Bayesian subset simulation

Julien Bect, Ling Li, Emmanuel Vazquez

SIAM/ASA J. UNCERTAINTY QUANTIFICATION
Vol. 5, pp. 762–786

Bayesian Subset Simulation*

Julien Bect†, Ling Li‡, and Emmanuel Vazquez†

Abstract. We consider the problem of estimating a probability of failure α, defined as the volume of the excursion set of a function $f : \mathbb{X} \subseteq \mathbb{R}^d \to \mathbb{R}$ above a given threshold, under a given probability measure on $\mathbb{X}$. In this paper, we combine the popular subset simulation algorithm [S.-K. Au and J. L. Beck, *Probab. Eng. Mech.*, 16 (2001), pp. 263–277] and our sequential Bayesian approach for the estimation of a probability of failure [J. Bect et al., *Stat. Comput.*, 22 (2012), pp. 773–793]. This makes it possible to estimate α when the number of evaluations of f is very limited and α is very small. The resulting algorithm is called Bayesian subset simulation (BSS). A key idea, as in the subset simulation algorithm, is to estimate the probabilities of a sequence of excursion sets of f above intermediate thresholds, using a sequential Monte Carlo (SMC) approach. A Gaussian process prior on f is used to define the sequence of densities targeted by the SMC algorithm and drive the selection of evaluation points of f to estimate the intermediate probabilities. Adaptive procedures are proposed to determine the intermediate thresholds and the number of evaluations to be carried out at each stage of the algorithm. Numerical experiments illustrate that BSS achieves significant savings in the number of function evaluations with respect to other Monte Carlo approaches.

Key words. probability of failure, computer experiments, sequential design, Gaussian process, stepwise uncertainty reduction, sequential Monte Carlo

AMS subject classifications. 62L05, 62K99, 62P30

DOI. 10.1137/16M1078276

1. Introduction.

Probabilistic reliability analysis has become over the last 30 years an essential part of the engineer's toolbox (see, e.g., [19, 44, 47]). One of the central problems in probabilistic reliability analysis is the computation of the probability of failure

$$\alpha = \int_{\mathbb{X}} \mathbb{1}_{f \le 0} \, \mathrm{d}\mathsf{P}_{\mathbb{X}} \tag{1}$$

of a system (or a component in a multicomponent system; see, e.g., [48]), where $\mathsf{P}_{\mathbb{X}}$ is a probability measure over some measurable space $(\mathbb{X}, \mathcal{B})$ representing all possible sources of uncertainty acting on the system—both epistemic and aleatory—and $f : \mathbb{X} \to \mathbb{R}$ is the so-called *limit-state function*, such that f takes positive values when the system behaves reliably, and negative values when the system behaves unreliably or fails. It is assumed in this paper that $\mathbb{X}$

*Received by the editors June 2, 2016; accepted for publication (in revised form) April 6, 2017; published electronically August 8, 2017. Parts of this work were previously published in *Proceedings of the PSAM* 11 *and ESREL* 2012 *Conference on Probabilistic Safety Assessment* [41] and in the Ph.D. thesis of the second author [40].
http://www.siam.org/journals/juq/5/M107827.html
Funding: This research was partially funded by the French Fond Unique Interministériel (FUI 7) in the context of the CSDL (Complex Systems Design Lab) project.

†Laboratoire des Signaux et Systèmes, CentraleSupélec, CNRS, Université Paris-Sud, Université Paris-Saclay, 3 rue Joliot-Curie, 92192 Gif-sur-Yvette, France (julien.bect@centralesupelec.fr, emmanuel.vazquez@centralesupelec.fr).
‡Schlumberger Technology Centre, Stonehouse, GL10 3SX, UK (ling.li.supelec@gmail.com).

is a subset of $\mathbb{R}^d$—in other words, we consider reliability problems where all uncertain factors can be described as a d-dimensional random vector. Numerous examples of applications that fall into this category can be found in the literature (see, for instance, [5, 18, 34, 39, 54, 55]).

Two major difficulties usually preclude a brute force Monte Carlo (MC) approach, that is, using the estimator

$$\widehat{\alpha}^{\mathrm{MC}} = \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}_{f(X_i)\le 0}\,, \quad X_i \overset{\text{i.i.d.}}{\sim} \mathsf{P}_{\mathbb{X}}\,,$$

which requires m evaluations of f. First, the evaluation of f for a given $x \in \mathbb{X}$ often relies on one or several complex computer programs (e.g., partial differential equation solvers) that take a long time to run. Second, in many applications, the failure region $\Gamma = \{x \in \mathbb{X} \mid f(x) \le 0\}$ is a *rare event* under the probability $\mathsf{P}_{\mathbb{X}}$; that is, the probability of failure $\alpha = \mathsf{P}_{\mathbb{X}}(\Gamma)$ is small. When α is small, the standard deviation of $\widehat{\alpha}^{\mathrm{MC}}$ is approximately $\sqrt{\alpha/m}$. To estimate α by Monte Carlo with a standard deviation of 0.1α thus requires approximately $100/\alpha$ evaluations of f. As an example, with $\alpha = 10^{-3}$ and 10 minutes per evaluation, this means almost two years of computation time.

The first issue—designing efficient algorithms to estimate α in the case of an expensive-to-evaluate limit-state function—can be seen as a problem of *design and analysis of computer experiments* (see, e.g., [50]), bearing some similarities to the problem of global optimization (see [53] and references therein). Several sequential design strategies based on Gaussian process models have been proposed in the literature, and spectacular evaluation savings have been demonstrated on various examples with moderately small α (typically, 10^{-2} or 10^{-3}); see [6] for a review of fully sequential strategies and [3, 27] for examples of two-stage strategies. The closely related problem of quantile estimation has also been investigated along similar lines [1, 13, 46].

A key idea in addressing the second issue—i.e., in estimating a small probability of failure—is to consider a decreasing sequence of events $\Gamma_1 \supset \Gamma_2 \supset \cdots \supset \Gamma_T = \Gamma$ such that the conditional probabilities $\mathsf{P}_{\mathbb{X}}(\Gamma_t \mid \Gamma_{t-1})$ are reasonably large and therefore easier to estimate than α itself. Then, sequential Monte Carlo (SMC) simulations [21] can be used to produce estimates $\widehat{p}_t$ of the conditional probabilities $\mathsf{P}_{\mathbb{X}}(\Gamma_t \mid \Gamma_{t-1})$, leading to a product-form estimate $\prod_{t=1}^{T} \widehat{p}_t$ for α. This idea, called *subset simulation*, was first proposed in [2] for the simulation of rare events in structural reliability analysis,[1] but actually goes back to the much older *importance splitting* (or *multilevel splitting*) technique used for the simulation of rare events in Markovian models (see, e.g., [36] and references therein). Subset simulation has since become one of the most popular techniques for the computation of small probabilities of failure, and the theoretical properties of several (most of the time idealized) variants of the algorithm have recently been investigated by several authors (see, e.g., [11, 14]). However, because of the direct use of a Monte Carlo estimator for $\widehat{p}_t$ at each stage t, the subset simulation algorithm is not applicable when f is expensive to evaluate.

In this paper we propose a new algorithm, called *Bayesian subset simulation* (BSS), which tackles both issues at once using ideas from the sequential design of computer experiments

[1] A very similar algorithm had, in fact, been proposed earlier by [23], but for a quite different purpose (estimating the probability of a rare event under the bootstrap distribution).

and from the literature on SMC methods. Section 2 reviews the subset simulation algorithm from the point of view of SMC techniques to pave the way for the introduction of our new algorithm. Section 3 describes the algorithm itself, and section 4 presents numerical results. Finally, section 5 concludes the paper with a discussion.

2. Subset simulation: A sequential Monte Carlo algorithm.

This section recalls the main ideas of the classical subset simulation algorithm [2], which, although not originally presented as such, can be seen as an SMC sampler [14, 21].

2.1. Idealized subset simulation (with fixed levels and independent and identically distributed (i.i.d.) sampling).

We consider the problem of estimating the probability α of a rare event Γ of the form $\Gamma = \{x \in \mathbb{X} : f(x) > u\}$, where $u \in \mathbb{R}$ and $f : \mathbb{X} \to \mathbb{R}$, using pointwise evaluations of f. Note that the limit-state function (see section 1) can be defined as $x \mapsto u - f(x)$ with our notation. Assuming, for the sake of simplicity, that $\mathsf{P}_{\mathbb{X}}$ has a probability density function $\pi_{\mathbb{X}}$ with respect to Lebesgue's measure, we have

$$\alpha = \int_{\mathbb{X}} \mathbb{1}_{f(x)>u}\, \pi_{\mathbb{X}}(x)\, \mathrm{d}x\,.$$

The key idea of the subset simulation algorithm is to introduce an increasing (finite) sequence of thresholds $-\infty = u_0 < u_1 < u_2 \cdots < u_T = u$, which determine a corresponding decreasing sequence of subsets,

$$\mathbb{X} = \Gamma_0 \supset \Gamma_1 \supset \cdots \supset \Gamma_T = \Gamma, \quad \Gamma_t \coloneqq \{x \in \mathbb{X} : f(x) > u_t\}\,,$$

of the input space $\mathbb{X}$. Let $\alpha_t = \mathsf{P}_{\mathbb{X}}(\Gamma_t)$. The decreasing sequence $(\alpha_t)_{0\le t\le T}$ obeys the recurrence formula

$$\alpha_{t+1} = \alpha_t\, \mathsf{P}_{\mathbb{X}}(\Gamma_{t+1} \mid \Gamma_t) = \alpha_t \int \mathbb{1}_{\Gamma_{t+1}}(x)\, q_t(x)\, \mathrm{d}x, \tag{2}$$

where q_t stands for the truncated density

$$q_t(x) = \frac{\mathbb{1}_{\Gamma_t}(x)\, \pi_{\mathbb{X}}(x)}{\int \mathbb{1}_{\Gamma_t}(y)\, \pi_{\mathbb{X}}(y)\, \mathrm{d}y}\,. \tag{3}$$

The small probability $\alpha = \alpha_T$ can thus be rewritten as a product of conditional probabilities, which are larger (and therefore easier to estimate) than α:

$$\alpha = \prod_{t=1}^{T} p_t, \quad p_t \coloneqq \mathsf{P}_{\mathbb{X}}(\Gamma_t \mid \Gamma_{t-1})\,.$$

Assume that for each $t \in \{0, 1, \ldots, T-1\}$, a sample $(Y_t^j)_{1\le j\le m}$ of i.i.d. random variables from the truncated density q_t is available. Then, each conditional probability p_t can be estimated by the corresponding Monte Carlo estimator $\widehat{p}_t = \frac{1}{m}\sum_{j=1}^{m} \mathbb{1}_{\Gamma_t}\big(Y_{t-1}^j\big)$, and α can be estimated by the product-form estimator $\widehat{\alpha}^{\mathrm{SS}} = \prod_{t=1}^{T} \widehat{p}_t$. By choosing the thresholds u_t in such a way that the conditional probabilities p_t are high, α can be estimated using fewer evaluations of f than what would have been necessary using a simple Monte Carlo approach (see section 2.4 for a quantitative example).

2.2. Sequential Monte Carlo simulation techniques. Generating exact i.i.d. draws from the densities q_t is usually not possible, at least not efficiently, even if a method to generate i.i.d. samples from $q_0 = \pi_{\mathbb{X}}$ is available. Indeed, although the accept-reject algorithm (see, e.g., [49, section 2.3]) could be used in principle, it would be extremely inefficient when t is close to T, that is, when $\mathsf{P}_{\mathbb{X}}\{\Gamma_t\}$ becomes small. This is where SMC simulation techniques are useful.

Given a sequence $(q_t)_{0\le t<T}$ of probability density functions over $\mathbb{X}$, SMC samplers sequentially generate, for each target density q_t, a weighted sample $\mathbb{Y}_t = ((w_t^j, Y_t^j))_{1\le j\le m}$, where $w_t^j \ge 0$, $\sum_j w_t^j = 1$, and $Y_t^j \in \mathbb{X}$. The random vectors Y_t^j are usually called *particles* in the SMC literature, and the weighted sample $\mathbb{Y}_t$ is said to *target* the distribution q_t. The particles are, in general, neither independent nor distributed according to q_t, but when the sample size m goes to infinity, their empirical distribution $\mu_t^{(m)} = \sum_{j=1}^m w_t^j \, \delta_{Y_t^j}$ converges to the target distribution—that is, to the distribution with probability density function q_t—in the sense that

$$\int_{\mathbb{X}} h(x)\,\mathrm{d}\mu_t^{(m)}(x) = \sum_{j=1}^m w_t^j\, h(Y_t^j) \;\to\; \int_{\mathbb{X}} h(x)\, q_t(x)\,\mathrm{d}x$$

for a certain class of integrable functions h.

In practice, each weighted sample $\mathbb{Y}_t$ is generated from the previous one, $\mathbb{Y}_{t-1}$, using transformations; SMC algorithms are thus expected to be efficient when each density q_t is, in some sense, close to its predecessor density q_{t-1}. The specific transformations that are used in the subset simulation algorithm are described next. The reader is referred to [21, 42] and references therein for a broader view of SMC sampling techniques, and to [25] for some theoretical results on the convergence (law of large numbers, central limit theorems) of SMC algorithms.

2.3. Reweight/resample/move. We now describe the reweight/resample/move scheme that is used in the subset simulation algorithm to turn a weighted sample $\mathbb{Y}_{t-1}$ targeting $q_{t-1} \propto \mathbb{1}_{\Gamma_{t-1}}\pi_{\mathbb{X}}$ into a weighted sample $\mathbb{Y}_t$ targeting $q_t \propto \mathbb{1}_{\Gamma_t}\pi_{\mathbb{X}}$. This scheme, used, for instance in [17], can be seen as a special case of the more general SMC sampler of [21].[2]

Assume a weighted sample $\mathbb{Y}_{t-1} = ((w_{t-1}^j, Y_{t-1}^j))_{1\le j\le m}$ targeting q_{t-1} has been obtained at stage $t-1$. The *reweight* step produces a new weighted sample $\mathbb{Y}_{t,0} = ((w_{t,0}^j, Y_{t-1}^j))_{1\le j\le m}$ that targets q_t by changing only the weights in $\mathbb{Y}_{t-1}$:

$$w_{t,0}^j \propto \frac{q_t\big(Y_{t-1}^j\big)}{q_{t-1}\big(Y_{t-1}^j\big)}\, w_{t-1}^j.$$

The *resample* and *move* steps follow the reweighting step. These steps aim at avoiding the degeneracy of the sequence of weighted samples—i.e., the accumulation of most of the probability mass on a small number of particles with large weights.

The simplest variant of resampling is the *multinomial resampling* scheme. It produces a new weighted sample $\mathbb{Y}_{t,1} = ((w_t^j, Y_{t,1}^j))_{1\le j\le m}$, where the new particles $Y_{t,1}^j$ have equal weights

[2] See in particular section 3.1.1, Remark 1, and section 3.3.2.3.

$w_t^j = \frac{1}{m}$ and are i.i.d. according to the empirical distribution $\sum_{j=1}^m w_{t,0}^j \, \delta_{Y_{t-1}^j}$. In this work, we use the slightly more elaborate *residual resampling* scheme (see, e.g., [42]), which is known to outperform multinomial resampling [24, section 3.2]. As in multinomial resampling, the residual resampling scheme produces a weighted sample with equal weights $w_t^j = \frac{1}{m}$.

The resampling step alone does not prevent degeneracy, since the resulting sample contains copies of the same particles. The move step restores some diversity by moving the particles according to a Markov transition kernel K_t (for instance, a random walk Metropolis–Hastings kernel; see, e.g., [49]) that leaves q_t invariant:

$$\int q_t(x) \, K_t(x, \mathrm{d}x') = q_t(x') \, \mathrm{d}x'.$$

Remark 1. In the special case of the subset simulation algorithm, all weights are actually equal *before* the reweighting step, and, considering the inclusion $\Gamma_t \subset \Gamma_{t-1}$, the reweighting formula takes the form

$$w_{t,0}^j \propto \mathbb{1}_{\Gamma_t}(Y_{t-1}^j).$$

In other words, the particles that are outside the new subset Γ_t are given a zero weight, and the other weights are simply normalized to sum to one. Note also that the resampling step discards particles outside of Γ_t (those with zero weight at the reweighting step).

Remark 2. Note that Au and Beck's original algorithm [2] does not use separate resample/move steps as described in this section. Instead, it uses a slightly different (but essentially similar) sampling scheme to populate each level: assuming that $L_t = m/m_t$ is an integer, where m_t denotes the number of particles from stage $t-1$ that belong to Γ_t, they start m_t independent Markov chains of length L_t from each of the particles (called "seeds"). Both variants of the algorithm have the property, in the case of fixed levels, that the particles produced at level t are exactly distributed according to q_t.

Remark 3. In the general version of the reweight/resample/move procedure, the resampling step is carried out only when some degeneracy criterion—such as the effective sample size (ESS)—falls below a threshold (see, e.g., [21, 22]).

2.4. Practical subset simulation: Adaptive thresholds. It is easy to prove that the subset simulation estimator $\widehat{\alpha}^{\mathrm{SS}} = \prod_{t=1}^T \widehat{p}_t$ is unbiased. Moreover, according to Proposition 3 in [14], it is asymptotically normal in the large-sample-size limit:

$$\sqrt{m} \, \frac{\widehat{\alpha}^{\mathrm{SS}} - \alpha}{\alpha} \xrightarrow[m\to\infty]{\mathcal{D}} \mathcal{N}\left(0; \sigma^2\right), \tag{4}$$

where $\xrightarrow{\mathcal{D}}$ denotes convergence in distribution and

$$\sigma^2 \approx \sum_{t=1}^T \frac{1 - p_t}{p_t}, \tag{5}$$

when the Markov chain Monte Carlo (MCMC) kernel has good mixing properties (see [14] for the exact expression of σ^2). For a given number T of stages, the right-hand side of (5) is minimal when all conditional probabilities are equal, that is, when $p_t = \alpha^{1/T}$.

In practice, however, the value of α is, of course, unknown, and it is not possible to choose the sequence of threshold beforehand to make all the conditional probabilities equal. Instead, a value p_0 is chosen—say, $p_0 = 10\%$—and the thresholds are tuned in such a way that, at each stage t, $\widehat{p}_t = p_0$. A summary of the resulting algorithm is provided in Algorithm 1.

Equations (4) and (5) can be used to quantify the number of evaluations of f required to reach a given coefficient of variation with the subset simulation estimator $\widehat{\alpha}^{\mathrm{SS}}$. Indeed, in the case where all conditional probabilities are equal, we have

$$\operatorname{var}\left(\widehat{\alpha}^{\mathrm{SS}}/\alpha\right) \approx \frac{T}{m}\,\frac{1-p_0}{p_0}, \tag{6}$$

with $T = \log(\alpha)/\log(p_0)$. For example, take $\alpha = 10^{-6}$. With the simple Monte Carlo estimator, the number of evaluations of f is equal to the sample size m: approximately $n = \delta^{-2}\,\alpha^{-1} = 10^8$ evaluations are required to achieve a coefficient of variation $\delta = \operatorname{std}(\widehat{\alpha}^{\mathrm{MC}})/\alpha = 10\%$. In contrast, with $p_0 = 10\%$, the subset simulation algorithm will complete in $T = \log(\alpha)/\log(p_0) = 6$ stages, thus achieving a coefficient of variation $\delta = \operatorname{std}(\widehat{\alpha}^{\mathrm{SS}})/\alpha = 10\%$ with $m = \delta^{-2}\,T\,(1-p_0)/p_0 = 5400$ particles. Assuming that the move step uses only one evaluation of f per particle, the corresponding number of evaluations would be $n = m+(T-1)(1-p_0)m = 29700 \ll 10^8$.

Remark 4. The value $p_0 = 0.1$ was used in the original paper of Au and Beck, on the ground that it had been "found to yield good efficiency" [2, section 5]. Based on the approximate variance formula (6), Zuev et al. [56] argue that the variance is roughly proportional for a given total number of evaluations to $(1 - p_0)/(p_0\,(\log(p_0))^2)$, and conclude[3] that any $p_0 \in [0.1; 0.3]$ should yield quasi-optimal results for any α.

3. Bayesian subset simulation.

3.1. Bayesian estimation and sequential design of experiments. Our objective is to build an estimator of α from the evaluation results of f at some points $X_1, X_2, \ldots, X_N \in \mathbb{X}$, where N is the total budget of evaluations available for the estimation. In order to design an efficient estimation procedure, by which we mean both the design of experiments and the estimator itself, we adopt a Bayesian approach: from now on, the unknown function f is seen as a sample path of a random process ξ. In other words, the distribution of ξ is a *prior* about f. As in [6, 15, 52], the rationale for adopting a Bayesian viewpoint is to design a good estimation procedure in an average sense. This point of view has been largely explored in the literature of computer experiments (see, e.g., [50]) and that of Bayesian optimization (see [32] and references therein).

For the sake of tractability, we assume as usual that under the prior probability which we denote by P_0, ξ is a Gaussian process (possibly with a linearly parameterized mean, whose parameters are then endowed with a uniform improper prior; see [6, section 2.3] for details).

[3] Their analysis is based on the observation that the total number of evaluations is equal to $m\,T$— in other words, that m new samples must be produced at each stage. Some authors (e.g., [11]) consider a variant where the particles that come from the previous stage are simply copied to the new set of particles, untouched by the move step. In this case, a similar analysis suggests that (1) the optimal value of p_0 actually depends on α and is somewhere between 0.63 (for $\alpha = 0.01$) and 1.0 (when $\alpha \to 0$), and (2) the value of δ^2 is only weakly dependent on p_0, as long as p_0 is not too close to 0 (say, $p_0 \geq 0.1$).

Algorithm 1

Subset simulation algorithm with adaptive thresholds.

Prescribe $m_0 < m$ a fixed number of "succeeding particles." Set $p_0 = \frac{m_0}{m}$.

1. **Initialization** (stage 0)
   (a) Generate an m-sample $Y_0^j \overset{\text{i.i.d.}}{\sim} \mathsf{P}_{\mathbb{X}}$, $1 \leq j \leq m$, and evaluate $f\left(Y_0^j\right)$ for all j.
   (b) Set $u_0 = -\infty$ and $t = 1$.
2. **Repeat** (stage t)
   (a) **Threshold adaptation**
   - Compute the $(m - m_0)$th order statistic of $(f(Y_{t-1}^j))_{1 \leq j \leq m}$ and call it u_t^0.
   - If $u_t^0 > u$, set $u_t = u$, $T = t$ and go to the estimation step.
   - Otherwise, set $u_t = u_t^0$ and $\Gamma_t = \{x \in \mathbb{X}; f(x) > u_t\}$.

   (b) **Sampling**
   - *Reweight*: Set $m_t = \operatorname{card}\{j \leq m \colon Y_{t-1}^j \in \Gamma_t\}$ and $w_{t,0}^j = \frac{1}{m_t} \mathbb{1}_{Y_{t-1}^j \in \Gamma_t}$.
   - *Resample*: Generate a sample $(\widetilde{Y}_t^j)_{1 \leq j \leq m}$ from the distribution $\sum_{j=1}^m w_{t,0}^j \delta_{Y_{t-1}^j}$.
   - *Move*: For each $j \leq m$, draw $Y_t^j \backsim K(\widetilde{Y}_t^j, \cdot)$. (NB: here, f is evaluated.)

   (c) Increment t.
3. **Estimation:** Let m_u be the number of particles such that $f(Y_{T-1}^j) > u$. Set
$$\widehat{\alpha}^{\text{SS}} = \frac{m_u}{m} p_0^{T-1}.$$

Denote by E_n (resp., P_n) the conditional expectation (resp., conditional probability) with respect to $X_1, \xi(X_1), \ldots, X_n, \xi(X_n)$ for any $n \leq N$, and assume, as in section 2, that $\mathsf{P}_{\mathbb{X}}$ has a probability density function $\pi_{\mathbb{X}}$ with respect to the Lebesgue measure. Then, a natural (mean-square optimal) Bayesian estimator of $\alpha = \mathsf{P}_{\mathbb{X}}(\Gamma)$ using n evaluations is the posterior mean

$$\mathsf{E}_n(\alpha) = \mathsf{E}_n\left(\int_{\mathbb{X}} \mathbb{1}_{\xi(x)>u}\, \pi_{\mathbb{X}}(x)\, \mathrm{d}x\right) = \int_{\mathbb{X}} \tilde{g}_{n,u}(x)\, \pi_{\mathbb{X}}(x)\, \mathrm{d}x, \tag{7}$$

where $\tilde{g}_{n,u}(x) \coloneqq \mathsf{E}_n(\mathbb{1}_{\xi(x)>u}) = \mathsf{P}_n(\xi(x) > u)$ is the coverage function of the random set Γ (see, e.g., [16]). Note that since ξ is Gaussian, $\tilde{g}_{n,u}(x)$ can be readily computed for any x using the kriging equations (see, e.g., [6, section 2.4]).

Observe that $\tilde{g}_{n,u} \approx \mathbb{1}_{\Gamma}$ when the available evaluation results are informative enough to classify most input points correctly (with high probability) with respect to u. This suggests that the computation of the right-hand side of (7) should not be carried out using a brute force Monte Carlo approximation, and would benefit from an SMC approach similar to the subset simulation algorithm described in section 2. Moreover, combining an SMC approach with the Bayesian viewpoint is also beneficial for the problem of choosing (sequentially) the sampling points $X_1, \ldots, X_N$. In our work, we focus on a *stepwise uncertainty reduction* (SUR) strategy [6, 52]. Consider the function $L : \widehat{\Gamma} \mapsto \mathsf{P}_{\mathbb{X}}(\Gamma \triangle \widehat{\Gamma})$, which quantifies the loss incurred by choosing an estimator $\widehat{\Gamma}$ instead of the excursion set Γ, where $\triangle$ stands for the symmetric difference operator. Here, at each iteration n, we choose the estimator $\widehat{\Gamma}_{n,u} = \{x \in \mathbb{X} \mid \tilde{g}_{n,u}(x) > 1/2\}$. A SUR strategy, for the loss L and the estimators $\widehat{\Gamma}_{n,u}$,

consists in choosing a point X_{n+1} at step n in such a way as to minimize the expected loss at step $n+1$:

$$X_{n+1} = \operatorname*{argmin}_{x_{n+1}\in\mathbb{X}} J_n\left(x_{n+1}\right), \tag{8}$$

where

$$J_n\left(x_{n+1}\right) \coloneqq \mathsf{E}_n\big(\mathsf{P}_{\mathbb{X}}(\Gamma \triangle \widehat{\Gamma}_{n+1,u}) \;\big|\; X_{n+1} = x_{n+1}\big). \tag{9}$$

For computational purposes, J_n can be rewritten as an integral over $\mathbb{X}$ of the expected probability of misclassification $\tau_{n+1,u}$ (see [6] for more details):

$$J_n(x_{n+1}) = \int_{\mathbb{X}} \mathsf{E}_n\left(\tau_{n+1,u}(x) \;\middle|\; X_{n+1} = x_{n+1}\right) \pi_{\mathbb{X}}(x)\,\mathrm{d}x, \tag{10}$$

where

$$\tau_{n,u}(x) \coloneqq \mathsf{P}_n\Big(x \in \Gamma \triangle \widehat{\Gamma}_{n,u}\Big) = \min\Big(\tilde{g}_{n,u}(x),\, 1-\tilde{g}_{n,u}(x)\Big). \tag{11}$$

For moderately small values of α, it is possible to use a sample from $\mathsf{P}_{\mathbb{X}}$ both for the approximation of the integral in the right-hand side of (10) and for an approximate minimization of J_n (by exhaustive search in the set of sample points). However, this simple Monte Carlo approach would require a very large sample size to be applicable for very small values of α; a subset simulation–like SMC approach will now be proposed as a replacement.

3.2. A sequential Monte Carlo approach. Assume that α is small, and consider a decreasing sequence of subsets $\mathbb{X} = \Gamma_0 \supset \Gamma_1 \supset \cdots \supset \Gamma_T = \Gamma$, where $\Gamma_t = \{x \in \mathbb{X} : f(x) > u_t\}$, as in section 2. For each $t \le T$, denote by $\widehat{\alpha}_t^{\mathrm{B}}$ the Bayesian estimator of $\alpha_t = \mathsf{P}_{\mathbb{X}}(\Gamma_t)$ obtained from n_t observations of ξ at points $X_1, \ldots, X_{n_t}$:

$$\widehat{\alpha}_t^{\mathrm{B}} \coloneqq \mathsf{E}_{n_t}\left(\alpha_t\right) = \int_{\mathbb{X}} g_t \,\mathrm{d}\mathsf{P}_{\mathbb{X}}, \tag{12}$$

where $g_t(x) \coloneqq \tilde{g}_{n_t,u_t}(x) = \mathsf{P}_{n_t}(\xi(x) > u_t)$.

The main idea of our new algorithm is to use an SMC approach to construct a sequence of approximations $\widehat{\widehat{\alpha}}_t^{\mathrm{BSS}}$ of the Bayesian estimators $\widehat{\alpha}_t^{\mathrm{B}}$, $1 \le t \le T$ (as explained earlier, the particles of these SMC approximations will also provide suitable candidate points for the optimization of a sequential design criterion). To this end, consider the sequence of probability density functions q_t defined by

$$q_t(x) \coloneqq \frac{\pi_{\mathbb{X}}(x)\, g_t(x)}{\int \pi_{\mathbb{X}}(y)\, g_t(y)\,\mathrm{d}y} = \frac{1}{\widehat{\alpha}_t^{\mathrm{B}}}\, \pi_{\mathbb{X}}(x)\, g_t(x). \tag{13}$$

We can write a recurrence equation for the sequence of Bayesian estimators $\widehat{\alpha}_t^{\mathrm{B}}$, similar to that used for the probabilities α_t in (2):

$$\widehat{\alpha}_{t+1}^{\mathrm{B}} = \int g_{t+1}(x)\, \pi_{\mathbb{X}}(x)\,\mathrm{d}x = \widehat{\alpha}_t^{\mathrm{B}} \int \frac{g_{t+1}(x)}{g_t(x)}\, q_t(x)\,\mathrm{d}x. \tag{14}$$

This suggests constructing recursively a sequence of estimators $(\hat{\hat{\alpha}}_t^{\mathrm{BSS}})$ using the following relation:

$$\hat{\hat{\alpha}}_{t+1}^{\mathrm{BSS}} = \hat{\hat{\alpha}}_t^{\mathrm{BSS}} \sum_{j=1}^{m} w_t^j \frac{g_{t+1}(Y_t^j)}{g_t(Y_t^j)}, \quad 0 \leq t < T, \tag{15}$$

where $(w_t^j, Y_t^j)_{1 \leq j \leq m}$ is a weighted sample of size m targeting q_t (as in section 2.2) and $\hat{\hat{\alpha}}_0^{\mathrm{BSS}} = 1$. The final estimator can be written as

$$\hat{\hat{\alpha}}_T^{\mathrm{BSS}} = \prod_{t=0}^{T-1} \frac{\hat{\hat{\alpha}}_{t+1}^{\mathrm{BSS}}}{\hat{\hat{\alpha}}_t^{\mathrm{BSS}}} = \prod_{t=0}^{T-1} \sum_{j=1}^{m} w_t^j \frac{g_{t+1}(Y_t^j)}{g_t(Y_t^j)}. \tag{16}$$

Remark 5. The connection between the proposed algorithm and the original subset simulation algorithm is clear from the similarity between the recurrence relations (2) and (14), and from the use of SMC simulation in both algorithms to construct recursively a product-type estimator of the probability of failure (see also [21, section 3.2.1], where this type of estimator is mentioned in a very general SMC framework).

Our choice for the sequence of densities $q_1, \ldots, q_T$ also relates to the original subset simulation algorithm. Indeed, note that $q_t(x) \propto \mathsf{E}_{n_t}(\mathbb{1}_{\xi > u_t}\, \pi_{\mathbb{X}})$, and recall from (3) that $q_t \propto \mathbb{1}_{\xi > u_t}\, \pi_{\mathbb{X}}$ is the target distribution used in the subset simulation algorithm at stage t. This choice of instrumental density is also used in [27, 28] in the context of a two-stage adaptive importance sampling algorithm. This is, indeed, a quite natural choice, since $\tilde{q}_t \propto \mathbb{1}_{\xi > u_t}\, \pi_{\mathbb{X}}$ is the optimal instrumental density for the estimation of α_t by importance sampling (see, e.g., [49, Theorem 3.12]).

3.3. The Bayesian subset simulation (BSS) algorithm. The algorithm consists of a sequence of stages (or iterations). For the sake of clarity, assume first that the sequence of thresholds (u_t) is given. Then, each stage $t \in \mathbb{N}$ of the algorithm is associated to a threshold u_t and the corresponding excursion set $\Gamma_t = \{f > u_t\}$.

The initialization stage ($t = 0$) starts with the construction of a *space filling* set of points $\{X_1, \ldots, X_{n_0}\}$ in $\mathbb{X}$[4] and an initial Monte Carlo sample $\mathbb{Y}_0 = \{Y_0^1, \ldots, Y_0^m\}$, consisting of a set of independent random variables drawn from the density $q_0 = \pi_{\mathbb{X}}$.

After initialization, each subsequent stage $t \geq 1$ of BSS involves two phases: an *estimation phase*, where the estimation of Γ_t is carried out, and a *sampling phase*, where a sample $\mathbb{Y}_t$ targeting the density q_t associated to u_t is produced from the previous sample $\mathbb{Y}_{t-1}$ using the reweight/resample/move SMC scheme described in section 2.3.

In greater detail, the estimation phase consists in making $N_t \geq 0$ new evaluations of f to refine the estimation of Γ_t. The number of evaluations is meant to be much smaller than the size m of the Monte Carlo sample—which would be the number of evaluations in the classical subset simulation algorithm. The total number of evaluations at the end of the

[4]See section 4.2.1 for more information on the specific technique used in this paper. Note that it is, of course, possible, albeit not required, to use the BSS algorithm to first perform a change of variables in order to work, e.g., in the standard Gaussian space. Whether this will improve the performance of the BSS algorithm is very difficult to say, in general, and will depend on the example at hand.

estimation phase at stage t is denoted by $n_t = n_{t-1} + N_t$. The total number of evaluations used by BSS is thus $n_T = n_0 + \sum_{t=1}^{T} N_t$. New evaluation points $X_{n_{t-1}+1}, X_{n_{t-1}+2}, \ldots, X_{n_t}$ are determined using a SUR sampling strategy[5] targeting the threshold u_t, as in section 3.1 (see supplementary material SM1 of M107827_01.pdf [local/web 348KB] for details about the numerical procedure).

In practice, the sequence of thresholds is not fixed beforehand, and adaptive techniques are used to choose the thresholds (see section 3.4) and the number of points per stage (see section 3.5).

The BSS algorithm is presented in pseudocode form in Algorithm 2.

Remark 6. Algorithms involving Gaussian process–based adaptive sampling and subset simulation have been proposed by Dubourg and coauthors [26, 29] and by Huang, Chen, and Zhu [37]. Dubourg's work addresses a different problem (namely, reliability-based design optimization). The paper by Huang, Chen, and Zhu, published very recently, addresses the estimation of small probabilities of failure. We emphasize that, unlike BSS, none of these algorithms involves a direct interaction between the selection of evaluation points (adaptive sampling) and subset simulation—which is simply applied, in its original form, to the posterior mean of the Gaussian process (also known as kriging predictor).

3.4. Adaptive choice of the thresholds u_t. As discussed in section 2.4, it can be proved that for an idealized version of the subset simulation algorithm with fixed thresholds $u_0 < u_1 < \cdots < u_T = u$, it is optimal to choose the thresholds to make all conditional probabilities $\mathsf{P}_{\mathbb{X}}(\Gamma_{t+1}|\Gamma_t)$ equal. This leads to the idea of choosing the thresholds adaptively in such a way that in the product estimate

$$\widehat{\alpha}_T^{\mathrm{SS}} = \prod_{t=1}^{T} \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}_{\Gamma_t}(Y_{t-1}^i),$$

each term but the last is equal to some prescribed constant p_0. In other words, u_t is chosen as the $(1-p_0)$-quantile of $\mathbb{Y}_{t-1}$. This idea was first suggested in [2, section 5.2] on the heuristic ground that the algorithm should perform well when the conditional probabilities are neither too small (otherwise they are hard to estimate) nor too large (otherwise a large number of stages is required).

Consider now an idealized BSS algorithm, where (a) the initial design of the experiment is independent of $\mathbb{Y}_0$; (b) the SUR criterion is computed exactly, or using a discretization scheme that does not use the $\mathbb{Y}_t$'s; (c) the minimization of the SUR criterion is carried out independently of the $\mathbb{Y}_t$'s; and (d) the particles Y_t^j are i.i.d. according to q_t. Assumptions (a)–(c) ensure that the sequence of densities $(q_t)_{1 \le t \le T}$ is deterministic, given ξ. Then (see Appendix A),

$$\operatorname{var}\left(\frac{\widehat{\widehat{\alpha}}_T^{\mathrm{BSS}}}{\widehat{\alpha}_T^{\mathrm{B}}} \,\middle|\, \xi \right) = \frac{1}{m} \sum_{t=1}^{T} \kappa_t + O\left(\frac{1}{m^2}\right), \tag{17}$$

[5]Other sampling strategies (also known as "sequential design" or "active learning" methods) could be used as well. See [6] for a review and comparison of sampling criteria.

Algorithm 2

Bayesian subset simulation algorithm.

1. **Initialization** (stage 0)
   (a) Evaluate f on a set of points $\{X_1, \ldots, X_{n_0}\}$, called the *initial design* (see section 4.2.1 for details).
   (b) Generate an i.i.d. sample $\mathbb{Y}_t = \{Y_0^1, \ldots, Y_0^m\}$ from $\mathsf{P}_{\mathbb{X}}$.
   (c) Choose a prior P_0 (see sections 3.1 and 4.2.1 for details).
   (d) Set $u_0 = -\infty$, $g_0 = \tilde{g}_{0,-\infty} = \mathbb{1}_{\mathbb{X}}$, $n = n_0$, and $t = 1$.
2. **Repeat** (stage t)
   (a) **Estimation**
   - Set $k = 0$ and repeat
     - Select a threshold $\tilde{u}_{t,k}$ by solving (20) for u_t (with $n_t = n$).
     - Stop if the condition (21) is met, with $n_t = n$ and $u_t = \tilde{u}_{t,k}$.
     - Select X_{n+1} using the SUR strategy (8)–(11) with respect to $\tilde{u}_{t,k}$.
     - Evaluate f at X_{n+1}. Increment n and k.
   - Set $N_t = k$, $n_t = n$, $u_t = \tilde{u}_{t,k}$ and $g_t = \tilde{g}_{n_t,u_t} = \mathsf{P}_{n_t}\big(\xi(\cdot) > u_t\big)$.

   (b) **Sampling**
   - *Reweight*: Calculate weights $w_{t,0}^j \propto g_t(Y_{t-1}^j)/g_{t-1}(Y_{t-1}^j)$, $1 \le j \le m$.
   - *Resample*: Generate a sample $(\widetilde{Y}_t^j)_{1\le j\le m}$ from the distribution $\sum_{j=1}^m w_{t,0}^j \delta_{Y_{t-1}^j}$.
   - *Move*: For each $j \le m$, draw $Y_t^j \rightsquigarrow K(\widetilde{Y}_t^j, \cdot)$.

   (c) Increment t.
3. **Estimation:** The final probability of failure is estimated by
$$\widehat{\widehat{\alpha}}_T^{\mathrm{BSS}} = \prod_{t=0}^{T-1} \left(\frac{1}{m} \sum_{j=1}^{m} \frac{g_{t+1}(Y_t^j)}{g_t(Y_t^j)} \right).$$

where

$$\kappa_t := \frac{\int_{\mathbb{X}} g_t^2/g_{t-1}\, \pi_{\mathbb{X}}}{\left(\widehat{\alpha}_t^{\mathrm{B}}\right)^2 / \widehat{\alpha}_{t-1}^{\mathrm{B}}} - 1. \tag{18}$$

Minimizing the leading term $\frac{1}{m}\sum_{t=1}^T \kappa_t$ in (17) by an appropriate choice of thresholds is not as straightforward as in the case of the subset simulation algorithm. Assuming that $g_{t-1} \approx 1$ wherever g_t is not negligible (which is a reasonable assumption, since $g_t(x) = \mathsf{P}_{n_t}(\xi(x) > u_t)$ and $u_t > u_{t-1}$), we get

$$\int_{\mathbb{X}} g_t^2/g_{t-1}\, \pi_{\mathbb{X}} \approx \int_{\mathbb{X}} g_t^2\, \pi_{\mathbb{X}} \le \int_{\mathbb{X}} g_t\, \pi_{\mathbb{X}} = \widehat{\alpha}_t^{\mathrm{B}},$$

and therefore the variance (17) is approximately upper-bounded by

$$\frac{1}{m} \sum_{t=1}^{T} \frac{1 - \widehat{p}_t^{\mathrm{B}}}{\widehat{p}_t^{\mathrm{B}}}, \tag{19}$$

where $\widehat{p}_t^{\mathrm{B}} \coloneqq \widehat{\alpha}_t^{\mathrm{B}}/\widehat{\alpha}_{t-1}^{\mathrm{B}}$. Minimizing the approximate upper bound (19) under the constraint

$$\prod_{t=1}^{T} \widehat{p}_t^{\mathrm{B}} = \widehat{\alpha}_T^{\mathrm{B}}$$

leads to choosing the thresholds u_t in such a way that $\widehat{p}_t^{\mathrm{B}}$ is the same for all stages t, that is, $\widehat{p}_t^{\mathrm{B}} = (\widehat{\alpha}_T^{\mathrm{B}})^{1/T}$. As a consequence, we propose to choose the thresholds adaptively using the condition that at each stage (but the last), the natural estimator $\widehat{\widehat{\alpha}}_t^{\mathrm{BSS}}/\widehat{\widehat{\alpha}}_{t-1}^{\mathrm{BSS}}$ of $\widehat{p}_t^{\mathrm{B}}$ is equal to some prescribed probability p_0. Owing to (15), this amounts to choosing u_t in such a way that

$$\frac{1}{m}\sum_{i=1}^{m} \frac{g_t(Y_{t-1}^i)}{g_{t-1}(Y_{t-1}^i)} = p_0 \tag{20}$$

should be satisfied.

Equation (20) is easy to solve, since the left-hand side is a strictly decreasing and continuous function of u_t (to be precise, continuity holds under the assumption that the posterior variance of ξ does not vanish on one of the particles). In practice, we solve (20) each time a new evaluation is made, which yields a sequence of intermediate thresholds (denoted by $\tilde{u}_{t,0}$, $\tilde{u}_{t,1}, \ldots$ in Algorithm 2) at each stage $t \geq 1$. The actual value of u_t at stage t is only known after the last evaluation of stage t.

Remark 7. Alternatively, the ESS could be used to select the thresholds, as proposed in [22]. This idea will not be pursued in this paper. The threshold selected by the ESS-based approach will be close to the threshold selected by (20) when all of the ratios $g_t(Y_{t-1}^i)/g_{t-1}(Y_{t-1}^i)$, or most of them, are either close to zero or close to one.

3.5. Adaptive choice of the number N_t of evaluation at each stage. In this section, we propose a technique to choose adaptively the number N_t of evaluations of f that must be done at each stage of the algorithm.

Assume that $t \geq 1$ is the current stage number; at the beginning of the stage, n_{t-1} evaluations are available from previous stages. After several additional evaluations, the number of available observations of f is $n \geq n_{t-1}$. We propose to stop adding new evaluations when the expected error of estimation of the set Γ_t, measured by $\mathsf{E}_n(\mathsf{P}_{\mathbb{X}}(\Gamma_t \mathbin{\triangle} \widehat{\Gamma}_{n,u_t}))$, becomes smaller than some prescribed fraction η_t of its expected volume $\mathsf{E}_n(\mathsf{P}_{\mathbb{X}}(\Gamma_t))$ under $\mathsf{P}_{\mathbb{X}}$. Writing these two quantities as

$$\begin{aligned}
\mathsf{E}_n\left(\mathsf{P}_{\mathbb{X}}\left(\Gamma_t\right)\right) &= \int_{\mathbb{X}} \tilde{g}_{n,u_t}(x)\, \pi_{\mathbb{X}}(x)\, \mathrm{d}x = \widehat{\alpha}_{t-1}^{\mathrm{B}} \int_{\mathbb{X}} \frac{\tilde{g}_{n,u_t}(x)}{g_{t-1}(x)} q_{t-1}(x)\, \mathrm{d}x, \\
\mathsf{E}_n\left(\mathsf{P}_{\mathbb{X}}\left(\Gamma \mathbin{\triangle} \widehat{\Gamma}_{n,u_t}\right)\right) &= \int_{\mathbb{X}} \tau_{n,u_t}(x)\, \pi_{\mathbb{X}}(x)\, \mathrm{d}x = \widehat{\alpha}_{t-1}^{\mathrm{B}} \int_{\mathbb{X}} \frac{\tau_{n,u_t}(x)}{g_{t-1}(x)} q_{t-1}(x)\, \mathrm{d}x,
\end{aligned}$$

where $\tilde{g}_{n,u_t}$ and τ_{n,u_t} have been defined in section 3.1, and estimating the integrals on the right-hand side using the SMC sample $\mathbb{Y}_{t-1}$, we end up with the stopping condition

$$\frac{1}{m}\sum_{i=1}^{m} \frac{\tau_{n,u_t}(Y_{t-1}^i)}{g_{t-1}(Y_{t-1}^i)} \leq \eta_t \cdot \frac{1}{m}\sum_{i=1}^{m} \frac{\tilde{g}_{n,u_t}(Y_{t-1}^i)}{g_{t-1}(Y_{t-1}^i)},$$

Table 1
Summary of test cases.

Example	Name	d	α_{ref}
1	four-branch series system	2	$5.596 \cdot 10^{-9}$
2	deviation of a cantilever beam	2	$3.937 \cdot 10^{-6}$
3	response of a nonlinear oscillator	6	$1.514 \cdot 10^{-8}$

which, if u_t is readjusted after each evaluation using (20), boils down to

$$\sum_{i=1}^{m} \frac{\tau_{n,u_t}(Y_{t-1}^i)}{g_{t-1}(Y_{t-1}^i)} \leq \eta_t m p_0. \tag{21}$$

Remark 8. In the case where several evaluations of the function can be carried out in parallel, it is possible to select evaluation points in batches during the sequential design phase of the algorithm. A batch-sequential version of the SUR strategy (8)–(11) has been proposed in [15].

Remark 9. The stopping criterion (21) is slightly different from the one proposed earlier in [41]: $\sum_{i=1}^{m} \tau_{n,u_t}(Y_{t-1}^i) \leq \eta' m$. If we set $\eta' = \eta_t p_0$ and assume (quite reasonably) that $g_{t-1}(Y_{t-1}^i) \approx 1$ for the particles where $\tau_{n,u_t}(Y_{t-1}^i)$ is not negligible, then it becomes clear that the two criteria are essentially equivalent. As a consequence, the left-hand side of (21) can also be interpreted, approximately, as the expected number of misclassified particles (where the expectation is taken with respect to ξ, conditionally on the particles).

4. Numerical experiments.

In this section, we illustrate the proposed algorithm on three classical examples from the structural reliability literature and compare our results with those from the classical subset simulation algorithm and the ^{2}SMART algorithm [10, 20]. These examples are not actually expensive to evaluate, which makes it possible to analyze the performance of the algorithms through extensive Monte Carlo simulations, but the results are nonetheless relevant to the case of expensive-to-evaluate simulators since performance is measured in terms of the number of function evaluations (see section 4.3.2 for a discussion).

The computer programs used to conduct these numerical experiments are freely available from https://sourceforge.net/p/kriging/contrib-bss under the Lesser General Public License [33]. They are written in the Matlab/Octave language and use the STK toolbox [7] for Gaussian process modeling. For convenience, a software package containing both the code for the BSS algorithm itself and the STK toolbox is provided as supplementary material file M107827_02.zip [local/web 767KB].

4.1. Test cases.

For each of the following test cases, the reference value for the probability α has been obtained from 100 independent runs of the subset simulation algorithm with sample size $m = 10^7$ (see Table 1).

Example 1 (four-branch series system). Our first example is a variation on a classical structural reliability test case (see, e.g., [30, Example 1], with $k = 6$), where the threshold u is modified to make α smaller. The objective is to estimate the probability $\alpha = \mathsf{P}_{\mathbb{X}}(f(X) < u)$,

where

$$f(x_1, x_2) = \min \left\{ \begin{array}{l} 3 + 0.1(x_1 - x_2)^2 - (x_1 + x_2)/\sqrt{2}, \\ 3 + 0.1(x_1 - x_2)^2 + (x_1 + x_2)/\sqrt{2}, \\ (x_1 - x_2) + 6/\sqrt{2}, \\ (x_2 - x_1) + 6/\sqrt{2} \end{array} \right\} \tag{22}$$

and $X_1, X_2 \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,1)$. Taking $u = -4$, the probability of failure is approximately $5.596{\cdot}10^{-9}$, with a coefficient of variation of about 0.04%. Figure 1 (left panel) shows the failure domain and a sample from the input distribution $\mathsf{P}_{\mathbb{X}}$.

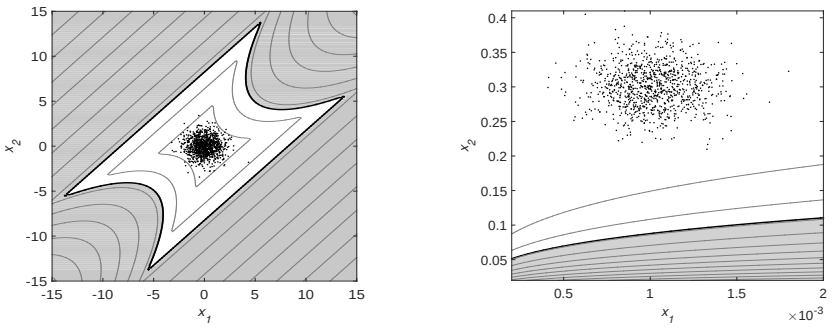


Figure 1. *Contour plots of f in Example 1 (left) and Example 2 (right), along with a sample of size $m = 10^3$ from $\mathsf{P}_{\mathbb{X}}$ (dots). A failure happens when x is in the gray area.*

Example 2 (deviation of a cantilever beam). Consider a cantilever beam, with a rectangular cross-section, subjected to a uniform load. The deflection of the tip of the beam can be written as

$$f(x_1, x_2) = \frac{3L^4}{2E} \frac{x_1}{x_2^3}, \tag{23}$$

where x_1 is the load per unit area, x_2 is the thickness of the beam, $L = 6\,\text{m}$, and $E = 2.6{\cdot}10^4\,\text{MPa}$. The input variables X_1 and X_2 are assumed independent, with $X_1 \sim \mathcal{N}(\mu_1, \sigma_1^2)$, $\mu_1 = 10^{-3}\,\text{MPa}$, $\sigma_1 = 0.2\mu_1$, and $X_2 \sim \mathcal{N}(\mu_2, \sigma_2^2)$, $\mu_2 = 0.3\,\text{m}$, $\sigma_2 = 0.1\mu_2$. A failure occurs when f is larger than $u = L/325$. The probability of failure is approximately $3.937{\cdot}10^{-6}$, with a coefficient of variation of about 0.03%. Note that the distribution of X_2 has been modified, with respect to the usual formulation (see, e.g., [35]), to make α smaller. Figure 1 (right panel) shows a contour plot of f, along with a sample of the input distribution.

Example 3 (response of a nonlinear oscillator). In this example (see, e.g., [31]), the input variable is six-dimensional, and the cost function is

$$f\,(x_1, x_2, x_3, x_4, x_5, x_6) = 3x_4 - \left| \frac{2x_5}{x_1 {w_0}^2} \sin\left(\frac{w_0 x_6}{2}\right) \right|, \tag{24}$$

where $w_0 = \sqrt{\frac{x_2 + x_3}{x_1}}$. The input variables are assumed independent and normal, with mean and variance parameters given in Table 2. A failure happens when the cost function is lower

Table 2
Example 3: Means and standard deviations of the input variables.

Variable	x_1	x_2	x_3	x_4	x_5	x_6
μ_i	1	1	0.1	0.5	0.45	1
σ_i	0.05	0.1	0.01	0.05	0.075	0.2

than the threshold $u = 0$. The probability of failure is approximately $1.514 \cdot 10^{-8}$, with a coefficient of variation of about 0.04%. This variant of the problem corresponds exactly to the harder case in [31].

4.2. Experimental settings.

4.2.1. BSS algorithm.

Initial design of experiments. We start with an initial design of size $n_0 = 5d$ (see [43] for a discussion on the size of the initial design in computer experiments), generated as follows. First, a compact subset $\mathbb{X}_0 \subset \mathbb{X}$ is constructed:[6]

$$\mathbb{X}_0 = \prod_{j=1}^{d} \left[q_\varepsilon^j ; q_{1-\varepsilon}^j \right],$$

where q_ε^j and $q_{1-\varepsilon}^j$ are the quantiles of order ε and $1-\varepsilon$ of the jth input variable. Then, a "good" Latin hypercube sampling (LHS) design on $[0;1]^d$ is obtained as the best design according to the maximin criterion [38, 45] in a set of Q random LHS designs, and then scaled to $\mathbb{X}_0$ using an affine mapping. The values $\varepsilon = 10^{-5}$ and $Q = 10^4$ have been used in all of our experiments.

Stochastic process prior. A Gaussian process prior with an unknown constant mean and a stationary anisotropic Matérn covariance function with regularity 5/2 is used as our prior information about f (see supplementary material SM2 of M107827_01.pdf [local/web 348KB] for more details). The unknown mean is integrated out as usual, using an improper uniform prior on $\mathbb{R}$; as a consequence, the posterior mean coincides with the so-called ordinary kriging predictor. The remaining hyperparameters (variance and range parameters of the covariance function) are estimated, following the empirical Bayes philosophy, by maximization of the marginal likelihood.[7] The hyperparameters are estimated first on the data from the initial design and then re-estimated after each new evaluation. In practice, we recommend checking the estimated parameters once in a while using, e.g., leave-one-out cross-validation.

SMC parameters. Several values of the sample size m will be tested to study the relation between the variance of the estimator and the number of evaluations: $m \in \{500, 1000, 2000, \ldots\}$. Several iterations of an adaptive Gaussian random walk Metropolis–Hastings algorithm, fully described in supplementary material SM3 of M107827_01.pdf [local/web 348KB], are used for the move step of the algorithm.

[6] A similar technique is used by Dubourg and coauthors in a context of reliability-based design optimization [26, 29].

[7] Used in combination with a uniform prior for the mean, for this specific model, the MML method is equivalent to the restricted maximum likelihood (ReML) method advocated in [51, section 6.4].

Stopping criterion for the SUR strategy. The number of evaluations selected using the SUR strategy is determined adaptively, using the stopping criterion (21) from section 3.5, with $\eta_t = 0.5$ for all $t < T$ (i.e., for all intermediate stages) and $\eta_T = 0.1\,\widehat{\delta}_{m,T}$, where $\widehat{\delta}_{m,T}$ is the estimated coefficient of variation for the SMC estimator $\widehat{\widehat{\alpha}}_T^{\mathrm{BSS}}$ of $\widehat{\alpha}_T^{\mathrm{B}}$ (see Appendix A). Furthermore, we require for robustness a minimal number $N_{\min}$ of evaluations at each stage, with $N_{\min} = 2$ in all our simulations.

Adaptive choice of the thresholds. The successive thresholds u_t are chosen using the adaptive rule proposed in section 3.4, by solving (20) with $p_0 = 0.1$. This value has been found experimentally to be neither too large (to avoid having a large number of stages) nor too small (to avoid losing too many particles during the resampling step).[8]

4.2.2. Subset simulation algorithm. The parameters used for the subset simulation algorithm are exactly the same, in all of our simulations, as those used in the "SMC part" of the BSS algorithm (see section 4.2.1). In particular, the number m_0 of surviving particles at each stage is determined according to the rule $p_0 = \frac{m_0}{m} = 0.1$ (see Table 1), and the adaptive MCMC algorithm described in supplementary material SM3 of M107827_01.pdf [local/web 348KB] is used to move the particles. The number of evaluations made by the subset simulation algorithm is considered to be $m + (T-1)(1-p_0)\,m$, as explained in section 2.4—in other words, in order to make the comparison as fair as possible, the additional evaluations required by the adaptive MCMC procedure are not taken into account.

4.2.3. ^{2}SMART algorithm. ^{2}SMART [10, 20] is another algorithm for the estimation of small probabilities, which is based on the combination of subset simulation with support vector machines. We will present results obtained using the implementation of ^{2}SMART proposed in the software package FERUM 4.1 [8], with all parameters set to their default values (which are equal to the values given in [10]).

Remark 10. Several other methods addressing the estimation of small probabilities of failure for expensive-to-evaluate functions have appeared recently in the structural reliability literature [4, 9, 12, 31, 37]. ^{2}SMART was selected as a reference method due to the availability of a free software implementation in FERUM. A more comprehensive benchmark is left for future work.

4.3. Results.

4.3.1. Illustration. We first illustrate how BSS works using one run of the algorithm on Example 1 with sample size $m = 1000$. Snapshots of the algorithm at stages $t = 1$, $t = 5$, and $t = T = 9$ are presented in Figure 2. Observe that the additional evaluation points selected at each stage using the SUR criterion (represented by black triangles) are located in the vicinity of the current level set. The actual number of points selected at each stage, determined by the adaptive stopping rule, is reported in Table 3. Observe also that the set of particles (black dots in the right column) is able to effectively capture the bimodal target

[8] Note that the considerations of Remark 4 on the choice of p_0 are not relevant here, since the computational cost of our method is mainly determined by the number of function evaluations, which is not directly related to the number of particles to be simulated. See also supplementary material SM4 of M107827_01.pdf [local/web 348KB].

Table 3

Number of evaluations per stage in Example 1 (four-branch series system). For the BSS algorithm, recall that the number of evaluations at each stage is chosen adaptively (see section 3.5) and is therefore random: the numbers shown here correspond to the run with $m = 1000$ that is shown in Figure 2. For the subset simulation algorithm, the number of evaluations is directly related to the m, $q_0 = 1 - p_0$, and T (see section 2.4).

Stage number t	0	1	2	3	4	5
BSS ($m = 1000$)	$2d = 10$	2	6	3	2	3
Subset simulation	m	$q_0 m$	$q_0 m$	$q_0 m$	$q_0 m$	$q_0 m$

Stage number t	6	7	8	9	Total
BSS ($m = 1000$)	2	3	2	28	61
Subset simulation	$q_0 m$	$q_0 m$	$q_0 m$	0	$m + (T-1)q_0 m$

distribution. Finally, observe that a significant portion of the evaluation budget is spent on the final stage—this is again a consequence of our adaptive stopping rule, which refines the estimation of the final level set until the bias of the estimate is (on average under the posterior distribution) small compared to its standard deviation.

4.3.2. Average results. This section presents average results over 100 independent runs for subset simulation, BSS, and ^{2}SMART.

Figure 3 represents the average number of evaluations used by the BSS algorithm as a function of the sample size m. The number of evaluations spent on the initial design is constant, since it depends only on the dimension d of the input space. The average number of evaluations spent on the intermediate stages ($t < T$) is also very stable[9] and independent of the sample size m. Only the average number of evaluations spent on the final stage—i.e., to learn the level set of interest—is growing with m. This growth is necessary if one wants the estimation error to decrease when m increases: indeed, the variance of $\widehat{\widehat{\alpha}}_T^{\mathrm{BSS}}$ automatically goes to zero at the rate $\frac{1}{m}$, but the bias $\widehat{\alpha}_T^{\mathrm{B}} - \alpha$ does not unless additional evaluations are added at the final level to refine the estimation of Γ.

Figure 4 represents the relative root-mean-square error (RMSE) of all three algorithms as a function of the average number of evaluations. For the subset simulation algorithm, the number of evaluations is directly proportional to m, and the RMSE decreases as expected like $\frac{1}{m}$ (with a constant much smaller than that of plain Monte Carlo simulation). ^{2}SMART clearly outperforms subset simulation but offers no simple way of tuning the accuracy of the final estimate (which is why only one result is presented, using the default settings of the algorithm). Finally, BSS clearly and consistently outperforms both ^{2}SMART and subset simulation in these three examples: the relative RMSE goes to zero at a rate much faster than for subset simulation (a feature that is made possible by the smoothness of the limit-state function, which is leveraged by the Gaussian process model), and the sample size m is the only tuning parameter that needs to be acted upon in order change the accuracy of the final estimate. Figure 5 provides more insight into the error of the BSS estimate by confirming that, as intended by design of the adaptive stopping rule, variance is the main component of the RMSE (in other words, the bias is negligible in these examples).

[9] Actually, for Examples 2 and 3, it is equal to $T\,N_{\min}$ for all runs; in other words, the adaptive stopping rule only came into play at intermediate stages for Example 1.

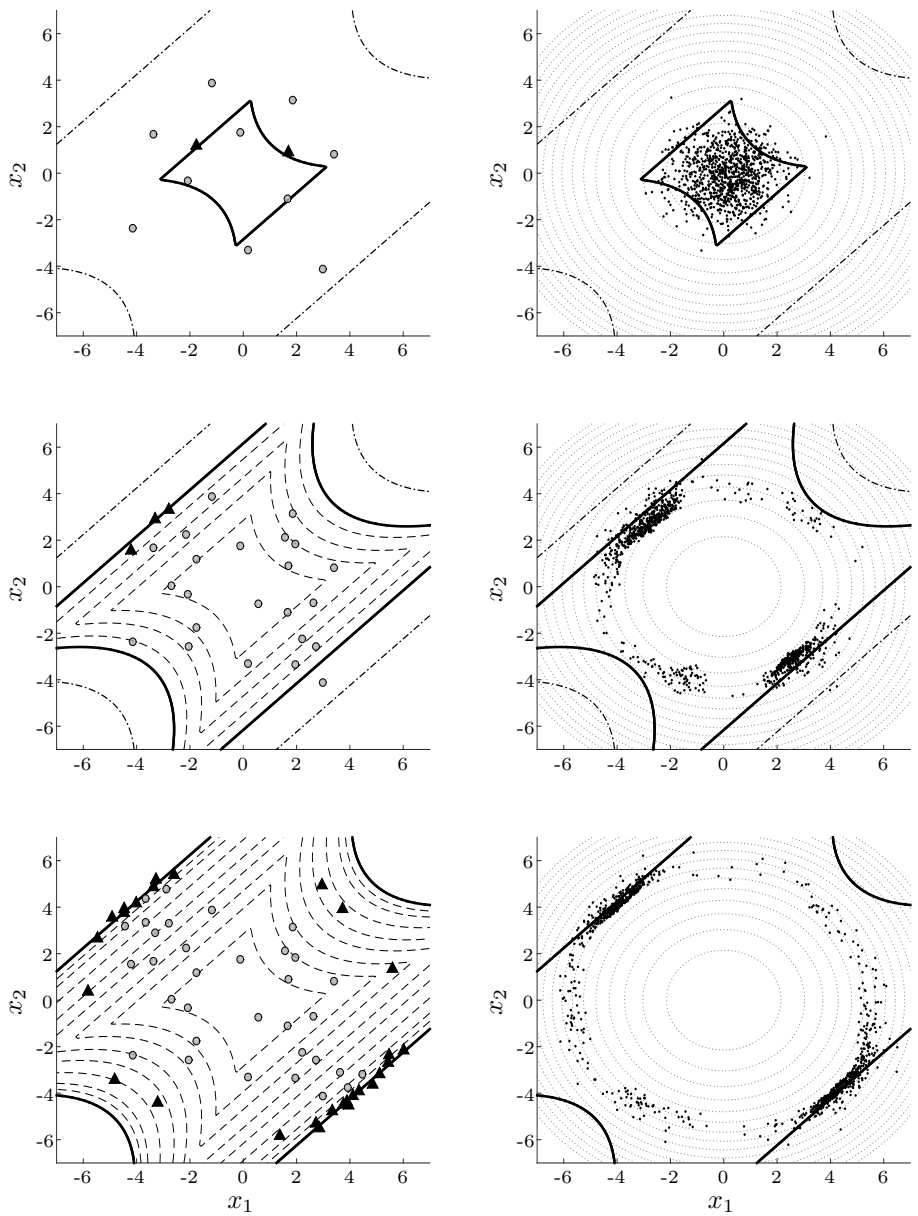
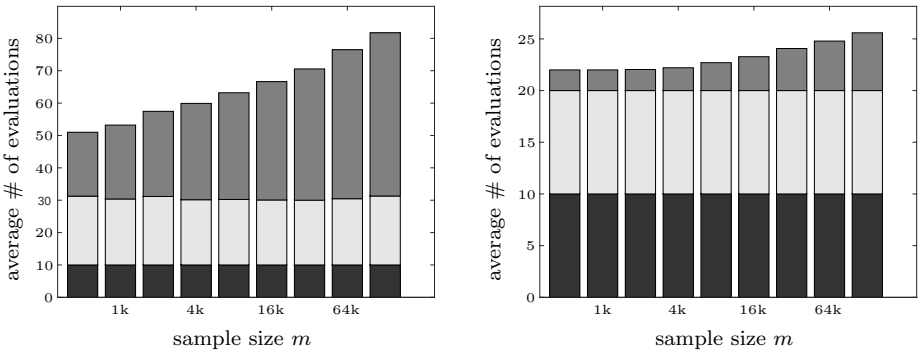
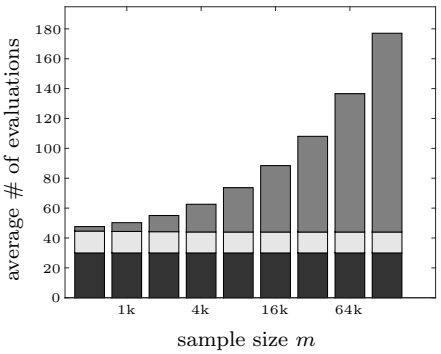


Figure 2. *Snapshots of the BSS algorithm running in Example 1 (four-branch series system) with sample size $m = 1000$. The first, second, and third rows correspond, respectively, to the end of the first stage ($t = 1$), the fifth stage ($t = 5$), and the last stage ($t = T = 9$). The true level set corresponding to current target level u_t is represented by a thick line, and in the left column, true level sets corresponding to previous levels (u_s, $s < t$) are recalled using dashed contours. Evaluation points from previous stages are represented by gray disks (in particular, the initial design of experiment of size $n_0 = 10$ is visible on the top-left panel), and new evaluations performed at the current levels are marked by black triangles. In the right column, the sample points Y_{t-1}^j, $1 \leq j \leq m$, and the level sets of the input density $\pi_{\mathbb{X}}$ (corresponding to probabilities $1 - 10^{-k}$, $k = 1, 2, \ldots$) are represented, respectively, by black dots and dotted lines.*



(a) Example 1: Four-branch.

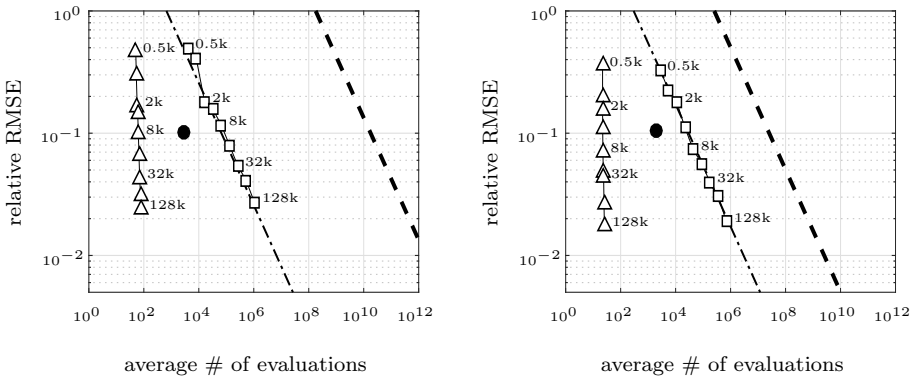
(b) Example 2: Cantilever beam.



(c) Example 3: Nonlinear oscillator.

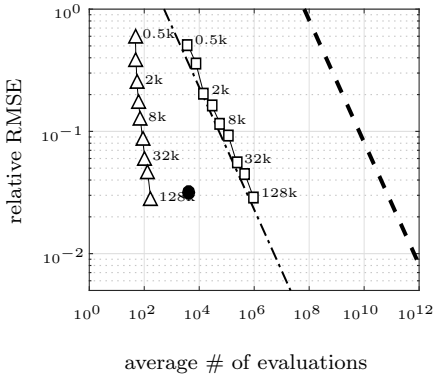
Figure 3. *Average number of evaluations used by the BSS algorithm, over* 100 *independent runs, as a function of the sample size m in Examples 1–3. The total number of evaluations is split into three parts: the size n_0 of the initial design (dark gray), the number $\sum_{t=1}^{T-1} N_t$ of evaluations in intermediate stages (light gray), and the number of evaluations N_T in the final stage (medium gray).*

Finally, note that the BSS estimation involves a computational overhead with respect to subset simulation. A careful analysis of the run times of BSS in the three examples (provided as supplementary material SM4.1 of M107827_01.pdf [local/web 348KB]) reveals that the computational overhead of BSS is approximately equal to $C_0 + C_1\, m\, N_{\mathrm{SUR}}$, where N_{SUR} is the total number of evaluations selected using the SUR criterion (i.e., all evaluations except for the initial design of experiments). This shows that in our implementation, the most time-consuming part of the algorithm is the selection of additional evaluation points using the SUR criterion. However, in spite of its computational overhead, BSS is preferable to the subset simulation algorithm in terms of computation time in the three test cases, as soon as the evaluation time τ_{sim} of f is large enough—larger than, say, 10 ms for the considered range of relative RMSE (see supplementary material SM4.2 of M107827_01.pdf [local/web 348KB] for details). Consider, for instance, Example 1 with $\tau_{\mathrm{sim}} = 1\,s$: BSS with sample size $m = 8000$



(a) Example 1: Four-branch.

(b) Example 2: Cantilever beam.



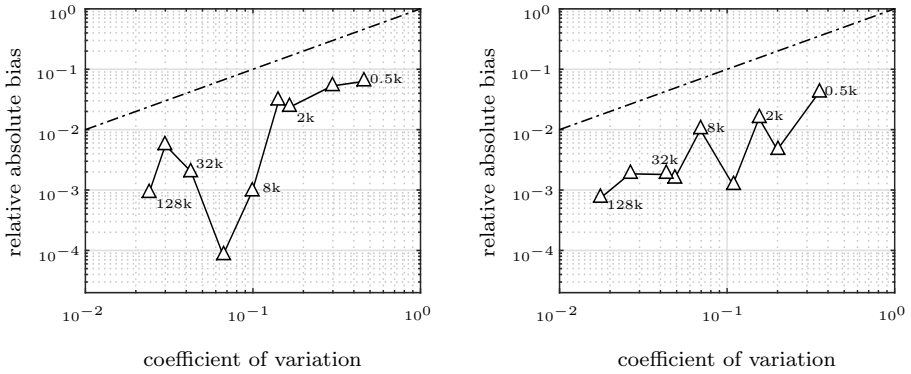
(c) Example 3: Nonlinear oscillator.

Figure 4. *Relative RMSE as a function of the average number of evaluations, over* 100 *independent runs, in Examples* 1–3. *For the subset simulation algorithm (squares) and for the BSS algorithm (triangles), the results are provided for several values of the sample size* ($m \in \{500, 1000, 2000, \ldots\}$). *For the* ^{2}*SMART algorithm (filled circles), only one result is presented, corresponding to the default settings of the algorithm. The expected performance of plain Monte Carlo sampling is represented by a dashed line. The mixed line represents a simple approximation of the relative RMSE for the subset simulation algorithm:* $\frac{T_\alpha}{m}\frac{1-p_0}{p_0}$, *where* $T_\alpha = \lceil \frac{\log \alpha}{\log p_0} \rceil$.

achieves a relative RMSE of approximately 10% in about 3 minutes,[10] while subset simulation requires about 19 hours to achieve a comparable accuracy.

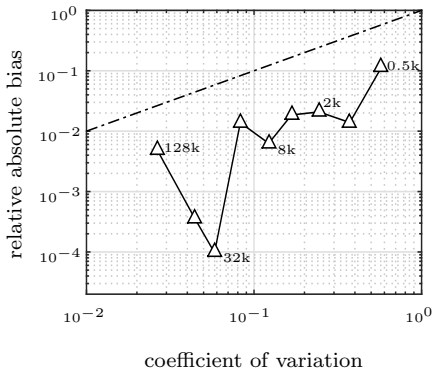
5. Discussion. We propose an algorithm called Bayesian subset simulation (BSS) for the estimation of small probabilities of failure—or, more generally, the estimation of the volume of excursion of a function above a threshold—when the limit-state function is expensive to evaluate. This new algorithm is built upon two key techniques: the SMC method known as subset simulation or adaptive multilevel splitting on the one hand, and the Bayesian (Gaussian process–based) SUR sampling strategy on the other hand. SMC simulation provides the means to evaluate the Bayesian estimate of the probability of failure, and to evaluate and optimize

[10]This computation time can be further decomposed as follows: 1 minute of evaluation time, corresponding to $\bar{N} = 63.2$ evaluations on average (see Figure 3), and 2 minutes of algorithm overhead.



(a) Example 1: Four-branch.

(b) Example 2: Cantilever beam.



(c) Example 3: Nonlinear oscillator.

Figure 5. *Relative absolute bias of the BSS estimator as a function of its coefficient of variation, estimated using* 100 *independent runs. The relative absolute bias is estimated using, for each test case, the reference value* α_{ref} *provided in Table* 1.

the SUR sampling criterion. In turn, the SUR sampling strategy makes it possible to estimate the level sets of the (smooth) limit-state function using a restricted number of evaluations, and thus to build a good sequence of target density for SMC simulation. Our numerical experiments show that this combination achieves significant savings in evaluations in three classical examples from the structural reliability literature.

An adaptive stopping rule is used in the BSS algorithm to choose the number of evaluations added by the SUR sampling strategy at each stage. Evaluations at intermediate stages are not directly useful in refining the final probability estimate, but their importance must not be overlooked: they make it possible to learn in a robust way the level sets of the limit-state function, and therefore to build a sequence of densities that converges to the boundary of the failure region. Achieving a better understanding of the connection between the number of evaluations spent on intermediate level sets and the robustness of the algorithm is an important perspective for future work. In practice, if the budget of evaluations permits, we recommend running several passes of the BSS algorithm, with decreasing tolerances for the

adaptive stopping rule, to make sure that no failure mode has been missed.

The adaptive stopping rule also makes it possible to refine the estimation of the final level set to make sure that the posterior model is good enough with respect to the SMC sample size. Other settings of the stopping rule could, of course, be considered. For instance, BSS could stop when the bias is expected to be of the same order as the standard deviation. Future work will focus on fully automated variants of the BSS algorithm, where the number of evaluations and the SMC sample size would be controlled in order to achieve a prescribed error level.

Appendix A. Computation of the variance in the idealized setting. This section provides a derivation of (17) and (18), together with an explicit expression of the estimated coefficient of variation $\widehat{\delta}_{m,T}$ used in section 4.2.1. Both are obtained in the setting of the idealized BSS algorithm described in section 3.4, where the samples $\mathbb{Y}_t = \{Y_t^1, \ldots, Y_t^m\}$ are assumed i.i.d. (with density q_t) and mutually independent.

Recall from (16) that the BSS estimator can be written as

$$\widehat{\widehat{\alpha}}_T^{\mathrm{BSS}} = \prod_{t=1}^{T} \frac{\widehat{\widehat{\alpha}}_t^{\mathrm{BSS}}}{\widehat{\widehat{\alpha}}_{t-1}^{\mathrm{BSS}}} = \prod_{t=1}^{T} \left(\frac{1}{m} \sum_{j=1}^{m} \frac{g_t(Y_{t-1}^j)}{g_{t-1}(Y_{t-1}^j)} \right) = \prod_{t=1}^{T} \widehat{\widehat{p}}_t^{\mathrm{BSS}}, \tag{25}$$

where we have set, for all $t \in \{1, \ldots, T\}$,

$$\widehat{\widehat{p}}_t^{\mathrm{BSS}} = \frac{1}{m} \sum_{j=1}^{m} \frac{g_t(Y_{t-1}^j)}{g_{t-1}(Y_{t-1}^j)}.$$

Observe that the random variables $\widehat{\widehat{p}}_t^{\mathrm{BSS}}$ are independent, with mean

$$\mathsf{E}\left(\widehat{\widehat{p}}_t^{\mathrm{BSS}}\right) = \int_{\mathbb{X}} \frac{g_t}{g_{t-1}} q_{t-1} = \int_{\mathbb{X}} \frac{g_t}{g_{t-1}} \frac{g_{t-1}\,\pi_{\mathbb{X}}}{\widehat{\alpha}_{t-1}^{\mathrm{B}}} = \frac{\widehat{\alpha}_t^{\mathrm{B}}}{\widehat{\alpha}_{t-1}^{\mathrm{B}}} = \widehat{p}_t^{\mathrm{B}}$$

and variance

$$\begin{aligned}
\mathrm{var}\left(\widehat{\widehat{p}}_t^{\mathrm{BSS}}\right) &= \frac{1}{m}\, \mathrm{var}\left(\frac{g_t\left(Y_{t-1}^1\right)}{g_{t-1}\left(Y_{t-1}^1\right)} \right) \\
&= \frac{1}{m} \left[\int_{\mathbb{X}} \frac{g_t^2}{g_{t-1}^2} \frac{g_{t-1}\,\pi_{\mathbb{X}}}{\widehat{\alpha}_{t-1}^{\mathrm{B}}} - \left(\frac{\widehat{\alpha}_t^{\mathrm{B}}}{\widehat{\alpha}_{t-1}^{\mathrm{B}}} \right)^2 \right] \\
&= \frac{1}{m} \left[\frac{1}{\widehat{\alpha}_{t-1}^{\mathrm{B}}} \int_{\mathbb{X}} \frac{g_t^2}{g_{t-1}}\, \pi_{\mathbb{X}} - \left(\frac{\widehat{\alpha}_t^{\mathrm{B}}}{\widehat{\alpha}_{t-1}^{\mathrm{B}}} \right)^2 \right] \\
&= \frac{1}{m}\, (\widehat{p}_t^{\mathrm{B}})^2\, \kappa_t,
\end{aligned}$$

where κ_t is defined by (18). Therefore, the coefficients of variation $\delta_{m,t}$ of the sequence of estimators $\widehat{\widehat{\alpha}}_t^{\mathrm{BSS}}$ obey the recurrence relation $\delta_{m,t}^2 = \frac{1}{m}\kappa_t + (1 + \frac{1}{m}\kappa_t)\, \delta_{m,t-1}^2$, and we conclude that

$$\delta_{m,T}^2 = \frac{1}{m}\kappa_T + \left(1 + \frac{1}{m}\kappa_T\right) \frac{1}{m}\kappa_{T-1}$$

$$+\left(1+\frac{1}{m}\kappa_T\right)\left(1+\frac{1}{m}\kappa_{T-1}\right)\frac{1}{m}\kappa_{T-2}+\cdots$$
$$=\frac{1}{m}\sum_{t=1}^{T}\kappa_t+O\left(\frac{1}{m^2}\right),$$

which proves (17)–(18). We construct an estimator of the coefficient of variation recursively, using the relation

$$\widehat{\delta}^2_{m,t}=\frac{1}{m}\widehat{\kappa}_t+\left(1+\frac{1}{m}\widehat{\kappa}_t\right)\widehat{\delta}^2_{m,t-1},$$

with

$$\widehat{\kappa}_t=\left(\widehat{\widehat{p}}_t^{\,\mathrm{BSS}}\right)^{-2}\cdot\frac{1}{m}\sum_{j=1}^{m}\left(\frac{g_t\left(Y_{t-1}^j\right)}{g_{t-1}\left(Y_{t-1}^j\right)}-\widehat{\widehat{p}}_t^{\,\mathrm{BSS}}\right)^2.$$

REFERENCES

[1] A. Arnaud, J. Bect, M. Couplet, A. Pasanisi, and E. Vazquez, *Évaluation d'un risque d'inondation fluviale par planification séquentielle d'expériences*, in 42èmes Journées de Statistique (JdS 2010), Marseille, France, 2010.

[2] S.-K. Au and J. L. Beck, *Estimation of small failure probabilities in high dimensions by subset simulation*, Probab. Eng. Mech., 16 (2001), pp. 263–277.

[3] Y. Auffray, P. Barbillon, and J.-M. Marin, *Bounding rare event probabilities in computer experiments*, Comput. Statist. Data Anal., 80 (2014), pp. 153–166.

[4] M. Balesdent, J. Morio, and J. Marzat, *Kriging-based adaptive Importance Sampling algorithms for rare event estimation*, Struct. Saf., 44 (2013), pp. 1–10.

[5] M. J. Bayarri, J. O. Berger, E. S. Calder, K. Dalbey, S. Lunagomez, A. K. Patra, E. B. Pitman, E. T. Spiller, and R. L. Wolpert, *Using statistical and computer models to quantify volcanic hazards*, Technometrics, 51 (2009), pp. 402–413.

[6] J. Bect, D. Ginsbourger, L. Li, V. Picheny, and E. Vazquez, *Sequential design of computer experiments for the estimation of a probability of failure*, Stat. Comput., 22 (2012), pp. 773–793.

[7] J. Bect, E. Vazquez, et al., *STK: A Small (Matlab/Octave) Toolbox for Kriging*, Release 2.4.2, http://kriging.sourceforge.net (20 May 2017).

[8] J.-M. Bourinet, *Ferum* 4.1 *User's Guide*, http://www.ifma.fr/FERUM (2010).

[9] J.-M. Bourinet, *Rare-event probability estimation with adaptive support vector regression surrogates*, Reliab. Eng. Syst. Saf., 150 (2016), pp. 210–221.

[10] J.-M. Bourinet, F. Deheeger, and M. Lemaire, *Assessing small failure probabilities by combined subset simulation and support vector machines*, Struct. Saf., 33 (2011), pp. 343–353.

[11] C.-E. Bréhier, L. Goudenège, and L. Tudela, *Central limit theorem for adaptive multilevel splitting estimators in an idealized setting*, in Monte Carlo and Quasi-Monte Carlo Methods (MCQMC 2014), R. Cools and D. Nuyens, eds., Springer, Cham, Switzerland, 2016.

[12] F. Cadini, F. Santos, and E. Zio, *An improved adaptive kriging-based importance technique for sampling multiple failure regions of low probability*, Reliab. Eng. Syst. Saf., 131 (2014), pp. 109–117.

[13] C. Cannamela, J. Garnier, and B. Iooss, *Controlled stratification for quantile estimation*, Ann. Appl. Stat., 2 (2008), pp. 1554–1580.

[14] F. Cérou, P. Del Moral, T. Furon, and A. Guyader, *Sequential Monte Carlo for rare event estimation*, Stat. Comput., 22 (2012), pp. 795–808.

[15] C. Chevalier, J. Bect, D. Ginsbourger, E. Vazquez, V. Picheny, and Y. Richet, *Fast parallel kriging-based stepwise uncertainty reduction with application to the identification of an excursion set*, Technometrics, 56 (2013), pp. 455–465.
[16] C. Chevalier, D. Ginsbourger, J. Bect, and I. Molchanov, *Estimating and quantifying uncertainties on level sets using the Vorob'ev expectation and deviation with Gaussian process models*, in mODa 10—Advances in Model-Oriented Design and Analysis, Contrib. Statist., Springer, Heidelberg, 2013, pp. 35–43.
[17] N. Chopin, *A sequential particle filter method for static models*, Biometrika, 89 (2002), pp. 539–552.
[18] E. De Rocquigny, N. Devictor, S. Tarantola, et al., *Determination of the risk due to personal electronic devices (PEDs) carried out on radio-navigation systems aboard aircraft*, in Uncertainty in Industrial Practice: A Guide to Quantitative Uncertainty Management, John Wiley & Sons, Chichester, UK, 2008, pp. 65–80.
[19] E. De Rocquigny, N. Devictor, S. Tarantola, eds., *Uncertainty in Industrial Practice: A Guide to Quantitative Uncertainty Management*, John Wiley & Sons, Chichester, UK, 2008.
[20] F. Deheeger, *Couplage mécano-fiabiliste: ^{2}SMART—Méthodologie d'apprentissage stochastique en fiabilité*, Ph.D. thesis, Université Blaise Pascal, Clermont-Ferrand II, France, 2008.
[21] P. Del Moral, A. Doucet, and A. Jasra, *Sequential Monte Carlo samplers*, J. R. Stat. Soc. Ser. B Stat. Methodol., 68 (2006), pp. 411–436.
[22] P. Del Moral, A. Doucet, and A. Jasra, *An adaptive sequential Monte Carlo method for approximate Bayesian computation*, Stat. Comput., 22 (2012), pp. 1009–1020.
[23] P. Diaconis and S. Holmes, *Three examples of Monte-Carlo Markov chains: At the interface between statistical computing, computer science, and statistical mechanics*, in Discrete Probability and Algorithms (Minneapolis, MN, 1993), IMA Vol. Math. Appl. 72, Springer, New York, 1995, pp. 43–56.
[24] R. Douc and O. Cappé, *Comparison of resampling schemes for particle filtering*, in Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis (ISPA), 2005, pp. 64–69.
[25] R. Douc and E. Moulines, *Limit theorems for weighted samples with applications to sequential Monte Carlo methods*, The Annals of Statistics, 36 (2008), pp. 2344–2376.
[26] V. Dubourg, *Adaptive Surrogate Models for Reliability Analysis and Reliability-based Design Optimization*, Ph.D. thesis, Université Blaise Pascal, Clermont II, France, 2011.
[27] V. Dubourg, F. Deheeger, and B. Sudret, *Metamodel-based importance sampling for the simulation of rare events*, in Applications of Statistics and Probability in Civil Engineering, CRC Press/Balkema, Leiden, The Netherlands, 2011, pp. 661–668.
[28] V. Dubourg, F. Deheeger, and B. Sudret, *Metamodel-based importance sampling for structural reliability analysis*, Probab. Eng. Mech., 33 (2013), pp. 47–57.
[29] V. Dubourg, B. Sudret, and J.-M. Bourinet, *Reliability-based design optimization using kriging surrogates and subset simulation*, Struct. Multidiscip. Optim., 44 (2011), pp. 673–690.
[30] B. Echard, N. Gayton, and M. Lemaire, *AK-MCS: An active learning reliability method combining kriging and Monte Carlo simulation*, Struct. Saf., 33 (2011), pp. 145–154.
[31] B. Echard, N. Gayton, M. Lemaire, and N. Relun, *A combined importance sampling and kriging reliability method for small failure probabilities with time-demanding numerical models*, Reliab. Eng. Syst. Saf., 111 (2013), pp. 232–240.
[32] P. Feliot, J. Bect, and E. Vazquez, *A Bayesian approach to constrained single- and multi-objective optimization*, J. Global Optim., 67 (2017), pp. 97–133.
[33] Free Software Foundation, *GNU Lesser General Public License, version* 2.1, http://www.gnu.org/licenses/old-licenses/lgpl-2.1.html (February 1999).
[34] E. Garcia, *Electromagnetic compatibility uncertainty, risk, and margin management*, IEEE Trans. Electromagn. Compat., 52 (2010), pp. 3–10.
[35] N. Gayton, J. M. Bourinet, and M. Lemaire, *CQ2RS: A new statistical approach to the response surface method for reliability analysis*, Struct. Saf., 25 (2003), pp. 99–121.
[36] P. Glasserman, P. Heidelberger, P. Shahabuddin, and T. Zajic, *Multilevel splitting for estimating rare event probabilities*, Oper. Res., 47 (1999), pp. 585–600.
[37] X. Huang, J. Chen, and H. Zhu, *Assessing small failure probabilities by AK-SS: An active learning method combining kriging and subset simulation*, Struct. Saf., 59 (2016), pp. 86–95.

[38] M. E. Johnson, L. M. Moore, and D. Ylvisaker, *Minimax and maximin distance designs*, J. Statist. Plann. Inference, 26 (1990), pp. 131–148.
[39] S. N. Jonkman, M. Kok, and J. K. Vrijling, *Flood risk assessment in the Netherlands: A case study for dike ring South Holland*, Risk Anal., 28 (2008), pp. 1357–1374.
[40] L. Li, *Sequential Design of Experiments to Estimate a Probability of Failure*, Ph.D. thesis, Supélec, Gif-sur-Yvette, France, 2012; available online at https://tel.archives-ouvertes.fr/tel-00765457.
[41] L. Li, J. Bect, and E. Vazquez, *Bayesian subset simulation: A kriging-based subset simulation algorithm for the estimation of small probabilities of failure*, in Proceedings of the PSAM 11 and ESREL 2012 Conference on Probabilistic Safety Assessment, Helsinki, Finland, 2012.
[42] J. S. Liu, *Monte Carlo Strategies in Scientific Computing*, Springer, New York, 2008.
[43] J. L. Loeppky, J. Sacks, and W. J. Welch, *Choosing the sample size of a computer experiment: A practical guide*, Technometrics, 51 (2009), pp. 366–376.
[44] R. E. Melchers, *Structural Reliability: Analysis and Prediction*, 2nd ed., John Wiley & Sons, New York, 1999.
[45] M. D. Morris and T. J. Mitchell, *Exploratory designs for computational experiments*, J. Statist. Plann. Inference, 43 (1995), pp. 381–402.
[46] J. Oakley, *Estimating percentiles of uncertain computer code outputs*, J. Roy. Statist. Soc. Ser. C, 53 (2004), pp. 83–93.
[47] P. P. O'Connor and A. Kleyner, *Practical Reliability Engineering*, John Wiley & Sons, Chichester, UK, 2012.
[48] M. Rausand and A. Høyland, *System Reliability Theory: Models, Statistical Methods, and Applications*, 2nd ed., John Wiley & Sons, Hoboken, NJ, 2004.
[49] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*, 2nd ed., Springer-Verlag, New York, 2004.
[50] T. J. Santner, B. J. Williams, and W. I. Notz, *The Design and Analysis of Computer Experiments*, Springer-Verlag, New York, 2003.
[51] M. L. Stein, *Interpolation of Spatial Data: Some Theory for Kriging*, Springer-Verlag, New York, 1999.
[52] E. Vazquez and J. Bect, *A sequential Bayesian algorithm to estimate a probability of failure*, in Proceedings of the 15th IFAC Symposium on System Identification (SYSID 2009), Saint-Malo, France, 2009.
[53] J. Villemonteix, E. Vazquez, and E. Walter, *An informational approach to the global optimization of expensive-to-evaluate functions*, J. Global Optim., 44 (2009), pp. 509–534.
[54] P. H. Waarts, *Structural Reliability Using Finite Element Methods*, Ph.D. thesis, Delft University of Technology, Delft, The Netherlands, 2000.
[55] E. Zio and N. Pedroni, *Estimation of the functional failure probability of a thermal-hydraulic passive system by Subset Simulation*, Nucl. Eng. Des., 239 (2009), pp. 580–599.
[56] K. M. Zuev, J. L. Beck, S.-K. Au, and L. S. Katafygiotis, *Bayesian post-processor and other enhancements of Subset Simulation for estimating failure probabilities in high dimensions*, Comput. & Structures, 92–93 (2012), pp. 283–296.