}^{(0,j,k)}$ s.t. $g(\boldsymbol{\theta}) \geq g_{t_1}$. Label them $\{\boldsymbol{\theta}^{(1,j,0)}\}$, $j = 1, \cdots, p_0^{-1}$.
While $g_{t_i} < g_f$:
    for $j = 1, \cdots, p_0^{-1}$:
        for $k = 1, \cdots, N \cdot p_0$:
            Generate a candidate state $\boldsymbol{\theta}^* = \{\theta_1^*, \cdots, \theta_n^*\}$ as follows:
            for $l = 1, \cdots, n$:
                Draw $\tau$ from $p_l(\cdot|\theta_l^{(i,j,k-1)})$
                With probability $\min\left(1, \alpha = \frac{\pi_l(\tau)}{\pi_l(\theta_l^{(i,j,k-1)})}\right)$:
                    set $\theta_l^* = \tau$
                Otherwise: set $\theta_l^* = \theta_l^{(i,j,k-1)}$
            end
            Set $\boldsymbol{\theta}^{(i,j,k)} = \begin{cases} \boldsymbol{\theta}^* & \text{if } \boldsymbol{\theta}^* \in F_i \\ \boldsymbol{\theta}^{(i,j,k-1)} & \text{otherwise} \end{cases}$
        end
    end
    Set $i = i + 1$
    Set new intermediate threshold $g_{t_i}$ equal to the $100 \cdot p_0$ percentile of the values $\{g(\boldsymbol{\theta}^{(i-1,j,k)})\}$
    Set the seeds of $F_i$ as those $\boldsymbol{\theta} = \boldsymbol{\theta}^{(i-1,j,k)}$ s.t. $g(\boldsymbol{\theta}) \geq g_{t_i}$. Label them $\{\boldsymbol{\theta}^{(i,j,0)}\}$, $j = 1, \cdots, p_0^{-1}$.
end
```

A single level of the algorithm—that is, one iteration of the while loop above—can be visualized in Figure 2.

Here, as in an ordinary Metropolis algorithm, a candidate $\tau$ is accepted as the next value in the chain only with probability $\min(1, \alpha)$. Where the initial values for the Markov chains come from the stationary distribution—which is the case here, since the seeds are drawn from $\boldsymbol{\theta}|F_i$— then this Metropolis step ensures that the new draws are from the same distribution. In contrast to an ordinary Metropolis algorithm, though, there are two opportunities here for a candidate $\tau$ to be rejected. It may be rejected, as in the usual Metropolis, because we first accept it only with probability $\min(1, \alpha)$, and then (even if it survives there) we will still reject it if it does not fall in the current intermediate failure region.

In the applications considered here, the parameter space is the unit hypercube, and truncated normals are used as the proposal densities $p_k(\cdot|\theta^{(i)}(k))$. Unlike in the modified Metropolis algorithm used by Au and Beck, this conditional proposal density is not symmetric. As a result, the algorithm above is modified slightly to act as an instance of the Metropolis-Hastings rather than Metropolis algorithm [5]. That is, one defines $\alpha$ in the acceptance probability

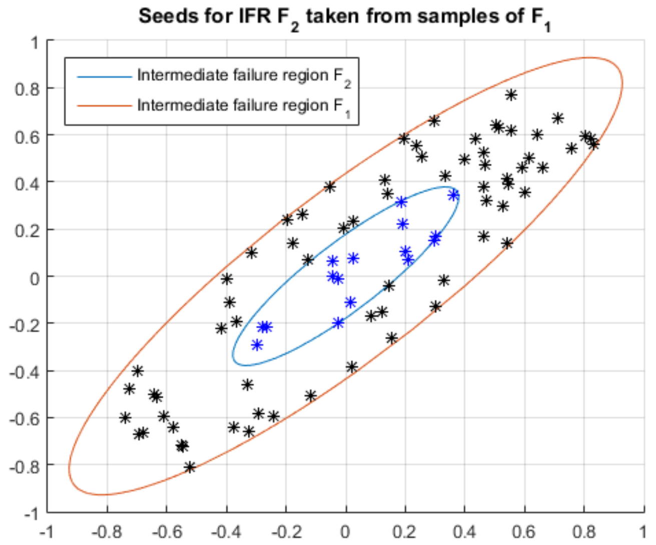


Figure 2: A single iteration of the while loop of the modified Metropolis algorithm. Samples (represented by asterisks) are drawn from $F_1$. For pre-specified $q$, samples in the top $q$ quantile with respect to the performance function are identified (blue asterisks). They are used to define $F_2$, and they become the seeds for the MCMC chains to sample from $F_2$. This ensures that the draws in the MCMC chains in the next loop of the algorithm will come from the same distribution as these seeds.

ratio as:

$$\alpha = \frac{\pi_l(\tau)}{\pi_l(\theta_l^{(i,j,k-1)})} \cdot \frac{p_k(\theta_l^{(i,j,k-1)}|\tau_l)}{p_k(\tau_l|\theta_l^{(i,j,k-1)})}$$

In the case of the truncated normal $p_k$ and uniform $\pi_k$, this expression simplifies to:

$$\alpha = \frac{\Phi(1-\theta_l^{(i,j,k-1)}) - \Phi(-\theta_l^{(i,j,k-1)})}{\Phi(1-\tau_l) - \Phi(-\tau_l)} \cdot I(\tau_l \in (0,1))$$

where $\Phi$ is the cdf of the standard normal and $I$ is the indicator function.

# 3 Multivariate draw algorithm (MVD)

MMA assumes that the $\theta_l$ are independent. This is why each of the $n$ candidates $\tau_l$ are drawn independently from the proposal distribution. Consider, though, a case in which the parameters are indeed independent in the parameter space, but nonetheless are correlated with respect to the failure region. That is, suppose that with respect to the failure region $F_k$, it turns out that $\theta_i$ and $\theta_k$ are highly correlated. In this case, drawing $\tau_i$ and $\tau_j$ individually is likely to lead to a candidate $\boldsymbol{\tau}$ which is not in $F_k$. As can be seen in the final part of the algorithm described above, in such a case $\boldsymbol{\tau}$ would be rejected. If the correlation is high enough, this can lead to a very high rejection rate of candidate parameter draws. This high rejection rate can mitigate the efficiency of the algorithm, thereby undermining the goal of subset simulation.

In such cases, one may harness the benefits of an adaptive multivariate proposal distribution that is estimated from the sample correlation amongst the sampled parameters [4]. Tailoring the proposal distribution to more closely match the support of the target distribution can lower the rate of rejection, thereby improving the efficiency of the algorithm.

Under this approach, a candidate $\boldsymbol{\tau}$ is drawn not componentwise, but rather as a single draw from a multivariate normal proposal distribution. In the example considered here, the model parameters are assumed to be uniformly distributed over their respective ranges. Thus, in order to employ a normal proposal distribution, we may first rescale each $\theta_l$ so that $\boldsymbol{\theta}$ lies in the unit hypercube, and then use the logit transform $\eta_l = \log\big(\theta_l/(1-\theta_l)\big)$.

In order for our proposal distribution to make use of the correlation amongst the parameters within each failure region, we must find an appropriate covariance matrix for the distribution. For the proposal distribution in the $i^{\text{th}}$ step (to generate samples in $F_i$ given seeds in $F_{i-1}$), let $S_i$ be the estimated sample covariance matrix of the draws generated in step $i-1$. Here, $F_0$ is the entire parameter space, and the $0^{\text{th}}$ step of the process is a direct Monte Carlo draw from $\boldsymbol{\theta}|F_0$ using the target distribution of the parameters.

Since we are considering cases in which there is strong correlation amongst the parameters, it may emerge that $S_i$ is almost singular, which makes the necessary matrix inversions computationally unstable. As in (e.g.) Ababou et al. [1], we may improve computational stability by adding a nugget—a small constant multiple of $\boldsymbol{I}$—to $\boldsymbol{S}_i$ [1]. That is, rather than let the covariance of the $i^{\text{th}}$ proposal distribution be $\boldsymbol{S}_i$, instead let $\boldsymbol{\Sigma}_i = \boldsymbol{S}_i + cI$, with $c$ a small constant.

Given the Gaussian conditional proposal distributions determined by the $\mathbf{\Sigma}_i$ (the mean of which is the value on which the distribution is conditioned), $N$ new samples are drawn in each step. The algorithm terminates when the target failure region has been reached. If, after the $i^{\text{th}}$ step, the target failure region has still not yet been reached, then a new covariance matrix $\mathbf{\Sigma}_{i+1}$ and a new intermediate failure region $F_{i+1}$ (and corresponding intermediate threshold) are defined using the $N$ samples from step $i$. The resulting multivariate draw (MVD) algorithm may be summarized as follows:

**Definitions:**

$n$ = the dimension of the parameter space

$N$ = the number of samples in the Markov Chain

$F$ = the failure region of interest

$F_i$ = the $i^{\text{th}}$ intermediate failure region, where $F_m = F$

$\boldsymbol{\theta}^{(i,j,k)} = k^{\text{th}}$ sample drawn in the $j^{\text{th}}$ Markov chain in the failure region $F_i$, where $\boldsymbol{\theta}^{(i,j,k)} = [\theta_1^{(i,j,k)}, \theta_2^{(i,j,k)}, \cdots, \theta_n^{(i,j,k)}]$

$\boldsymbol{\eta}^{(i,j,k)}$ = the logit transformation of $\boldsymbol{\theta}^{(i,j,k)}$, so that $\eta_l^{(i,j,k)} = \log\left(\frac{\theta_l^{(i,j,k)}}{1-\theta_l^{(i,j,k)}}\right)$

$p(\cdot|\cdot)$ = multivariate conditional proposal distribution for drawing $\boldsymbol{\eta}|F_i$

$\pi_k(\cdot)$ = target distribution of parameter $\theta_l$

$p_0$ = proportion of samples in each step used to define next intermediate failure region; $p_0 \in (0,1)$

$g_f$ = performance function value defining the target failure region; i.e., $\boldsymbol{\theta} \in F \iff g(\boldsymbol{\theta}) \geq g_f$

$\mathbf{S}_i$ = sample covariance of $\{\boldsymbol{\eta} \in F_i : \exists j, k \text{ s.t. } \boldsymbol{\eta} = \boldsymbol{\eta}^{(i-1,j,k)}\}$

**MVD Algorithm:**

```
Generate initial sample set and covariance matrix:
    Generate samples $\boldsymbol{\theta}^{(0,j,k)}$, $j = 1, \cdots, p_0^{-1}$, $k = 1, \cdots, N \cdot p_0$ using $\pi_k(\cdot)$.
    Set first intermediate threshold $g_{t_1}$ equal to the $100 \cdot p_0$ percentile of
    the values $\{g(\boldsymbol{\theta}^{(0,j,k)})\}$.
    Set the seeds of $F_1$ as those $\boldsymbol{\theta} = \boldsymbol{\theta}^{(0,j,k)}$ s.t. $g(\boldsymbol{\theta}) \geq g_{t_1}$. Label them
    $\{\boldsymbol{\theta}^{(1,j,0)}\}$, $j = 1, \cdots, p_0^{-1}$.
While $g_{t_i} < g_f$:
    Define $p(\cdot|\boldsymbol{\eta}^{(i+1,j,k)})$ using $\mathbf{\Sigma}_{i+1} = \mathbf{S}_i + c\mathbf{I}$
    for $j = 1, \cdots, p_0^{-1}$:
        for $k = 1, \cdots, N \cdot p_0$:
            Draw a candidate state $\boldsymbol{\tau}$ using $p(\cdot|\boldsymbol{\eta}^{(i,j,k-1)})$
            With probability $\min\left(1, \alpha = \left[\prod_{k=1}^{n} \frac{\pi_k(\theta_j^*(k))}{\pi_k(\theta_j^{(i)}(k))}\right] \cdot \frac{p(\theta^{(i)}|\theta^*)}{p(\theta^*|\theta^{(i)})}\right)$:
            set $\boldsymbol{\eta}^* = \tau$
            Otherwise: set $\boldsymbol{\eta}^* = \boldsymbol{\eta}^{(i,j,k-1)}$
            Set $\boldsymbol{\eta}^{(i,j,k)} = \begin{cases} \boldsymbol{\eta}^* & \text{if } \boldsymbol{\eta}^* \in F_i \\ \boldsymbol{\eta}^{(i,j,k-1)} & \text{otherwise} \end{cases}$
        end
    end
    Set $i = i + 1$
    Recover the $\boldsymbol{\theta}^{(i-1,j,k)}$ by setting $\theta_l^{(i-1,j,k)} = \frac{\eta_l^{(i-1,j,k)}}{1+\eta_l^{(i-1,j,k)}}$ for all $l$
    Set new intermediate threshold $g_{t_i}$ equal to the $100 \cdot p_0$ percentile of
    the values $\{g(\boldsymbol{\theta}^{(i-1,j,k)})\}$
    Set the seeds of $F_i$ as those $\boldsymbol{\theta} = \boldsymbol{\theta}^{(i-1,j,k)}$ s.t. $g(\boldsymbol{\theta}) \geq g_{t_i}$. Label them
    $\{\boldsymbol{\theta}^{(i,j,0)}\}$, $j = 1, \cdots, p_0^{-1}$.
end
```

In the problems under consideration here, we have that $\pi_k$ is uniform for all $k$. The proposal distribution $p(\cdot|\theta)$ is a normal distribution of the logit-transformed $\eta$, with $\eta(k) = \log\left(\frac{\theta(k)}{1-\theta(k)}\right)$. Thus the Metropolis-Hastings acceptance ratio is given by

$$\alpha = \left[\prod_{k=1}^{n} \frac{\pi_k(\theta_j^*(k))}{\pi_k(\theta_j^{(i)}(k))}\right] \cdot \frac{p(\theta_j^{(i)}|\theta_j^*)}{p(\theta_j^*|\theta_j^{(i)})} = \frac{p(\theta_j^{(i)}|\theta_j^*)}{p(\theta_j^*|\theta_j^{(i)})} = \prod_{k=1}^{n} \frac{\theta_j^*(k)(1-\theta_j^*(k))}{\theta_j^{(i)}(k)(1-\theta_j^{(i)}(k))}$$

# 4 Example: hyperellipsoid

MMA and MVD may be compared using a simple example in which the parameter space is taken to be a $1000-$dimensional hypercube extending from $-1$ to 1 in each dimension, where the failure region $F$ describes a small hyperellipsoid in four of the dimensions of the space and fills the space in all other dimensions. (A $1000-$dimensional parameter space is used instead of a $4-$dimensional space so that the problem can exemplify high dimensionality.) We introduce a high level of correlation amongst the parameters by appropriately tuning the lengths of the semi axes of the hyperellipsoid. In order that the correlation be amongst many of the 1000 components rather than merely amongst 4 of them, a rotation matrix is used to turn the hyperellipsoid so that its semi axes do not align with any particular coordinate axis of the parameter space, or indeed with any linear combination of a small number of coordinate axes. $F$ can be thought of as a higher-dimensional version of, say, a cylinder in $\mathbb{R}^3$, the two faces of which are $2-$dimensional ellipses. $F$ is then a $1000-$dimensional “cylinder”, the faces of which are $4-$dimensional hyperellipsoids.

To accomplish this, define a performance function $t : \mathbb{R}^{1000} \to \mathbb{R}$ as follows. First, set $Q$ to be a given rotation matrix in $\mathbb{R}^{1000}$, and define $h : \mathbb{R}^4 \to \mathbb{R}$ as

$$h(z) = \sum_{i=1}^{4} c \cdot i^4 \cdot z_i^2$$

where $c$ is a constant. Finally, let

$$t(x) = h\big((Qx)_{200}, (Qx)_{400}, (Qx)_{600}, (Qx)_{800}\big).$$

In other words: $t$ rotates $x$ using $Q$ and then, using four of the dimensions of the rotated point, finds on which level curve of $h$ the corresponding four-dimensional point lies.

The resulting performance function allows us to define a failure region with significant correlation amongst the parameters. Setting $c = 2.314$ sets the resulting $4-$dimensional hyperellipsoid to have volume $10^{-4}$. The volume of this hyperellipsoid is invariant under rotation, since it is contained entirely in the parameter space under any rotation. However, recall that $F$ is not itself this $4-$dimensional hyperellipsoid, but is rather a "cylinder" with hyperellipsoidal faces, and rotation changes how much of that cylinder is contained within the hypercube parameter space. Though it is possible to find the volume of $F$ analytically, it is much simpler to do so either via direct Monte Carlo simulation or via subset simulation. Since the latter is under investigation here, direct Monte Carlo simulation is used to establish the volume of $F$. Drawing $10^7$ samples from the parameter space, 3228 were found to be within $F$. Thus we may estimate the volume of $F$ to be $3.228 \cdot 10^{-4}$. (And therefore, under a uniform distribution across the parameter space, $F$ considered as an event has probability $3.228 \cdot 10^{-4}$.) Drawing these $10^7$ samples required 7090.91 seconds of runtime on a Windows computer with a 2.5GHz Intel Core i7-6500U CPU and 8GB of RAM[4].

The 4-D hyperellipsoid described by $F$ has its longest semiaxis with length $1/\sqrt{c}$; its shortest semiaxis is $1/(16\sqrt{c})$. Thus there is appreciable correlation amongst the parameters with respect to the failure region, such that varying the location of a point in $F$ in the dimension of that shortest semiaxis is quite likely to result in the point leaving $F$, whereas moving in the dimension of the longest semiaxis is far less likely to result in an exit from $F$.

## 4.1 Comparison: MVD vs. MMA

MMA and MVD are both used to sample from $F$, and the results are compared below. For reproducibility, all the trials of MMA and MVD described in this section are preceded by setting the random seed of MATLAB to 1, via `rng(1)`.

For MMA, the total time required to successfully complete the algorithm, and thereby sample from the target failure region $F$, was 2714.73 seconds. We may produce a point estimate of $P(F)$ as follows. The algorithm was run with $p_0 = 0.1$, so that $P(F_{i+1}|F_i) \approx 0.1$ for each pair of adjacent intermediate failure regions $F_i, F_{i+1}$. A total of four intermediate failure regions were required (with $F_0$ being the entire parameter space). The algorithm was run with $N$ set to equal 5000. Therefore, 5000 samples were drawn from each intermediate failure region $F_i$, $i = 0, 1, 2, 3$. The number of samples in each intermediate failure region $F_i$ that were found also to be in $F$ itself, for $i = 0, 1, 2, 3$, are given in Table 1.

| $F_0$ | $F_1$ | $F_2$ | $F_3$ |
|---|---|---|---|
| 1 | 17 | 124 | 1281 |

Table 1: Samples in $F$ using MMA with $N = 5000$.

For each $F_i$ from which samples are drawn, where $N_f$ is the number of samples found to be in $F$ itself, we have that $N_f/N$ is a point estimate of $P(F|F_i)$. Using the fact that $P(F_i|F_{i-1}) \approx p_0 = 0.1$, then, we have by (1):

$$P(F) = P(F|F_i)P(F_i) \approx P(F|F_i)p_0^i \approx \frac{N_f}{N}p_0^i = \frac{N_f}{N} \cdot 10^{-i}$$

Since we have one such point estimate of $P(F)$ for each intermediate failure region from which we drew, we may arrive at a final point estimate of $P(F)$ by

[4] All subsequent reported runtimes are from the same machine.

averaging them:

$$P(F) \approx \left( \frac{1}{5000} \cdot p_0^0 + \frac{17}{5000} \cdot p_0^1 + \frac{124}{5000} \cdot p_0^2 + \frac{1281}{5000} \cdot p_0^3 \right) /4 \approx 2.6105 \cdot 10^{-4}$$

Though the difference between the estimate found above via direct Monte Carlo simulation and that found via MMA is less than $10^{-4}$, nonetheless the MMA estimate is just over $\frac{4}{5}$ the former. Though not a perfect estimate, it must be kept in mind that it is made using only $20,000$ samples to the direct Monte Carlo's $10^7$. However, the distance between the two estimates may be due to the fact that MMA's acceptance rate for samples becomes quite low by the last loop of the algorithm. Recall that in the modified Metropolis algorithms represented by MMA and MVD, a candidate draw is accepted only with probability $\alpha$ (defined above in Sections 2 and 3 for MMA and MVD respectively), and that a candidate draw may furthermore be rejected if it fails to fall within the current intermediate failure region. If a candidate draw is rejected, then the next sample in the MCMC chain merely copies the previous one. Therefore, when the rejection rate is high, the samples drawn may represent few unique points from the support of the target distribution $\boldsymbol{\theta}|F$. Here, of the 5000 samples drawn in the final step of the algorithm, only 68 unique points are represented; that is, in the MCMC chains, the candidate draw was rejected 4932 out of 5000 times.

By contrast, using MVD, the total runtime required to sample from the target failure region $F$ was only 324.52 seconds—an improvement of over 88%. Furthermore, the samples drawn in the last step of the algorithm included 1641 distinct points, in contrast to the 68 found when MMA was used. Again four total intermediate failure regions were required. Of the 5000 samples taken in each intermediate failure region, the number falling in $F$ itself are given in Table 2.

| $F_0$ | $F_1$ | $F_2$ | $F_3$ |
|---|---|---|---|
| 1 | 23 | 196 | 1937 |

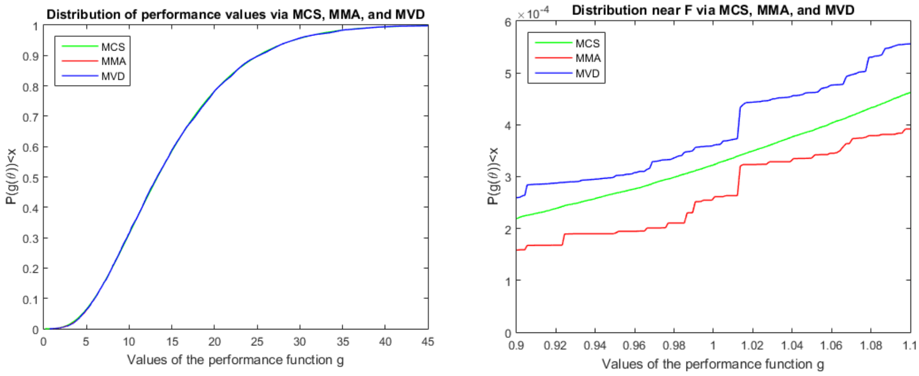
Table 2: Samples in $F$ using MVD with $N = 5000$.

Therefore we may estimate:

$$P(F) \approx \left( \frac{1}{5000} \cdot p_0^0 + \frac{23}{5000} \cdot p_0^1 + \frac{196}{5000} \cdot p_0^2 + \frac{1937}{5000} \cdot p_0^3 \right) /4 \approx 3.5985 \cdot 10^{-4}$$

This is closer to the direct Monte Carlo simulation estimate of $3.228 \cdot 10^{-4}$ than that found via MMA. The accuracy here has improved by 40% in comparison with $MMA$.

The probability curve estimates resulting from each of direct Monte Carlo simulation, MMA and MVD are given in Figure 3. These curves are found using the same procedure as that used to find $P(F)$ above. That is: for any value $g_v$ of the performance function, we may estimate $P(g_v)$ by taking the mean of the proportions $(p_0)^i \cdot N_{i,v}/N$ from each intermediate failure region $F_i$ such that $g_{t_i} \geq g_v$, where $N_{i,v}$ is the number of samples $\boldsymbol{\theta}$ in step $i$ such that $g(\boldsymbol{\theta}) \leq g_v$. Notice that MVD and MMA cannot be distinguished except near performance value 1. This is because for performance values much greater than 1, the estimate of the curve is based solely on the initial $N$ samples generated



(a) Distribution over all performance values up to 45.

(b) Distribution over performance values near 1.

Figure 3: Distribution of $g(\theta)$, the values of the performance function, under each of the three simulation methods. The left panel shows the entire distribution up to a performance value of 45. The right panel zooms in to the low-probability event of a performance value near 1, which is the target threshold of the failure region $F$.

via Monte Carlo simulation which both MMA and MVD require. Here, those two sets of $N$ Monte Carlo samples are identical, because each algorithm was preceded by setting `rng(1)`.

Notice also that the estimate is quite bad for performance values much greater than 1, which is neither unexpected nor troubling. The nature of subset simulation is to target a specific low-probability region of the parameter space. The higher the probability of a region $F$ of the parameter space, the fewer intermediate failure regions will contain it, and so the fewer samples will be drawn that can be used to estimate the probability of that region. For example, if $F$ had probability $0.2 = 2 \cdot p_0 = 0.1$, then both MMA and MVD would terminate immediately after their initial Monte Carlo draw. In this case, both MMA and MVD would be equivalent to a direct Monte Carlo draw of $N$ samples. Indeed, the entire point of subset simulation is to improve efficiency in sampling from a low-probability $F$ specifically by refraining (as much as possible) from sampling from other regions of the parameter space.

# 5 Example: Business downtime following seismic hazard

An application in which MMA and MVD may be compared is given by Prabhu et al. [8]. Here, a fault tree analysis of the components of a power plant is used to induce a performance function which outputs downtime following an earthquake. A fault tree analysis describes the relationship amongst components of a system with respect to failure propagation, using logic gates. For example, a higher-level system component $\Xi$ may comprise several lower-level components $\xi_i$, $i = 1, \cdots, m$, to which it is connected via an OR gate, meaning that the failure of any one $\xi_i$ suffices for the failure of $\Xi$. Or $\Xi$ might be connected to the

$\xi_i$ via an AND gate, meaning that all $\xi_i$ must fail in order for $\Xi$ to fail. Other logic gates, more complex relationships, and nested hierarchies are employed as well; the resulting model may be used for reliability analysis of the total system.

In the example considered here, the performance function is fairly computationally expensive, requiring roughly one tenth of a second to evaluate a point. As a result, for a low-probability failure region $F$, direct Monte Carlo sampling is inefficient for sampling from $F$. Instead, the use of subset simulation is more desirable.

In this application, the failure region is defined as being the event that downtime is under 62.5 days. Drawing 400,000 samples from the parameter space via direct Monte Carlo simulation yields 22 samples in this failure region $F$, so the probability of $F$ is roughly $5.5 \cdot 10^{-5}$. Drawing these 22 samples from $F$ via direct Monte Carlo simulation required 36,272.83 seconds of runtime.

## 5.1 Comparison: MVD vs. MMA

Again for reproducibility, each of the trials discussed here was immediately preceded by setting `rng(1)`. Employing MMA with $N = 1000$, the total runtime required to sample from the target failure region $F$ is 1612.92 seconds. The algorithm required a total of five intermediate failure regions, including $F_0$ as the entire parameter space. In each of the four intermediate failure regions, the number of samples found to be in $F$ itself are given in Table 3. Using the same technique for point estimation described in Section 4, we have via MMA that $P(F) \approx 3.218 \cdot 10^{-5}$. Recall that the estimate via direct Monte Carlo simulation is $5.5 \cdot 10^{-5}$. Though the estimate is just over half that found via direct Monte Carlo, it must again be remembered that MMA achieves this estimate using a mere 5000 samples; its accuracy is far greater than what could possibly be achieved by drawing the same number of samples in direct Monte Carlo.

| $F_0$ | $F_1$ | $F_2$ | $F_3$ | $F4$ |
|---|---|---|---|---|
| 0 | 0 | 7 | 48 | 429 |

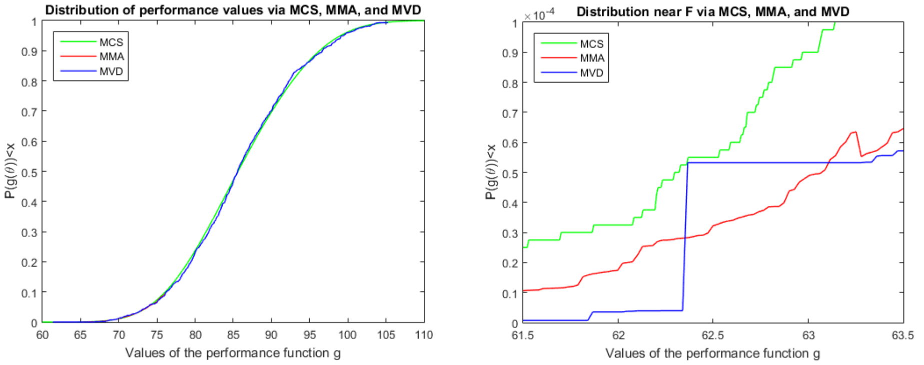
Table 3: Samples in $F$ using MMA with $N = 1000$.

MVD required only 336.61 seconds of runtime to sample from $F$; an improvement of over 79% over MMA's runtime of 1612.92 seconds. In each of the four intermediate failure regions, the number of samples found to be in $F$ are given in Table 4. These values gives us $P(F) \approx 5.322 \cdot 10^{-5}$. This is an improvement of over 92% in approximating the estimate generated via direct Monte Carlo simulation, of $5.5 \cdot 10^{-5}$.

| $F_0$ | $F_1$ | $F_2$ | $F_3$ | $F4$ |
|---|---|---|---|---|
| 0 | 0 | 9 | 89 | 871 |

Table 4: Samples in $F$ using MVD with $N = 1000$.

The probability curve estimates resulting from each of direct Monte Carlo simulation, MMA and MVD are given in Figure 4. The curves shown are calculated in the way described above in Section 4. Notice that, as in the case of Figure 3, MVD and MMA cannot be distinguished except near performance



(a) Distribution over all performance values from 60 to 110.

(b) Distribution over performance values near 62.5.

Figure 4: Distribution of performance values under each of the three simulation methods for business interruption after seismic hazard. The left panel shows the entire distribution up to a performance value of 110 days. The right panel zooms in to the low-probability event of a performance value near 62.5.

value 62.5. This is again because for performance values much greater than the target threshold of $F$ 62.5, the estimate of the curve is based on an identical sample set of size 1000, generated by direct Monte Carlo simulation.

# 6 Conclusion

Subset simulation is more tractable when the parameters are independent of one another [2]. However, even in the case of independence, it may occur that the parameters are correlated with respect to a target failure region of interest. When this is known (as in the ellipsoid example) or suspected (as in the seismic failure example) to be the case, a multivariate draw algorithm with an adaptive covariance matrix can improve the efficiency of subset simulation. Furthermore, boundary conditions for parameters can be accommodated by employing a logit transform, and (if employing e.g., a normal proposal density on the logit-transformed parameters) making use of the Metropolis-Hastings rather than the Metropolis algorithm.

The method of splitting developed by Ching, Au and Beck [3] utilizes direct Monte Carlo simulation rather than MCMC. Since the current proposal is a version of an MCMC approach, it cannot be used in conjunction with splitting. However, the present project is fully compatible with the modifications to subset simulation proposed by Zuev et al. [9] and by Papaioannou et al. [7]. Though the current project finds only a point estimate of $P(F)$, it is fully compatible with the Bayesian post-processing methodology used by Zuev et al. to generate instead a probability distribution on $P(F)$.

Like the technique suggested by Papaioannou et al., the current project also involves an adaptive proposal distribution. However, here no attempt is made to tune specifically the scale of the variance of the proposal distribution, as is done in their approach. There is nonetheless no reason that such an adaptive scaling of the magnitude of the proposal variance could not be combined with the current approach; and indeed it seems likely that doing so would improve the efficiency of the algorithm.

# References

[1] Rachid Ababou, Amvrossios C. Bagtzoglou, and Eric F. Wood. “On the condition number of covariance matrices in kriging, estimation, and simulation of random fields”. In: *Mathematical Geology* 26.1 (1994), pp. 99–133. ISSN: 0882-8121. DOI: 10.1007/BF02065878. URL: http://link.springer.com/10.1007/BF02065878.

[2] Siu Kui Au and James L. Beck. “Estimation of small failure probabilities in high dimensions by subset simulation”. In: *Probabilistic Engineering Mechanics* 16.4 (2001), pp. 263–277.

[3] J. Ching, S.K. Au, and J.L. Beck. “Reliability estimation for dynamical systems subject to stochastic excitation using subset simulation with splitting”. In: *Computer Methods in Applied Mechanics and Engineering* 194.12-16 (2005), pp. 1557–1579. ISSN: 00457825. DOI: 10.1016/j.cma.2004.05.028. URL: http://linkinghub.elsevier.com/retrieve/pii/S0045782504004050.

[4] Heikki Haario, Eero Saksman, and Johanna Tamminen. “An adaptive Metropolis algorithm”. In: *Bernoulli* 7.2 (2001), p. 223. ISSN: 13507265. DOI: 10.2307/3318737. URL: http://www.jstor.org/stable/3318737?origin=crossref.

[5] W. K. Hastings. “Monte Carlo sampling methods using Markov chains and their applications”. In: *Biometrika* 57.1 (1970), p. 97. ISSN: 00063444. DOI: 10.2307/2334940. URL: http://www.jstor.org/stable/2334940?origin=crossref.

[6] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. “Equation of state calculations by fast computing machines”. In: *The Journal of Chemical Physics* 21.6 (1953), pp. 1087–1092. ISSN: 0021-9606. DOI: 10.1063/1.1699114. URL: http://aip.scitation.org/doi/10.1063/1.1699114.

[7] Iason Papaioannou, Wolfgang Betz, Kilian Zwirglmaier, and Daniel Straub. “MCMC algorithms for subset simulation”. In: *Probabilistic Engineering Mechanics* 41 (2015), pp. 89–103. ISSN: 02668920. DOI: 10.1016/j.probengmech.2015.06.006. URL: http://linkinghub.elsevier.com/retrieve/pii/S0266892015300205.

[8] Saurabh Prabhu, Carl Ehrett, D. Andrew Brown, Mohammad Javanbarg, Sez Atamturktur, and Marc Lehmann. “Uncertainty quantification in fault tree analysis: estimating business interruption due to seismic hazard”. In: *In progress* (2017).

[9] Konstantin M. Zuev, James L. Beck, Siu-Kui Au, and Lambros S. Katafygiotis. “Bayesian post-processor and other enhancements of subset simulation for estimating failure probabilities in high dimensions”. In: *Computers & Structures* 92-93 (2012), pp. 283–296. ISSN: 00457949. DOI: 10.1016/j.compstruc.2011.10.017. URL: http://linkinghub.elsevier.com/retrieve/pii/S0045794911002720.