History matching with subset simulation

Z. T. Gong[a], F. A. DiazDelaO[b], P. O. Hristov[c,*], M. Beer[d]

[a]*CRRC Qingdao Sifang Co., LTD., China.*
[b]*Clinical Operational Research Unit, Department of Mathematics, University College London, London WC1H 0BT, UK*
[c]*Institute for Risk and Uncertainty, School of Engineering, University of Liverpool, Liverpool L69 7ZF, UK*
[d]*Institute for Computer Science in Civil Engineering, Leibniz University, Hannover 30167, Germany*

Abstract

Computational cost often hinders the calibration of complex computer models. In this context, history matching is becoming a widespread calibration strategy, with applications in many disciplines. History matching (HM) uses a statistical approximation - also known as an emulator - to the model output, in order to mitigate computational cost. The process starts with an observation of a physical system. It then produces progressively more accurate emulators to determine a non-implausible domain: a subset of the input space that provides a good agreement between the model output and the data, conditional on the model structure, the sources of uncertainty and an implausibility measure. In HM, it is essential to generate samples from the non-implausible domain, in order to run the model and train the emulator until a stopping condition is met. However, this sampling can be very challenging, since the non-implausible domain can become orders of magnitude smaller than the original input space very quickly. This paper proposes a solution to this problem using subset simulation, a rare event sampling technique that works efficiently in high dimensions. The proposed approach is demonstrated via calibration and robust design examples from the field of aerospace engineering.

Keywords: History matching, Subset simulation, Gaussian process emulation, Robust design.

*Corresponding author.
Email address: `p.hristov@liv.ac.uk` (P. O. Hristov)

1. Introduction

The use of computer models (also known as simulators) to study complex systems and environments is indispensable in modern scientific research. The reliability of these models depends critically on how well they are calibrated to experimental data. Otherwise, model-based decisions run the risk of being misguided or ill-informed. One of the challenges of model calibration is that several sources of uncertainty must be taken into account. This uncertainty originates (for instance) due to process idealisations, model assumptions and computational cost. In order to provide evidence of predictive reliability, it is essential that any model is calibrated taking into account these sources of uncertainty.

Typically, high-fidelity computer models of complex phenomena are computationally expensive. In the context of uncertainty quantification, this characterisation usually describes models whose evaluation time prohibits their repeated use in any form of sampling-based analysis. This feature presents a challenge to classical calibration techniques, which require a considerable number of simulator runs to identify an acceptable match between model and data. Furthermore, the analyst might face an added challenge if not only the model, but also the generation of experimental data, is expensive or unfeasible. Despite the importance and necessity of efficient calibration methods for complex computer codes, their development has lagged behind their application [1]. Instead, simple goodness-of-fit measures such as distance-based methods and least squares (see *e.g.* [2]) or likelihood functions (consult [3] and references therein) are applied. Neither of these may be suitable when computational cost and high dimensional input are considered. This is due to the fact that goodness-of-fit measures typically require large data sets to achieve a reliable quantification of the degree of agreement between observations and simulator realizations. Likewise, the likelihood of complex simulators is usually intractable and approximations may be required (see *e.g.* [4]).

History matching (HM) [5, 6] is a form of calibration for complex and computationally expensive numerical models. It uses Bayesian emulation [7] to tackle computational cost. Emulation means building a statistical approximation to the original simulator, thus allowing affordable inference about its output. History matching also defines an implausibility measure, which is used to reduce the input space by finding an input subspace that provides a reasonable match between the model output and experimental data, given the model structure and various sources of uncertainty. This input space reduction is achieved by building progressively more accurate emulators, which in practice results in HM becoming an iterative process. The resulting input subspace is known in the literature as non-implausible domain, non-implausible set or Not-Ruled-Out-Yet (NROY) space [8]. History matching has been successfully applied in epidemiology [1], galaxy formation modeling [8], oil reservoir analysis [9] and large climate systems modelling [10], amongst many other applications.

The sequential generation of samples from the non-implausible domain at every HM iteration has remained an open and complex problem. This mainly stems from the fact that the non-implausible domain can be orders of magnitude smaller than the original input space [11]. A notable example of a field of study in which a similar challenge is encountered is *engineering reliability analysis*. The main aim of this type of reliability analysis is to identify the conditions under which a physical system fails. In that context, failure means that the demand has exceeded the capacity of the system, according to a model of the system and a criterion guided by expert knowledge. Reliability analysis aims at generating samples from the *failure set*, that is, the set of model input configurations that lead to failure. This allows the characterisation of different modes in which the system can fail and to estimate the probability of failure. If an engineering system is well-designed and the model is a good representation of the system, the volume of the failure domain is expected to be orders of magnitude smaller compared to the input space. Since this can also be the case for the non-implausible domain within HM, this opens the prospect of treating it as if it were a failure set within reliability analysis.

Subset simulation (SuS) [12] is a widely used technique in engineering reliability computations and rare event simulation. Unlike direct Monte Carlo, SuS models a rare event, which has a small failure probability, as contained in a nested sequence of less-rare events. Eventually, the probability of failure can be computed as the product of larger conditional probabilities given the occurrence of each preceding event. Markov chain Monte Carlo (MCMC) is used to generate the conditional samples that belong to the intermediate failure events. Based on this strategy, SuS generates samples selectively, to efficiently populate the target failure set. Given the potential disparity in the size of the original input space and the non-implausible domain (in the context of HM); and the potential disparity in the size of the original input space and the failure domain (in the context of reliability analysis), this paper proposes the use of SuS as an efficient sampler of the non-implausible domain within each wave of HM.

The remainder of the paper is organized as follows. Section 2 presents an overview of HM. Section 3 reviews the details of SuS. Section 4 presents the proposed approach, in which SuS is used to sample from the non-implausible domain in HM. The resulting procedure is demonstrated in a calibration context in Section 5 and in an industrial context for robust aircraft design in Section 6. Finally, section 7 provides some conclusions.

2. History matching

2.1. Procedure overview

A rigorous description of the relationship between a model and the underlying physical process requires the identification and inclusion of different sources of uncertainty. Let y denote the true value of the physical process. A modeller analysing the process can only observe a noisy version of this value. Let $z = y + \epsilon_{me}$ be this noisy observation, where ϵ_{me} is measurement error. This type of error, also called *observational uncertainty*, can be thought of as a random variable with zero mean and finite variance. The modeller then represents the physical process through a numerical simulator, which defines an input-output mapping $\eta : \mathbb{R}^d \rightarrow \mathbb{R}$. Let $\eta(\mathbf{x})$ denote the simulator output as a function of some input vector $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$. Even if all model parameters were known exactly, the process y cannot be represented perfectly. This is due to unavoidable modelling assumptions, simplifications, or incomplete knowledge of the underlying physics. This disparity is known as *model discrepancy* [13] and is denoted by ϵ_{md}. The modelled physical process can therefore be described by $y = \eta(\mathbf{x}_c) + \epsilon_{md}$, where $\mathbf{x}_c$ is an input configuration, such that $\eta(\mathbf{x}_c)$ summarizes all of the information the simulator carries about the system. Finally[1], the value of $\eta(\mathbf{x})$ is unknown until the model is evaluated at the input combination $\mathbf{x}$. When the model is computationally expensive, the analyst will only be able to run the model in a limited number

[1]If the simulator is stochastic in nature, *i.e.* evaluating η at a fixed input combination $\mathbf{x}$ returns a different output value, $\eta(\mathbf{x})$ every time, another source of uncertainty called *ensemble variation* can be added. See discussion in Section 2.3

of input configurations, which induces another source of uncertainty, called *code uncertainty* [14].

Given the sources of uncertainty introduced above, HM is designed to explore the input space $\mathcal{X}$ and discard regions which are unlikely to produce the measured system response. This is achieved through: (i) the use of an *implausibility measure*, which quantifies the distance between the measurement z and the output of the model $\eta(\mathbf{x})$, normalized by the different sources of uncertainty; and (ii) a *Bayesian emulator* to alleviate the cost of running the complex model. The technical details behind Bayesian emulation are given in Section 2.2. Due to the use of an emulator, HM becomes an iterative procedure in practice. At each iteration, also called *wave* [8], HM builds increasingly accurate emulators and the implausibility measure provides a rule to discard the subsets of the input domain that are unlikely to produce an acceptable match between model output and observed data. Once the non-implausible domain in the current wave is identified, HM selects a handful of points at which to evaluate $\eta(\cdot)$. This data is then used to refine the approximation provided by the emulator in the non-implausible domain. The process continues until a predefined stopping condition is satisfied.

In contrast to conventional calibration methods, which seek a single point $\mathbf{x}_c$, HM identifies a set of input combinations that are likely to produce a match between model prediction and measured data, within some level of uncertainty. Furthermore, whereas standard Bayesian calibration will always find a posterior distribution for acceptable inputs, HM can discover that the model is an inadequate representation of the physical process by returning an empty non-implausible domain. Thus, some authors regard HM as a pre-calibration strategy [8].

Generating an initial design to run the simulator is the first step for a typical HM workflow. Initially, the whole input domain $\mathcal{X}$ is considered. To explore the model output across the input domain with as few data points as possible, a design with good space-filling qualities is generated. A Latin hypercube sampling (LHS) plan [15] is often used. As suggested in [16], a common choice is to have the number of sample points equal to $n = 10d$, where d is the dimension of the input. In practice, the choice of n is often determined by the computational budget. Once an LHS design is specified, the simulator $\eta(\cdot)$ is evaluated at each input point $\mathbf{x}_i$, producing a corresponding output $\eta(\mathbf{x}_i)$ for $i = 1 \dots n$. The resulting input-output pairs constitute an experimental design denoted by $\mathcal{D} = \{\mathbf{x}_i, \eta(\mathbf{x}_i)\}_{i=1}^{n}$.

2.2. Emulation

An emulator is a statistical approximation to the output of an expensive computer model. Gaussian process emulators [17] are widely used to infer the output of expensive simulators based on a small number of training runs. In this case, the experimental design $\mathcal{D}$ defined in the previous subsection provides such runs. Emulators provide a full probabilistic characterisation of the output at untried input configurations. Their widespread use is due to the fact that they not only provide a fast surrogate to the output of the simulator, but also produce an analytic expression for the uncertainty arising due to the limited number of model evaluations (referred to in Section 2.1 as code uncertainty). The applications of Gaussian process emulators span many fields of science and technology. Some recent examples include modelling submarine sliding and tsunami formation, [18] and reducing the cost of engineering reliability analysis [19].

It is important to note that the original HM approach presented in [20] and [21] is based on the concepts of Bayes linear emulation [22], which uses mathematical expectation, instead of probability, as a primitive. This further aids the mitigation of computational cost. In this paper however, we assume a Bayesian emulator is of the form:

$$\hat{\eta}(\mathbf{x}) = h(\mathbf{x})^\intercal \boldsymbol{\beta} + Z(\mathbf{x}) \tag{1}$$

where $h : \mathbb{R}^d \to \mathbb{R}^q$ is a vector of known functions, $\boldsymbol{\beta} \in \mathbb{R}^q$ is a vector of coefficients and $Z(\mathbf{x})$ is a zero mean Gaussian process with covariance function $\sigma^2 c(\mathbf{x}, \mathbf{x}'; \boldsymbol{\psi})$, also known as covariance kernel [23]. The regression term $h(\mathbf{x})^\intercal \boldsymbol{\beta}$ models the global trend of the output, whilst the Gaussian process models local variations. The covariance of the Gaussian process at two distinct inputs, $\mathbf{x}$ and $\mathbf{x}'$, is the product of a process variance parameter σ^2 and a positive semi-definite correlation function $c(\cdot, \cdot; \boldsymbol{\psi})$, parameterised by $\boldsymbol{\psi}$. In this work, the Matérn (5/2) correlation function [24] is employed. This function was chosen because it is stationary and because it exhibits a moderate degree of smoothness, which is suitable for many applications [25]. The Matérn (5/2) correlation function has the following mathematical expression:

$$c(\mathbf{x}, \mathbf{x}'; \boldsymbol{\psi}) = \left(1 + \frac{\sqrt{5}\delta(\mathbf{x}, \mathbf{x}')}{\boldsymbol{\psi}} + \frac{5\delta^2(\mathbf{x}, \mathbf{x}')}{3\boldsymbol{\psi}^2}\right) \exp\left(-\frac{\sqrt{5}\delta(\mathbf{x}, \mathbf{x}')}{\boldsymbol{\psi}}\right) \tag{2}$$

where $\delta(\mathbf{x}, \mathbf{x}')$ is the Euclidean distance between $\mathbf{x}$ and $\mathbf{x}'$.

In order to estimate the values of each of the parameters $\boldsymbol{\beta}$, σ^2 and $\boldsymbol{\psi}$, prior probability distributions can be imposed, and their posterior distributions can be computed by conditioning on the training runs $\mathcal{D}$. A weak prior [26] can be used for $\boldsymbol{\beta}$ and σ^2, namely

$$p(\boldsymbol{\beta}, \sigma^2) \propto \sigma^{-2} \tag{3}$$

Conditional on $\mathcal{D}$, the two parameters are distributed according to a normal-inverse-gamma distribution [27], with expected values given by

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{H}^\intercal \boldsymbol{C}^{-1} \boldsymbol{H})^{-1} \boldsymbol{H}^\intercal \boldsymbol{C}^{-1} \boldsymbol{f} \tag{4}$$

$$\hat{\sigma}^2 = \frac{\boldsymbol{f}^\intercal (\boldsymbol{C}^{-1} - \boldsymbol{C}^{-1} \boldsymbol{H} (\boldsymbol{H}^\intercal \boldsymbol{C}^{-1} \boldsymbol{H}^{-1}) \boldsymbol{H}^\intercal \boldsymbol{C}^{-1}) \boldsymbol{f}}{n - q - 2} \tag{5}$$

where $\mathbf{H} = [h(\mathbf{x}_1), \ldots, h(\mathbf{x}_n)]^\intercal$, $\mathbf{C} \in \mathbb{R}^{n \times n}$ such that $C_{ij} = c(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\psi})$, and $\boldsymbol{f} = [\eta(\mathbf{x}_1), \ldots, \eta(\mathbf{x}_n)]^\intercal$. The posterior distribution of $\boldsymbol{\psi}$ can be computed using a full Bayesian approach [28, 29]. Due to the potentially high computational cost of Bayesian computations, some authors instead prefer resorting to maximum likelihood estimation [30].

It can be shown [17] that, conditional on the parameter estimates in Eq. (4) and Eq. (5), the posterior predictive distribution for the simulator output is

$$\eta(\mathbf{x}) \sim m(\mathbf{x}) + \sigma_c(\mathbf{x}) t_{n-q} \tag{6}$$

where t_{n-q} is the Student's-t distribution with $n - q$ degrees of freedom. The emulator's posterior mean $m(\mathbf{x})$ and posterior variance $\sigma_c^2(\mathbf{x})$ are given by

$$m(\mathbf{x}) = h(\mathbf{x})^\intercal \hat{\boldsymbol{\beta}} + t(\mathbf{x})^\intercal \mathbf{C}^{-1} (\boldsymbol{f} - \mathbf{H} \hat{\boldsymbol{\beta}}) \tag{7}$$

$$\begin{aligned} \sigma_c^2(\mathbf{x}) = \hat{\sigma}^2 \big[& c(\mathbf{x}, \mathbf{x}) - t(\mathbf{x})^\intercal \boldsymbol{C}^{-1} t(\mathbf{x}) \\ & + (h(\mathbf{x})^\intercal - t(\mathbf{x})^\intercal \boldsymbol{C}^{-1} \boldsymbol{H})(\boldsymbol{H}^\intercal \boldsymbol{C}^{-1} \boldsymbol{H})^{-1} \\ & \times (h(\mathbf{x})^\intercal - t(\mathbf{x})^\intercal \boldsymbol{C}^{-1} \boldsymbol{H})^\intercal \big] \end{aligned} \tag{8}$$

where $t(\mathbf{x}) = [c(\mathbf{x}, \mathbf{x}_1; \boldsymbol{\psi}), \ldots, c(\mathbf{x}, \mathbf{x}_n; \boldsymbol{\psi})]^\intercal$.

2.3. Implausibility threshold

Let $I : \mathbb{R}^d \to \mathbb{R}$ be a function that measures the implausibility that an input configuration $\mathbf{x}$ will produce a simulator output matching the experimental observation z. When the simulator is expensive, this implausibility can be defined as the distance between z and the emulator mean, $m(\mathbf{x})$. This distance can be normalised in order to express it in terms of the number of standard deviations of the overall uncertainty [1]. This results in the following expression:

$$I(\mathbf{x}) = \frac{|z - m(\mathbf{x})|}{\sqrt{\sigma_{me}^2 + \sigma_{md}^2 + \sigma_c^2(\mathbf{x})}} \tag{9}$$

where σ_{me}^2 and σ_{md}^2 are respectively the variances of the measurement error term ϵ_{me} and the model discrepancy term ϵ_{md}, as defined in Section 2.1. The term $\sigma_c^2(\mathbf{x})$, corresponding to the (current level) emulator's posterior predictive variance in Eq. (8), quantifies the code uncertainty. In this work, the simulator is assumed deterministic: for the same input configuration, the output is fixed. It is possible to add a term in the denominator of equation Eq. (9) to account for *ensemble variability* in case the simulator is stochastic [1].

Suppose that the implausibility measure $I(\cdot)$ is evaluated at a particular sample point $\mathbf{x}^*$. For $I(\cdot)$ to be meaningful, it should be true that the smaller the value of $I(\mathbf{x}^*)$, the more likely it is that $\mathbf{x}^*$ yields an output that matches the experimental data within the specified level of uncertainty. A criterion for setting an implausibility threshold is provided by Pukelsheim's rule [31], which states that if a random variable X has a unimodal distribution with mean μ and standard deviation σ, such as the Student's-t in Eq. (6), then

$$P(|X - \mu| > 3\sigma) \leq 0.05 \tag{10}$$

Hence, a natural criterion for accepting $\mathbf{x}^*$ as a non-implausible input combination is $I(\mathbf{x}^*) \leq 3$. Sample points that fail this criterion are considered implausible. The new wave of HM begins by sampling from the non-implausible domain identified using this rule.

2.4. Sampling design for new waves

The initial design $\mathcal{D}$ to train the emulator can be generated through LHS. After the first wave, sampling from the non-implausible domain can become challenging very quickly. This can be due to, for example, rapid reduction in its size. An additional challenge is that the non-implausible domain may become disconnected or exhibit a complex topology, which can further complicate the sampling procedure.

The most intuitive strategy to deal with this problem is to generate an LHS plan on the whole input space $\mathcal{X}$, then discard all implausible points, determined by $I(\mathbf{x})$. This simple acceptance-rejection strategy can quickly become inefficient if the non-implausible domain reduces to a small fraction of $\mathcal{X}$. Multiple solutions have been proposed in the literature. An implausibility-driven evolutionary Monte Carlo algorithm (IDEMC) was proposed in [32]. This generates uniform designs for the target space using an implausibility ladder, which might be challenging to determine. Another approach, discussed in [1] is to generate normally distributed samples centered on each point from the non-implausible domain of the current wave. The covariance matrix of the non-implausible samples is scaled to give a relatively flat distribution. The challenge in this approach is to determine an optimal scaling factor, which determines the rate at which the input space is explored. As an alternative to the above methods, this paper proposes to sample the non-implausible domain using subset simulation, a rare-event sampling method used in engineering reliability analysis.

3. Subset simulation

Subset simulation (SuS) is an advanced Monte Carlo method that efficiently estimates probabilities of failure of engineering systems [33]. Let $g : \mathbb{R}^d \to \mathbb{R}$ be a *performance function* used to model a physical system. That is, $g(\cdot)$ encodes all the available information about the system's behaviour and attributes, such as its geometry, material properties and loads. When the system is large and complex, specifying deterministic inputs of the performance function can be unrealistic. Thus, the inputs $\mathbf{x}$ can be modelled as distributed according to a joint probability density function (PDF) $\pi(\mathbf{x})$[2]. The output of $g(\cdot)$ then becomes a random variable $Y = g(\mathbf{x})$, and failure is formulated as the exceedance of this random variable over a prescribed threshold $b \in \mathbb{R}$. The main interest of reliability analysis is to determine the *probability of failure* $P(Y > b)$, given by

[2]Even though precise characterisation for the inputs can be specified, one may want to investigate different scenarios by varying those inputs according to some probability distributions

$$P_F = P(Y > b) = \int \pi(\mathbf{x})\, \mathbf{1}(\mathbf{x} \in \mathcal{F})\, d\mathbf{x}, \tag{11}$$

where F denotes the *failure event* defined as

$$F = \{Y > b\} = \{\mathbf{x} \in \mathcal{F}\} = \{\mathbf{x} \in \mathbb{R}^d : g(\mathbf{x}) > b\} \tag{12}$$

and $\mathcal{F} \subseteq \mathbb{R}^d$ is the failure region of the input space. The indicator function $\mathbf{1}(\cdot)$ is equal to 1 if $\mathbf{x} \in \mathcal{F}$ and is zero otherwise.

The idea behind SuS is to model F as contained in a nested sequence of events $F = F_m \subset F_{m-1} \subset \cdots \subset F_1 \subset F_0 = \{\mathbf{x} \in \mathcal{X}\}$ such that the probability of failure can be computed as

$$P_F = P\left(\bigcap_{i=1}^{m} F_i\right) = P(F_1) \times P(F_2|F_1) \times \cdots \times P(F_m|F_{m-1}) \tag{13}$$

This means that sampling from F is done by sampling progressively from more frequent conditional events. Every intermediate failure event corresponds to an iteration *level* in the SuS algorithm, whereby level 0 corresponds to initial Monte Carlo sampling of the whole input space $\mathcal{X}$. There are two important parameters in the algorithm: the level probability, denoted by p_0 and defined as $p_0 \equiv P(F_i|F_{i-1})$, and the number of samples in each level, N. Both are determined by the modeller, such that $p_0 N$ and $1/p_0$ are integers. The level probability p_0 directly influences the properties of the estimator for P_F. The recommended range to minimise its coefficient of variation is $p_0 \in [0.1, 0.3]$ [34]. The number of samples at each level, N, can be set to achieve a given coefficient of variation in the estimation of P_F. In our experience, however, its prescribed value is mainly driven by the available computational budget. It is worth noting that, in industrial settings, the performance function $g(\cdot)$ is rarely analytical or inexpensive to compute. In practice, it usually consists of one or more expensive computer model. Different authors (see [19] and references therein) have proposed different strategies to tackle this cost, some of which include using the emulators discussed in Section 2.2.

The SuS algorithm proceeds as follows. At the unconditional level 0, SuS starts by generating N independent samples $\mathbf{x}_1, \ldots, \mathbf{x}_N \sim \pi(\mathbf{x})$. The performance function $g(\cdot)$ is evaluated and the corresponding output values are sorted in descending order, resulting in the list $\{b_k^{(0)} : k = 1, \ldots, N\}$. The value $b_k^{(0)}$ gives the estimated output value corresponding to the exceedance probability $p_k^{(0)} = P(Y > b_k^{(0)})$ where

$$p_k^{(0)} = \frac{k}{N}, \qquad k = 1, \ldots, N. \tag{14}$$

The first intermediate failure level, b_1 is defined as the midpoint between $b_{p_0 N}^{(0)}$ and $b_{p_0 N+1}^{(0)}$. This way, the conditional failure relation

$$p_0 = P(Y > b_1) = P(F_1 | F_0), \tag{15}$$

is satisfied. Note that, by construction, the $p_0 N$ top-ranked samples have responses greater or equal to b_1. Thus, they are guaranteed to belong to the first intermediate failure level $\mathcal{F}_1$. The generation of new samples from $\mathcal{F}_1$ is done by exploiting this property. The $p_0 N$ top-ranked samples are used as seeds to generate independent Markov chains from the target density $\pi(\mathbf{x}|F_1) \propto \pi(\mathbf{x})\mathbf{1}(\mathbf{x} \in \mathcal{F}_1)$. This results in generating $N_c = p_0 N$ Markov chains, each with length

$$N_s = \frac{N}{N_c} = \frac{1}{p_0} \tag{16}$$

Since the seeds already belong to the intermediate failure domain $\mathcal{F}_1$, there is no burn-in period, usually required in MCMC simulations to generate a single Markov chain. The MCMC scheme employed in the original SuS algorithm [12], was the *modified Metropolis algorithm*, which uses a component-wise Metropolis-Hastings sampling to generate the Markov chains. Throughout the years, different strategies have been proposed and developed. An account of those strategies can be consulted in [35].

Subset simulation follows the same principle iteratively: the i^{th} level (for $i \geq 1$) is defined as $F_i = \{Y > b_i\}$, where b_i is determined as the midpoint between $b_{N_c}^{(i-1)}$ and $b_{N_c+1}^{(i-1)}$. Thus, at each intermediate failure level, the equation $p_0 = P(F_i | F_{i-1})$ is satisfied. At level i, N_c independent Markov chains are generated from the target density $\pi(\cdot | F_i)$, each with length N_s. The process is repeated until the target threshold level b is reached. As before, let m denote the final intermediate level. The threshold value satisfies $b_m \geq b$ and thus the number of conditional samples with responses greater than b, exceeds N_c. The estimate of the failure probability is derived from Eq. (13), which can be written as

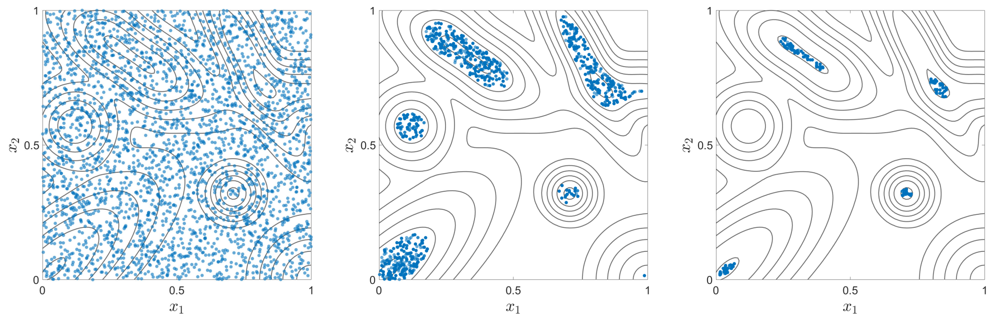
$$\hat{P}_F = p_0^{m-1} \frac{1}{N} \sum_{k=1}^{N} \mathbf{1}(\mathbf{x}_k \in \mathcal{F}_m), \tag{17}$$

where $\frac{1}{N} \sum_{k=1}^{N} \mathbf{1}(\mathbf{x}_k \in \mathcal{F}_m)$ is the estimate of the conditional failure probability at level m.

Subset simulation is capable of efficiently sampling from disconnected failure regions that are, potentially, orders of magnitude smaller than the original input space. In order to illustrate this, consider the performance function $g : [0,1]^2 \to \mathbb{R}$ given by

$$g(\mathbf{x}) = \sum_{i=1}^{9} w_i \phi(\mathbf{x}|\mu_i, \mathbf{C}_i) \tag{18}$$

where each $w_i \in (0,1)$ is a weight, μ_i the mean and $\mathbf{C}_i$ the covariance matrix of the i^{th} Gaussian PDF, $\phi(\mathbf{x}|\mu_i, \mathbf{C}_i)$. The numerical values of these parameters are given in Table A.1 and the level contours of $g(\mathbf{x})$ are shown in Figure 1. Let the failure thershold be $b = 2.75$. The failure domain would then be $\mathcal{F} = \{\mathbf{x} \in [0,1]^2 : g(\mathbf{x}) > 2.75\}$, which results in the disjoint failure set shown in Figure 1(c). The successive subplots in Figure 1 depict how SuS steers the sampling towards $\mathcal{F}$.



(a) Level 0 (b) Level 1 (c) Level 2

Figure 1: Sampling from a small probability event via subset simulation. (a) Samples from the unconditional failure domain $\mathcal{F}_0$ (i.e. the entire input space); (b) samples in the first intermediate failure domain $\mathcal{F}_1 \subseteq \mathcal{F}_0$; (c) samples in the failure domain $\mathcal{F} \subseteq \mathcal{F}_1 \subseteq \mathcal{F}_0$ generated by MCMC.

The example above suggests that a natural analogy can be established between the non-implausible domain introduced in Section 2 and a failure set. Firstly, both are defined by specifying a threshold (for non-implausibility and for failure, respectively). Secondly, both may be significantly smaller than the original input space. Thirdly, they may become disconnected. This motivates treating the non-implausible domain within HM as if it were a failure set. The corresponding sampling can therefore be done using SuS within HM.

4. Proposed Approach

As discussed in the previous section, the main aim of SuS is to estimate the probability of failure given by Eq. (11). In order to do so, the algorithm produces samples within each intermediate failure domain and eventually from the failure domain $\mathcal{F}$. The proposed approach for HM takes advantage of this property, since the prime objective is to eventually sample from the non-implausible domain. It should be noted that SuS has previously been used in the context of calibration, for the estimation of parameter posterior distributions [36, 37]. However, as discussed in Section 2.1, whilst Bayesian calibration always delivers a posterior distribution, HM might determine that the non-implausible domain is empty.

In order to use SuS within HM, sampling is done by treating the non-implausible domain as if it were the failure domain defined by

$$\mathcal{F} = \{\mathbf{x} : I(\mathbf{x}) < 3\} \tag{19}$$

where the implausibility measure $I(\mathbf{x})$, defined in Eq. (9), takes the role of the performance function.

The proposed approach begins with sampling the input domain of the computer model and evaluating it to get an initial data set, $\mathcal{D}_1$. This data set is split and then used to train and validate the initial GPE[3] [27]. At this point, the parameters of SuS are set as per Section 3. It is important to note that the direction of the inequality in $\mathcal{F} = \{\mathbf{x} : I(\mathbf{x}) < 3\}$ is the opposite to that of the inequality in the definition of the failure domain given in Eq. (12). This feature is accounted for by sorting the negative of the values of the implausibility function evaluated at the candidate samples. The algorithm progresses by sequentially discarding regions of the input domain,

[3]If the code is very computationally expensive, the emulator can be validated using cross-validation instead of a separate validation set (see Section 2.1 in [30]).

according to their implausibility $I(\mathbf{x})$, as explained in Section 2.3. When SuS converges, it returns the set $\mathbf{X}_{SuS}$, which belongs to the non-implausible domain. A subset of these samples, which maximises the predictive variance of the GPE at the current level is selected and denoted as $\mathbf{X}_{add}$. Other approaches to selecting $\mathbf{X}_{add}$ exist, such as the maximin strategy outlined in [1]. The GPE for the ℓ^{th} wave of HM is trained using an augmented data set, $\mathcal{D}_\ell = \mathcal{D}_{\ell-1} \cup \{\mathbf{X}_{add}, \eta(\mathbf{X}_{add})\}$. It should be pointed out that, even if the model itself is a highly non-linear function, it becomes smoother in the plausible region as the latter shrinks after each level of HM, which in turn leads to an increase in the accuracy of the emulator [1]. At the same time, the training points become denser. The algorithm terminates once the code uncertainty, quantified through the emulator variance, becomes smaller than the other sources of error. The proposed approach is summarised in Algorithm 1.

The next two sections present applications of the proposed SuS-based HM. In both examples, the modified Metropolis algorithm is used to sample from the intermediate failure domains. This is not a constraint, since any of the MCMC sampling schemes reviewed in [35] could in principle be used. Previous work on the comparison of some of these schemes within SuS-based HM can be found in [38].

5. Calibration: wing weight reduction

This section demonstrates the proposed approach by using HM to calibrate a model of the weight of a light aircraft wing. Weight is a critical factor in aircraft design and ensuring the model at hand can reliably match experimental weights is of vital importance.
The analytical model considered here is derived from historical data and is given by

$$W = 0.036 S_w^{0.758} W_{fw}^{0.0035} \left(\frac{A_w}{\cos^2\Lambda}\right)^{0.6} q^{0.006} \lambda^{0.04} \left(\frac{100 t_c}{\cos\Lambda}\right)^{-0.3} (N_z W_{dg})^{0.49} + S_w W_p \tag{20}$$

Eq. (20) was introduced in its original form in [39]. The last term on the right hand side representing the weight of the paint on the wing was added in [30]. A brief description of the inputs of the model, together with their ranges is provided in Table 1.

A simulated observation for the wing weight was set at $z = 130$lb. A measurement error of ± 5lb was imposed, corresponding to a standard deviation

Algorithm 1 History matching with subset simulation

```
1: Provide an experimental observation $z$ and relevant standard deviations: $\sigma_{me}$ for observational uncertainty and $\sigma_{md}$ for model discrepancy.
2: Set parameter values for SuS: $p_0,\ N$.
3: Define $\mathcal{F} = \{\mathbf{x} : I_\ell(\mathbf{x}) < 3\}$ where $I_\ell(\mathbf{x})$ is the implausibility measure for the emulator at $\ell^{\text{th}}$ wave of HM as defined in Eq. (9).
4: Generate a space-filling plan, $\mathbf{X} \in \mathbb{R}^{n\times d}$ and form $\mathcal{D}_1 = \{\mathbf{x}_i,\ \eta(\mathbf{x}_i)\}_{i=1}^{n}$
5: Train a GPE $\hat{\eta}_1(\mathbf{x}) \sim m_1(\mathbf{x}) + \sigma_{c_1}(\mathbf{x})t_{n-q}$ on $\mathcal{D}_1$ and validate it.
6: $\ell \leftarrow 1$.
7: while $\sigma_{me} < \sigma_{c_\ell}$ and $\sigma_{md} < \sigma_{c_\ell}$ do
8:     Sample from the non-implausible domain using SuS:
9:     Subset simulation
10:        Obtain an MC sample $\mathbf{X}_{SuS} \in \mathbb{R}^{N\times d} \sim \pi(\mathbf{x})$.
11:        $N_F \leftarrow 0$.
12:        $j \leftarrow 0$.
13:        while $N_F < p_0 N$ do
14:            $j = j + 1$.
15:            Evaluate $\hat{\eta}_\ell(\mathbf{X}_{SuS})$.
16:            Compute $I_{SuS} \equiv -I_\ell(\mathbf{X}_{SuS})$ and sort in descending order.
17:            Renumber $\mathbf{X}_{SuS}$ to match the order of $I_{SuS}$.
18:            Select $\left\{\mathbf{x}_{SuS}^{(i)}\right\}_{i=1}^{p_0 N}$ as seeds for MCMC.
19:            Compute intermediate threshold $b_j = \frac{1}{2}\left[I_{SuS}^{(p_0 N)} + I_{SuS}^{(p_0 N+1)}\right]$
20:            Define intermediate failure domain $F_j = \{I_{SuS} > b_j\}$
21:            Obtain a sample, $\mathbf{X}_{SuS}$, from $\pi(\mathbf{x}|F_j)$ using an MCMC scheme.
22:            $N_F = \sum_{i=1}^{N} \mathbf{1}(I_{SuS} > -3)$
23:        end while
24:    $\ell = \ell + 1$
25:    Let $\mathbf{X}_{add}$ be a subset of points from $\mathbf{X}_{SuS}$.
26:    Construct $\mathcal{D}_\ell \leftarrow \mathcal{D}_{\ell-1} \cup \{\mathbf{X}_{add}, \eta(\mathbf{X}_{add})\}$
27:    Train a GPE $\hat{\eta}_\ell(\mathbf{x}) \sim m_\ell(\mathbf{x}) + \sigma_{c_\ell}(\mathbf{x})t_{n-q}$ on $\mathcal{D}_\ell$.
28: end while
```

Table 1: Inputs for the light aircraft wing weight model.

Notation	**Name**	**Range**	**Unit**
S_w	Wing area	$[150, 200]$	ft^2
W_{fw}	Weight of fuel in the wing	$[220, 300]$	lb
A_w	Aspect ratio	$[6, 10]$	-
Λ	Quarter-chord sweep	$[-10, 10]$	deg
q	Dynamic pressure at cruise	$[16, 45]$	lb/ft^2
λ	Taper ratio	$[0.5, 1]$	-
t_c	Airfoil thickness to chord ratio	$[0.08, 0.18]$	-
N_z	Ultimate load factor	$[2.5, 6]$	-
W_{dg}	Design gross weight	$[1700, 2500]$	lb
W_p	Paint weight per unit area	$[0.025, 0.08]$	lb/ft^2

of $\sigma_{me} = 1.7$lb. Since, in this case, z is a synthetic surrogate for a physical observation, there is no direct meaning to the term model discrepancy and it is identically 0. Despite this, if the observation were coming from a real physical measurement the discrepancy term would have had some nonzero value. For this example, the model discrepancy was set to $\sigma_{md} = 1$, a value sufficient to make sure σ_{md} is included in the procedure, yet small enough so as to not overpower the uncertainty coming from the simulated measurement. The treatment of model discrepancy is an important problem in uncertainty quantification and an area of research in itself, see for example [13]. Finally, the simulator described by Eq. (20) is deterministic and has no ensemble error.

Following the ideas outlined in Section 2.2 a Gaussian process emulator was trained with 100 points from an LHS design. The global trend term in Eq. (1) was chosen as $h(\mathbf{x}) = 1$ in this case, so that the Gaussian process component of the emulator was responsible for taking into account any deviations from the mean. This choice is subjective and was motivated by the lack of knowledge of the general shape of the function. Specifying more complex forms for $h(\mathbf{x})$ is possible and can be informed by exploratory analysis. The samples in the training set were normalized in $[0, 1]$ due to the large variation of the input scales. This preprocessing step facilitates the search for optimal correlation lengths, $\boldsymbol{\psi}$, and makes the results more easily interpretable. A genetic algorithm was used to search the likelihood of the emulator for $\boldsymbol{\psi}$, while $\boldsymbol{\beta}$ and σ^2 were computed from the expected values of their respective distributions, given in Eq. (4) and Eq. (5).

At each wave, SuS was used to sample the non-implausible domain with 6000 points per subset level, and each level was given a target probability, $p_0 = 0.1$. Out of the final sample, 10 points from the non-implausible domain were added to the design at locations where the predictive variance from the emulator, given by Eq. (8), was the largest. The number of samples is such that there is at least one point representing each input. Additionally, sites with maximum predictive variance were chosen to rapidly reduce the uncertainty about the non-implausible domain. The GPE was retrained after each wave.

After 9 HM waves and 80 additional evaluations of the model in Eq. (20), the standard deviation of the code uncertainty, σ_c had decreased below that of the measurement error, σ_{me} and the analysis was terminated. Each wave required between 5 and 6 SuS levels, implying that the probability of the non-implausible domain is on the order of 10^{-6} to 10^{-5}. Figure 2 depicts the *optical depth* projections of the input space, introduced by [8]. These projections show the logarithm, base 10, of the empirical probability of finding a non-implausible sample in a given region of the input space, when projected onto a two-dimensional subspace. In this manner, optical depth projections provide a way to visualize the non-implasuible domain conditioned on the pair of inputs in each subfigure. To generate these plots, the input subspace of each pair of inputs was discretised in a 20×20 grid of point values. The remaining 8 dimensions, which vary between subfigures, were represented by a $50,000$ point LHS sample. In this manner, to produce a single optical depth plot, the emulator for the appropriate wave of HM was evaluated $20 \times 20 \times 50,000 = 20,000,000$ times.

The panels in the lower and upper triangles of Figure 2 show the projection plots from the first and last wave of HM, respectively. Several observations can be made from these plots. Firstly, many of the two-dimensional projections of the input space exhibit subtle, but quantifiable reduction in area from the first to the last wave of the analysis. This behaviour can be attributed to the function being relatively smooth and the fact that the GPE mean was capable of representing it with reasonable accuracy early on in the procedure. This is to say that even though the mean of the emulator was able to match the non-implausible domain reasonably well, its distance from the training sample caused the predictive variance of the GPE to be larger than the other sources of uncertainty, preventing the analysis from terminating. For some projections, such as $\Lambda - q$ and $\lambda - q$ the whole space seems to have been discarded as implausible. This outcome is due to the sample-based nature of the optical depth plots and the difficulty of producing data in the non-implausible domain, by uniformly sampling the hidden dimensions. Secondly, the scale of the log-probabilities is indicative of the overall size of the non-implausible domain, with between 1 and 230 samples per $50,000$ producing acceptable matches. Simple-looking problems such as these, show the inadequacy of rejection-based uniform sampling and emphasise the importance of effective methods to identify the non-implausible domain in each wave of HM. Finally, the plots reveal how *active* certain inputs are, which could lead to better understanding of the underlying model. For example, the quarter-chord sweep angle Λ is not identified as important for satisfying the measurement z, as seen from the fact that non-implausible samples are uniformly distributed in its range. Similar conclusions can be made for W_{fw}, q and W_p. On the other hand, the wing area, S_w, the aspect ratio, A_w and the load factor, N_z in particular are all influential in producing an acceptable match to a relatively light wing, confirming the engineering intuition that smaller, less-loaded wings can be made lighter.

The effect of HM on the output of the model is shown in Figure 3. The samples identified to belong to the final non-implausible domain results in the output values shown in Figure 3(a). In this figure, there is a tendency for the outputs to cluster to the upper boundary of the prescribed region. This behaviour serves as an evidence to the restrictive target weight used in the analysis. For comparison, one of the most recognizable general aviation aircraft, Cessna 172, has a wing weight of approximately 236 lb [40]. Figure 3(b) depicts a kernel estimation of the final distribution of the wing weights, which is considerably narrower than the one used to train the initial GPE.

The correlation between samples from SuS can be calculated using the procedure outlined in Section 6 of [12] to determine the quality of the information they provide. In the above example, the coefficient of variation, accounting for sample correlation varies in $\delta = [0.039, 0.068]$.

To illustrate how SuS is capable of sampling more efficiently from the non-implausible domain, HM for the wing weight model was repeated using MC sampling instead of SuS. All other aspects of the analysis were kept the same, except for the number of MC samples. Since MC extracts all of its information in one step, as opposed to SuS, which uses levels, the number of samples required by MC to explore $\mathcal{X}$ is much larger. At each wave, $n_{MC} = 324,000$ samples were generated in $\mathcal{X}$, out of which $m = 10$ samples

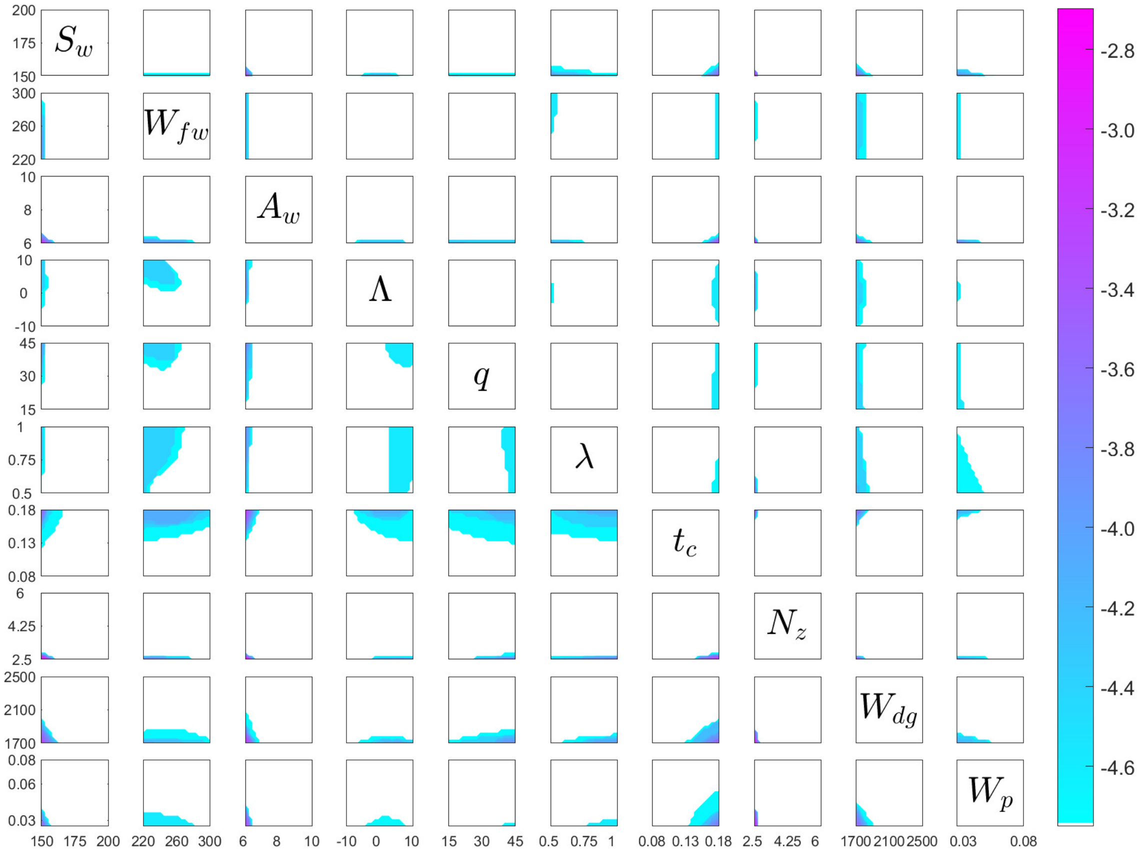
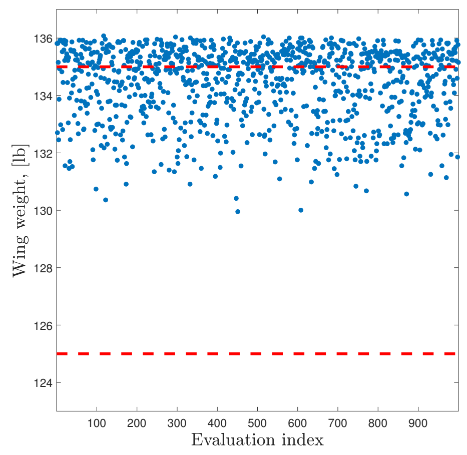
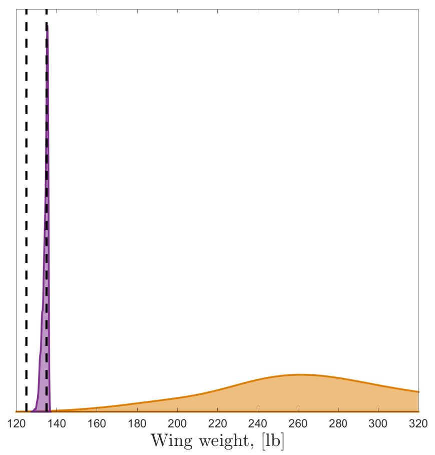


Figure 2: Pairwise optical depth plots for the first (lower triangle) and final (upper triangle) waves of history matching for wing weight. The plots show the evolution of the size of the non-implausible domain and reflect the decreasing log-probability of finding acceptable input combinations (color bar) in different regions of the input space. (color online).



(a)



(b)

Figure 3: Wing weight realizations from the final wave of history matching; (a) emulator predictions (blue dots) compared to the specified target range; (b) kernel density estimation of the initial code output distribution (orange fill) and that from samples in the final non-implausible domain (purple fill). Dashed lines in both figures show the target range (color online).

were to be added to the training set for the next wave emulator. In this comparison, n_{MC} is calculated as the total number of samples used in HM with SuS (9 waves, with 6 SuS levels each and 6000 samples per level). Due to the small volume of the non-implausible domain at each wave, MC was unable to populate it densly enough and as a result $m < 10$ samples were added in each wave. This outcome reveals one of the important advantages of using SuS for sampling the non-implausible domain: unless the set of acceptable matches is empty, SuS is able to populate it according to requirement. The MC-based HM terminated after $\ell = 4$ waves, due to the inability of MC to find samples in the non-implausible domain. A total of 9 samples were added across the 4 waves, which gives a GPE equivalent to the one in Wave 2 in SuS-based HM. It must be pointed out that the efficiency of SuS compared to MC comes at the cost of producing samples that cannot be guaranteed to be uniformly distributed over the non-implausible domain. However, the coefficient of variation accounting for sample correlation δ can be computed, as it was done above. This allows the analyst to monitor efficiency. An interesting question arises when this coefficient of variation is unacceptably large, due to high sample correlation. A potential solution could involve designing a thinning strategy for the modified Metropolis algorithm, but this is the subject of future work.

Figure 4 shows the *minimum implausibility plots* for three pairs of inputs from MC-based HM in the top row, compared to the ones from SuS-based HM in the bottom row. These plots depict the minimum implausibility in the high-dimensional domain, if a given pair of inputs were fixed to a particular value [8]. These plots reveal several things. Firstly, for all pairs of inputs, the non-implausible domain differs in topology. A particularly noticeable difference exists in the space spanned by thickness-to-chord ratio t_c and wing fuel weight W_{fw}. History matching with MC sampling identifies the non-implausible domain to be much more diffuse than the one in the SuS-based analysis. That is, it is larger and at the same time has higher implausibility overall. Secondly, the GPE trained on data with a dense set of non-implausible samples from SuS achieves better accuracy compared to the true function. In the case of the wing weight simulator, the code can be run affordably without the need of an emulator, to explore the implausibility landscape without code uncertainty. The implausibility threshold, $I(x) = 3$, is shown on each of the contour plots as a black dashed line. In all three cases, the agreement is better for the lower line of plots. Finally, it must be noted that the efficiency of the HM process increases when using SuS as a sampler, since the quality of the GPE in the non-implausible domain increases more rapidly when using informative samples. This decreases both the number of potentially costly code evaluations and the number of waves, and thereby emulators, to be generated.

Note that, even if the model in this case study is not computationally expensive, it demonstrates the challenges in calibrating models with even moderately-sized input domains.

6. Robust design: aircraft wing-engine matching

The second application of SuS within HM presented here is *robust design*. In engineering, the term robust design refers to the process of seeking not only an optimal mean value of a system performance metric, but also to ensure that this optimum is insensitive to variations which could lead to undesired system behaviour [41, 42]. The essence of the robust design problem is prescribing a target value for quantities of interest that determine the performance of a system. The designer's task is to then find one or more design input configurations that deliver this target within certain tolerance.

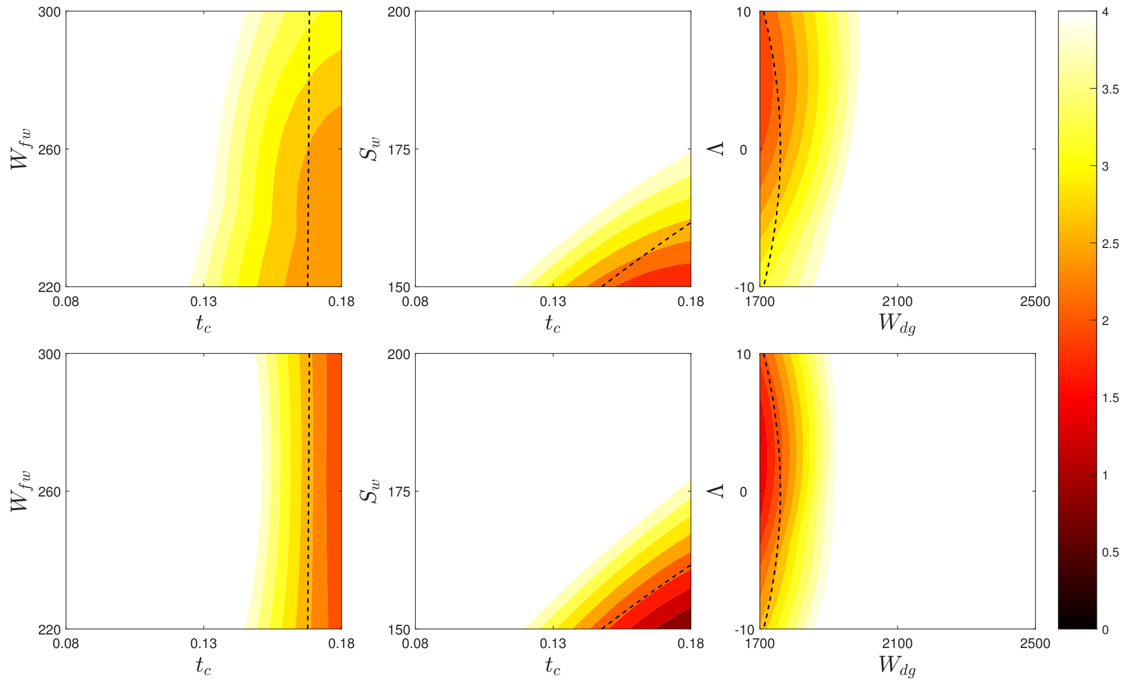


Figure 4: Minimum implausibility plots for three pairs of inputs to the wing weight model. Top row: wave 4 HM results with Monte Carlo sampling. Bottom row: wave 2 HM results with SuS sampling. Dotted line: decision boundary of the non-implausible domain without code uncertainty (color online).

Suppose that a target value for a quantity of interest is treated as if it were an experimental measurement. Also, suppose that the corresponding tolerance can be treated as the underlying uncertainties. This treatment provides an analogy between matching a model output with experimental data (given the sources of uncertainty) and matching a design target within a prescribed tolerance. Therefore, the proposed SuS sampling for HM can also be used to solve the robust design problem by identifying the set of input values that yield an output consistent with a design target within certain tolerance. This results in a reduced input space that can be further explored by an analyst in the search for an optimal design. Since HM can deliver an empty non-implausible domain, the designer might conclude that there is no input configuration that complies with the system requirements, given the current model. This information can be very valuable in terms of improving the model or rethinking the feasibility of the design targets.

This section develops the idea with an application to aircraft design. Subset simulation has previously been used in different optimisation-related problems [43, 44]. However, to the authors' knowledge, it has not been used in robust design. The application proposed in this section demonstrates how SuS-based HM can be used in contexts beyond model calibration.

6.1. Problem description

Modern aircraft are expected to operate within very stringent performance and regulatory limits to reduce their environmental impact, whilst keeping their profitability as a mode of transportation. Increasingly demanding regulations are coming into effect worldwide, which impose bounds on the amount of nitrous oxide (NO_X), among other greenhouse gases produced by aircraft [45]. Such requirements necessitate a highly structured approach to early stage aircraft design, acknowledging the complex nature of interactions and dependencies between different systems. For the purposes of this study, and following the work in [46], the conceptual aircraft is defined as a combination of different wings and engines, in an approach known as *set-based design*. Each wing and engine are in turn defined by the parameters given in Table 2.

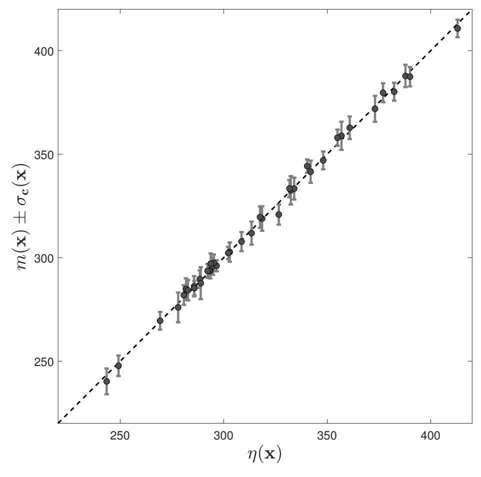
Whilst the modelling process is multi-disciplinary and multi-organisational, here it is presented in an abstract form as a chain of coupled analyses implemented in a tool called AirCADia [47]. AirCADia is a framework for interactive composition and exploration of conceptual aircraft design configurations. In this case study, six parameters were varied within AirCADia to achieve the target emissions value. In order to collect all required data, the model was run on a Lenovo ThinkCentre M900 Tower, with an Intel® Core™ i7-6700, 3.4 GHz CPU. On this machine, each evaluation took 0.5 seconds.

Table 2: Inputs and output for the climb-cruise case study, with respective parent system and target ranges.

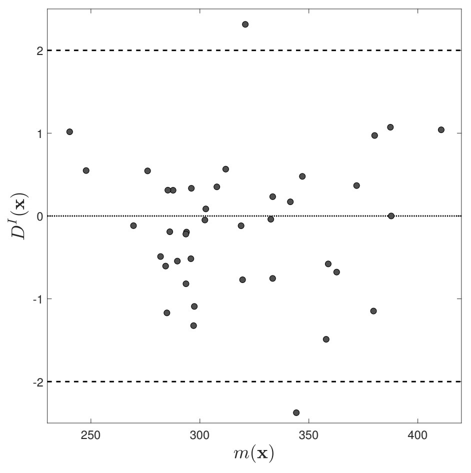
Notation	**Name**	**System**	**Range**	**Unit**
S_W	Wing area	Airframe	$[1300, 1400]$	ft^2
A_W	Aspect ratio	Airframe	$[9, 11]$	-
SLST	Static thrust	Engine	$[26, 32]$	lbf $\times 10^3$
FPR	Fan pressure ratio	Engine	$[1.5, 1.8]$	-
OPR	Overall pressure ratio	Engine	$[30, 40]$	-
BPR	Bypass ratio	Engine	$[6, 8]$	-
NO_X	Nitrous oxide emissions	Output	240 ± 10	lb

6.2. *History matching NO_X*

The level of NO_X emissions was selected as the target output variable that would drive the design. Initially, a GPE was trained on $n = 60$ Latin hypercube points, using a global trend term, $h(\mathbf{x}) = [1,\ \mathbf{x}]^\intercal$. The emulator was validated with another $m = 40$ LHS samples to verify its accuracy in representing AirCADia's output. The plot of simulator outputs against GPE predictions is displayed in Figure 5(a). It shows the degree to which predictions from the emulator correspond with simulator observations. If the GPE were a perfect predictor, the scatter would have lain along the 45 degree dashed line. The error bars indicate the 95% credible interval associated with each point. Most of the predicted points contain the 45 degree line in their credible intervals. As seen in Eq. (6), each prediction from the emulator follows a Student's-t distribution. Therefore, the residuals between simulator output and prediction should occupy the interval $[-2, 2]$ with around 95% probability. These normalised residuals, often termed *individual prediction errors* [27], are plotted against predictions in Figure 5(b). The residuals are uniformly spread around 0 with no discernible patterns, or significant number of outliers. Jointly, these visual diagnostics suggest that the emulator is a reasonably accurate representation of the simulator. After validation, the test points were added to the design of experiments and the trend coefficients in Eq. (4), and process variance in Eq. (5) were re-estimated.



(a)



(b)

Figure 5: Predictive diagnostics for the NO_X GPE. (a) correlation between prediction and observations with 95% credible intervals depicted as error bars; (b) individual prediction errors for the validation set.

After consultation with the developers of AirCADia, at the Centre for Aeronautics at Cranfield University, the target range for NO_X was chosen as 240 ± 10 lb over a 3000 nautical mile trip, including landing and take-off [47]. The reader is reminded that the end goal of the robust design task is to attain a pre-specified level, with tolerance, of a quantity of interest. This is in contrast to the aim of optimisation, where, typically, the analyst seeks to attain an optimum level of the quantity of interest, possibly subject to constraints. The experts' reasoning behind choosing the specific NO_X target is not provided herein, since it is not in line with the main aim of the paper. As outlined before, in the robust design setting the target range can be treated as measurement plus corresponding uncertainties. Therefore, all uncertainties for HM are accumulated into the measurement error term. In order to ensure that the target range is respected, HM was carried out with an error term which ensures that 95% of the responses will lie in the correct region. Thus, the final values for the analysis were set as $z = 240$ and $\sigma_{me} = 3.33$.

In each wave of HM, SuS was run with $N = 6,000$ samples per level and level probability, $p_0 = 0.1$. In the first wave, two levels of SuS were required to populate the non-implausible domain, implying that its probability is on the order of 10^{-2}. The two levels sampled the emulator a total of $12,000$ times, obtaining over $3,500$ samples in the non-implausible domain. For comparison, a direct Monte Carlo simulation would have required approximately $350,000$ samples on average to achieve a similar result. The code uncertainty associated with some of the samples from SuS exceeded the measurement error and therefore it was necessary to continue with the analysis.

The analysis was terminated after three waves, when the emulator variance $\sigma_c^2(\mathbf{x})$ was reduced sufficiently in comparison with the imposed uncertainty[4]. From the denominator in Eq. (9), it can be seen that in this example, $\sigma_c^2(\mathbf{x})$ is the only source of uncertainty that is free to change. Once it becomes small in comparison with the other components, the implausibility measure does not change significantly. Further substantial reduction in the non-implausible domain becomes unlikely.

Figure 6 shows diagnostics from the final wave of HM. The sub-figures in the upper triangle contain the optical depth plots and those in the lower triangle show the minimum implausibility plots. Together, these two representations visualize the extent of the non-implausible domain. In Figure 6, it can be seen that the inputs relating to engine pressure (OPR, FPR) have significant contribution to the value of $\mathrm{NO_X}$, since their domain was significantly reduced to achieve the specified target range. In particular, it was not likely to find matching outputs for high values of OPR and low values of FPR, regardless of the values of the other inputs. An interesting interaction is the one between sea-level static thrust (SLST) and wing aspect ratio (A_W), which indicates that low powered engines must be matched to efficient, slender wings to attain the required $\mathrm{NO_X}$ level.

The values of $\mathrm{NO_X}$ corresponding to the non-implausible samples are shown in Figure 7. Note that the values of the emissions in Figure 7(a) exceed the specified range. This is due to the code uncertainty introduced using the emulator instead of the original code. This uncertainty can be reduced further, but an increase in the computational cost of the analysis will be incurred, owing to the additional code evaluations needed to refine the surrogate model. Figure 7(b), provides a visual comparison between the pre- and post-history matching distributions of the output.

[4]Despite the seemingly quick running times of the simulator, the analyses would have taken approximately 5 hours for SuS and just over 2.5 days for DMC, if the simulator was sampled directly.

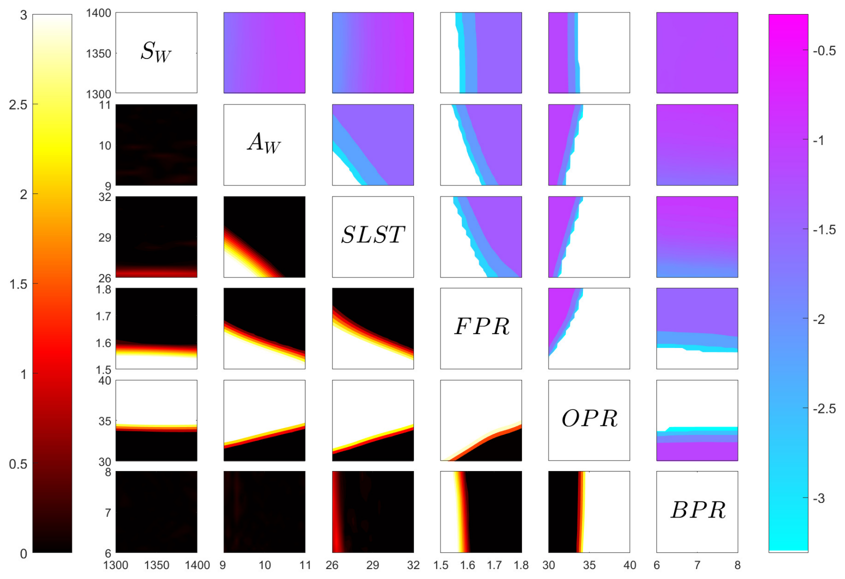
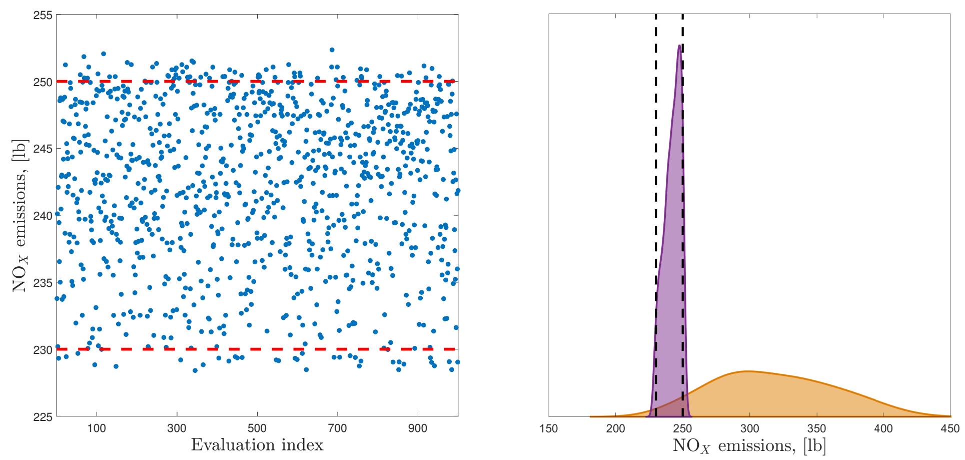


Figure 6: Pairwise minimum implausibility (lower triangle, left color bar) and optical depth (upper triangle, right color bar) plots from the last wave of NO_X history matching. The color bar on the right depicts the empirical log-probability of finding a non-implausible sample in a given region of the input domain, whereas the one on the left indicates the expected implausibility value of that sample. Inputs belonging to the “Engine” subsystem are clearly affected more than those belonging to the “Airframe” subsystem in Table 2 (color online).



(a) (b)

Figure 7: History matching identifies input configurations, which result in output values lying in the specified target range (dashed lines); (a) emulator predictions (blue dots) and the observation error distribution; (b) kernel density estimation of code outputs before (orange fill) and after (purple fill) history matching (color online).

7. Conclusions

A solution to an important problem in model calibration with history matching was proposed. The solution involves the use of subset simulation to generate samples from the non-implausible domain of an expensive computer model. It was shown that, within history matching, the volume of the non-implausible domain may shrink by several orders of magnitude as compared to the original input space. Thus, the non-implausible domain was treated as a failure set, analogous to that in engineering reliability analysis. This allowed the use of subset simulation as an efficient sampler, which provided good coverage of the non-implausible domain with a moderate number of samples. The method selected highly informative input configurations, which were used to train a Bayesian emulator. This led to a reduction in computational time and fast convergence of the analysis.

The advantages of the proposed approach were demonstrated in two examples. The first one dealt with the calibration of an analytical wing model to match a restrictively low target weight. The second example showed how the proposed approach can be used as a pre-processor for robust design in an industrial context. Future research based on this work includes exploring the link between history matching and robust design with several, possibly conflicting, design objectives. Another problem that requires attention is that of local variations in the behaviour of the simulator. This might require fitting different emulators if the non-implausible domain is disconnected.

Acknowledgements

The authors are grateful for the kind support from Dr. Xin Chen, Dr. Arturo Molina-Cristobal, Dr. Atif Riaz and Prof. Marin Guenov, from the Centre for Aeronautics at Cranfield University. Their team develops and maintains AirCADia, and provided the authors with access to the model.

F. A. DiazDelaO acknowledges the support of the Data-centric engineering programme at The Alan Turing Institute, where he was a visiting fellow under the EPSRC grant EP/S001476/1.

Author Contributions

The method was conceived by F. A. DiazDelaO, Z. T. Gong and M. Beer. Initial coding and experiments were carried out by Z. T. Gong. An earlier version of the manuscript was written for a conference by Z. T. Gong, F. A. DiazDelaO and M. Beer [48]. More detailed coding and experiments were carried out by P. O. Hristov. An extensive revision of the manuscript was carried out by P. O. Hristov and F. A. DiazDelaO.

References

References

[1] Andrianakis, I., Vernon, I.R., McCreesh, N., McKinley, T.J., Oakley, J.E., Nsubuga, R., Goldstein, M., and White, R.G., Bayesian history matching of complex infectious disease models using emulation: A tutorial and a case study on HIV in Uganda., *PLoS Computational Biology*, 11(1):1–29, 2015.

[2] Steyerberg, E.W., Vickers, A.J., Cook, N.R., Gerds, T., Gonen, M., Obuchowski, N., Pencina, M.J., and Kattan, M.W., Assessing the Performance of Prediction Models, *Epidemiology*, 21(1):128–138, 2010.

[3] Cheng, Q.B., Chen, X., Xu, C.Y., Reinhardt-Imjela, C., and Schulte, A., Improvement and comparison of likelihood functions for model calibration and parameter uncertainty analysis within a Markov chain Monte Carlo scheme, *Journal of Hydrology*, 519:2202 – 2214, 2014.

[4] Gibson, G.J. and Renshaw, E., Estimating parameters in stochastic compartmental models using markov chain methods, *Mathematical Medicine and Biology: A Journal of the IMA*, 15(1):19–40, 1998.

[5] Alfi, M. and Hosseini, S.A., Integration of reservoir simulation, history matching, and 4D seismic for CO2-EOR and storage at Cranfield, Mississippi, USA., *Fuel*, 175:116 – 128, 2016.

[6] Anterion, F., History matching: A one day long competition: classical approaches versus gradient method., In *First International Forum On Reservoir Simulation*, 1998.

[7] O'Hagan, A., Bayesian analysis of computer code outputs: A tutorial., *Reliability Engineering & System Safety*, 91(10):1290 – 1300, 2006.

[8] Vernon, I., Goldstein, M., and Bower, R., Galaxy formation: A Bayesian uncertainty analysis., *Bayesian Analysis*, 5(4):619–669, 2010.

[9] Craig, P.S., Goldstein, M., Rougier, J.C., and Seheult, A.H., Bayesian forecasting for complex systems using computer simulators, *Journal of the American Statistical Association*, 96:717–730, 2001.

[10] Williamson, D. and Blaker, A.T., Evolving Bayesian emulators for structured chaotic time series, with application to large climate models., *Society for Industrial and Applied Mathematics*, 2014.

[11] Vernon, I., Goldstein, M., and Bower, R., Galaxy formation: Bayesian history matching for the observable universe, *Statistical Science*, 29(1):81–90, 2014.

[12] Au, S.K. and Beck, J., Estimation of small failure probabilities in high dimensions by subset simulation., *Probabilistic Engineering Mechanics*, 16(4):263–277, 2001.

[13] Brynjarsdóttir, J. and O'Hagan, A., Learning about physical parameters: The importance of model discrepancy, *Inverse Problems*, 30(11):114007, 2014.

[14] Kennedy, M.C. and O'Hagan, A., Bayesian calibration of computer models, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3):425–464, 2001.

[15] McKay, M.D., Beckman, R.J., and Conover, W.J., A comparison of three methods for selecting values of input variables in the analysis of output from a computer code., *Technometrics*, 21:239–245, 1979.

[16] Loeppky, J.L., Sacks, J., and Welch, W.J., Choosing the sample size of a computer experiment: A practical guide., *Technometrics*, 51:366–378, 2009.

[17] Oakley, J.E. and O'Hagan, A., Probabilistic sensitivity analysis of complex models: A Bayesian approach., *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 66:751–770, 2004.

[18] Salmanidou, D.M., Guillas, S., Georgiopoulou, A., and Dias, F., Statistical emulation of landslide-induced tsunamis at the Rockall Bank, N-E atlantic, *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473(2200):20170026, 2017.

[19] Hristov, P.O., DiazDelaO, F.A., Farooq, U., and Kubiak, K.J., Adaptive Gaussian process emulators for efficient reliability analysis, *Applied Mathematical Modelling*, 71:138 – 151, 2019.

[20] Craig, P., Goldstein, M., Seheult, H., and Smith, J.A., Bayes linear strategies for matching for matching hydrocarbon reservoir history, *Bayesian Statistics*, 5:69 – 95, 1996.

[21] Craig, P., Goldstein, M., Seheult, H., and Smith, J.A., Pressure matching for hydrocarbon reservoirs: A case study in the use of Bayes linear strategies for large computer experiments, In *Case Studies in Bayesian Statistics*, pp. 37–93. Springer New York, 1997.

[22] Goldstein, M. and Wooff, D., *Bayes Linear Statistics, Theory and Methods*, Wiley Series in Probability and Statistics, Wiley, 2007.

[23] Rasmussen, C.E. and Williams, C.K.I., *Gaussian processes for machine learning.*, MIT Press, 2006.

[24] Minasny, B. and McBratney, A.B., The Matérn function as a general model for soil variograms., *Geoderma*, 128(3-4):192 – 207, 2005.

[25] Duvenaud, D., Automatic model construction with Gaussian processes, PhD thesis, University of Cambridge, 2014.

[26] Oakley, J., Bayesian uncertainty analysis for complex computer codes., PhD thesis, University of Sheffield, 1999.

[27] Bastos, L.S. and O'Hagan, A., Diagnostics for Gaussian process emulators., *Technometrics*, 51(4):425–438, 2009.

[28] Garbuno-Inigo, A., DiazDelaO, F.A., and Zuev, K.M., Transitional annealed adaptive slice sampling for Gaussian process hyper-parameter estimation, *International Journal for Uncertainty Quantification*, 6(4):341–359, 2016.

[29] Garbuno-Inigo, A., DiazDelaO, F.A., and Zuev, K.M., Gaussian process hyper-parameter estimation using parallel asymptotically independent Markov sampling, *Computational Statistics & Data Analysis*, 103:367 – 383, 2016.

[30] Forrester, A.I.J., Sobester, A., and Keane, A.J., *Engineering Design Via Surrogate Modelling: A Practical Guide*, J. Wiley, 2008.

[31] Pukelsheim, F., The three sigma rule., *The American Statistician*, 88:88–91, 1994.

[32] Williamson, D. and Vernon, I. Efficient uniform designs for multi-wave computer experiments. arXiv:1309.3520, 2013.

[33] Au, S.K. and Patelli, E., Rare event simulation in finite-infinite dimensional space., *Reliability Engineering and System Safety*, 148:67 – 77, 2016.

[34] Zuev, K.M., Beck, J.L., Au, S.K., and Katafygiotis, L., Bayesian post-processor and other enhancements of subset simulation for estimating failure probabilities in high dimensions, *Computers & Structures*, 92–93:283–296, 2012.

[35] Papaioannou, I., Betz, W., Zwirglmaier, K., and Straub, D., MCMC algorithms for subset simulation, *Probabilistic Engineering Mechanics*, 41:89 – 103, 2015.

[36] Chiachio, M., Beck, J.L., Chichio, J., and Rus, G., Approximate Bayesian computation by subset simulation, *SIAM Journal of Scientific Computing*, 36(3):A1339 – A1358, 2014.

[37] DiazDelaO, F., Garbuno-Inigo, A., Au, S., and Yoshida, I., Bayesian updating and model class selection with subset simulation, *CMAME*, 317:1102–1121, 2017.

[38] Gong, Z.T., DiazDelaO, F.A., and Beer, M., Sampling schemes for history matching using subset simulation, In *Proceedings of the* 2^{nd} *International Conference on Uncertainty Quantification in Computational Sciences and Engineering*, 2017.

[39] Raymer, D., *Aircraft Design: A Conceptual Approach*, AIAA Education Series, American Institute of Aeronautics & Astronautics, 1999.

[40] Taylor, J.W. (Ed.), *Jane's All the World Aircraft 1977-78*, Jane's Information Group, 68 edition, 1978.

[41] Beyer, H.G. and Sendhoff, B., Robust optimization – A comprehensive survey, *Computer methods in applied mechanics and engineering*, 196(33-34):3190–3218, 2007.

[42] Zang, C., Friswell, M., and Mottershead, J., A review of robust optimal design and its application in dynamics, *Computers & structures*, 83(4-5):315–326, 2005.

[43] Li, H.S. and Au, S.K., Design optimization using subset simulation algorithm., *Structural Safety*, 32(6):384 – 392, 2010.

[44] Dubourg, V., Sudret, B., and Bourinet, J.M., Reliability-based design optimization using Kriging surrogates and subset simulation., *Structural and Multidisciplinary Optimization*, 44(5):673–690, 2011.

[45] EASA, European Aviation Environmental Report, Tech. rep., European Aviation Safety Agency, 2016.

[46] Riaz, A., Guenov, M.D., and Molina-Cristobal, A., Set-Based Approach to Passenger Aircraft Family Design, *Journal of Aircraft*, 54(1):310–326, 2017.

[47] Guenov, M., Nunez, M., Molina-Cristóbal, A., Datta, V.C., and Riaz, A., Aircadia – An Interactive Tool for the Composition and Exploration of Aircraft Computational Studies At Early Design Stage, *29th Congress of the International Council of the Aeronautical Sciences*, pp. 1–12, 2014.

[48] Gong, Z.T., DiazDelaO, F.A., and Beer, M., Bayesian model calibration using subset simulation, In *Risk, Reliability and Safety: Innovating Theory and Practice: Proceedings of ESREL 2016, Glasgow, Scotland.*, 2016.

Appendix A. SuS illustration function

The function in Eq. (18), whose contour levels are shown in Figure 1 is a mixture of nine bivariate Gaussian random variables with mean, covariance and weight given in Table A.1.

Table A.1: Parameter values for $g(\mathbf{x})$ in Eq. (18).

i	w_i	$\mu_i^\intercal$	$\mathbf{C}_i$
1	0.327	$\begin{bmatrix} 0.04 & 0.04 \end{bmatrix}$	$\begin{bmatrix} 0.030 & 0.020 \\ 0.020 & 0.025 \end{bmatrix}$
2	0.096	$\begin{bmatrix} 0.98 & 0.70 \end{bmatrix}$	$\begin{bmatrix} 0.020 & 0 \\ 0 & 0.003 \end{bmatrix}$
3	0.143	$\begin{bmatrix} 0.75 & 0.85 \end{bmatrix}$	$\begin{bmatrix} 0.010 & -0.015 \\ -0.015 & 0.030 \end{bmatrix}$
4	0.038	$\begin{bmatrix} 0.71 & 0.32 \end{bmatrix}$	$\begin{bmatrix} 0.002 & 0 \\ 0 & 0.002 \end{bmatrix}$
5	0.161	$\begin{bmatrix} 0.33 & 0.83 \end{bmatrix}$	$\begin{bmatrix} 0.020 & -0.010 \\ -0.010 & 0.010 \end{bmatrix}$
6	0.023	$\begin{bmatrix} 0.43 & 0.73 \end{bmatrix}$	$\begin{bmatrix} 0.005 & 0 \\ 0 & 0.005 \end{bmatrix}$
7	0.026	$\begin{bmatrix} 0.23 & 0.93 \end{bmatrix}$	$\begin{bmatrix} 0.005 & 0 \\ 0 & 0.005 \end{bmatrix}$
8	0.104	$\begin{bmatrix} 1.00 & 0.00 \end{bmatrix}$	$\begin{bmatrix} 0.008 & 0 \\ 0 & 0.008 \end{bmatrix}$
9	0.081	$\begin{bmatrix} 0.12 & 0.57 \end{bmatrix}$	$\begin{bmatrix} 0.005 & 0 \\ 0 & 0.005 \end{bmatrix}$