49th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials
7 - 10 April 2008, Schaumburg, IL

AIAA 2008-2267

49th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Materials Conference, 7-10 April 2008, Schaumburg, IL

Bayesian Emulator Approach for Complex Dynamical Systems

F. A. Díaz De la O[*] and S. Adhikari[†]

Swansea University, Swansea, United Kingdom

In the dynamical analysis of complex systems, running a detailed high-resolution finite element model can be expensive, even for obtaining the dynamic response at few frequency points. In this paper, a new computational tool based on Bayesian emulators is developed. Using this approach, the dynamic response over a frequency range can be approximated using only a few response values, obtained either from experimental measurements or by running a finite element model at carefully selected input frequency points. The proposed Bayesian emulator method is applied to two numerical example problems and one problem with experimentally measured data. It is shown that the method can be an effective predictive tool for mid and high frequency vibration problems.

Nomenclature

$\mathbf{x}$	Input of the simulator
$\mathbf{y}$	Output of the simulator
n	Size of the training set
N	Number of degrees of freedom
$\eta(\cdot)$	Simulator
Δ_η	Domain of the simulator
$Z(\cdot)$	Gaussian stochastic process
$\mathbf{M}$	Mass matrix
$\mathbf{C}$	Damping matrix
$\mathbf{K}$	Stiffness matrix
ω	Excitation frequency
$\mathbf{D}(\omega)$	Dynamic stiffness matrix
$\mathbf{f}(t)$	Forcing vector
$\mathbf{q}(t)$	Displacement vector
$\bar{\mathbf{f}}(\omega)$	Fourier transform of $\mathbf{f}(t)$
$\bar{\mathbf{q}}(\omega)$	Fourier transform of $\mathbf{q}(t)$
$\mathbf{I}$	Identity matrix
$\mathbf{B}$	Smoothness parameters matrix
D	Initial design
$\mathbb{R}^d$	Euclidean space of dimension d
$\mathbb{R}^{m\times N}$	Space of matrices of size $m \times N$
i	Unit complex number

[*] Graduate Student, School of Engineering, Swansea University, Singleton Park, Swansea SA2 8PP, UK.

[†] Professor of Aerospace Engineering, School of Engineering, Swansea University, Singleton Park, Swansea SA2 8PP, UK, AIAA Senior Member.

$\mathbf{E}[\cdot]$	Expectation operator
$\mathbf{E}[\cdot\vert\cdot]$	Conditional expectation operator
$\vert\cdot\vert$	Modulus operator
$\sim$	Distributed as

Subscript

j, k, ℓ	Variable number

I. Introduction

MANY engineering dynamical systems are complex enough to render physical experimentation impossible. As a consequence, these systems are often investigated running computer codes. O'Hagan[1] refers to such codes, as well as to the underlying mathematical models, as *simulators*. In mathematical terms, a simulator is a function $\eta(\cdot)$ that given an input $\mathbf{x}$, it produces an output $\mathbf{y} = \eta(\mathbf{x})$. Simulators are a common tool when studying intricate phenomena, however, they can have a high cost of execution. This cost may be measured in terms of employed CPU time, number of floating point operations performed or required computer capability. Consider the example given by Goldstein,[2] in which the most sophisticated climate models can take months to complete a single run. Similarly, Thomke et al.[3] note the prodigious computer power needed to simulate the outcome of a rollover car accident. The terms "expensive" and "computer intensive" are equivalent and will be used indistinctly throughout the present paper. The same applies to "simulator" and "model".

Several approaches, such as local polynomial regression and neural networks have been devised to reduce the computational cost of expensive simulators. One of these approaches, which has been in constant development over the last two decades, is Bayesian emulation. Based on the Analysis and Design of Computer Experiments (see Satner et al.[4]), and using concepts of Bayesian Statistics, such technology consists in constructing an approximation to the simulator, called an *emulator*. More precisely, an emulator is a statistical approximation to the simulator. Not only does it approximate $\eta(\cdot)$, it provides a probability distribution for it. Broadly speaking, emulation works in the following way: A set of runs of the simulator are treated as data that will be used to update the prior beliefs about it. As it will be explained later, these prior beliefs take the form of a Gaussian stochastic process. The number of runs will be small relative to the size of the input domain of the simulator, since by assumption it is computer intensive. After the updating, the emulator will produce an interpolation/extrapolation of the available data at untried inputs, whereas it will return the known value of the simulator at each of the initial runs.

Emulators have already been implemented in a number of different scientific fields. Kennedy et al.[5] presented three case studies related to environmental computer models. First, they emulated a vegetation dynamic model. They also emulated a model of ecosystem photosynthesis and water balance, and finally another one that estimates the UK carbon budget. Challenor et al.[6] emulated what they consider to be a moderately complex climate model. Rougier[7] presented another application to a climate model. Bates et al.[8] emulated a model of a complete revolution of a piston's shaft. More examples of applications can be found in the literature that develops the theoretical framework of emulators. Haylock and O'Hagan[9] emulated a model of doses to organs of the body after ingestion of a radioactive substance. Oakley and O'Hagan[10] worked with a simulator of the cost resulting from bone fractures for patients suffering from osteoporosis. Bearing these applications in mind, and considering that Structural Dynamics is prone to the use of computer intensive models, it could greatly benefit from emulators.

The structure of the paper is the following. Section II will introduce the Structural Dynamics model to be emulated. Section III will provide an overview of the mathematical theory behind emulators, continued in Section IV with some discussion on relevant details for their implementation. In Section V, a demonstration of the capabilities of emulators will be carried out, with an increasing level of complexity. A simulator of the frequency response function (FRF) of a simple damped spring-mass system with three degrees of freedom will be used. A much more complex simulator, involving a system with thousands of degrees of freedom will also be emulated. The main objective of presenting these examples is to illustrate the potential use of emulators as convenient surrogates for expensive engineering computer models. Additionally, an exploration of the applicability of emulators in a circumstance in which there is no mathematical/computational model available for a given phenomenon, due perhaps to the lack of knowledge of the physics of the system, will be conducted. In such case there might only be a few experimental data available to build the emulator. The results will be compared with a set of real data, taken from an experiment performed by Adhikari et al.[11]

II. Linear Structural Dynamics

Consider the problem of modeling the response of a structural system to different frequency ranges of vibration. The equation of motion of a damped N-degree-of-freedom linear structural system is given by

$$\mathbf{M}\ddot{\mathbf{q}}(t) + \mathbf{C}\dot{\mathbf{q}}(t) + \mathbf{K}\mathbf{q}(t) = \mathbf{f}(t) \tag{1}$$

where $\mathbf{f}(t) \in \mathbb{R}^N$ is the forcing vector, $\mathbf{q}(t) \in \mathbb{R}^N$ is the response vector and $\mathbf{M} \in \mathbb{R}^{N\times N}$, $\mathbf{C} \in \mathbb{R}^{N\times N}$ and $\mathbf{K} \in \mathbb{R}^{N\times N}$ are respectively the mass, damping and stiffness matrices.

Equation (1) can be expressed in terms of the excitation frequency level, $\omega \in [0, ..., \infty)$, as

$$\mathbf{D}(\omega)\bar{\mathbf{q}}(\omega) = \bar{\mathbf{f}}(\omega) \tag{2}$$

where $\bar{\mathbf{q}}(\omega)$ and $\bar{\mathbf{f}}(\omega)$ are the Fourier transforms of $\mathbf{q}$ and $\mathbf{f}$ respectively. The matrix $\mathbf{D}(\omega)$ is known as *dynamic stiffness matrix* and is given by

$$\mathbf{D}(\omega) = -\omega^2\mathbf{M} + i\omega\mathbf{C} + \mathbf{K} \tag{3}$$

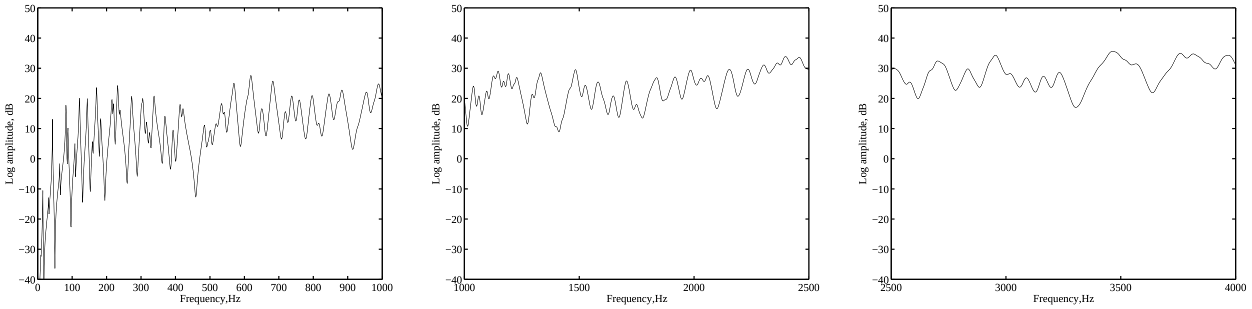
Provided that $\mathbf{D}^{-1}(\omega)$ exists, the response vector becomes $\bar{\mathbf{q}}(\omega) = \mathbf{D}^{-1}(\omega)\bar{\mathbf{f}}(\omega)$. Suppose there is interest in working with some linear function of the elements of $\bar{\mathbf{q}}(\omega)$, namely

$$\phi(\omega) = \mathbf{Q}[-\omega^2\mathbf{M} + i\omega\mathbf{C} + \mathbf{K}]^{-1}\bar{\mathbf{f}}(\omega) \tag{4}$$

where $\mathbf{Q} \in \mathbb{R}^{m\times N}$ is a rectangular matrix. Note that $\phi(\cdot)$ is a complex-valued function. For the purpose of measuring the frequency response, only the value of its modulus is relevant. That way, let

$$\eta(\omega) = \left|\mathbf{Q}[-\omega^2\mathbf{M} + i\omega\mathbf{C} + \mathbf{K}]^{-1}\bar{\mathbf{f}}(\omega)\right| \tag{5}$$

Equation (5) cannot always be solved analytically. Adhikari et. al[11] proposed a simulation technique to produce the FRF of a plate subject to vibration. To simulate the response of a dynamical system with 1200 degrees of freedom, they considered 0 - 1.0 kHz as the low-frequency range, 1.0 - 2.5 kHz as the medium-frequency range, and 2.5 - 4.0 kHz as the high-frequency range. The outcome of such simulator is shown in Figure 1. This example illustrates how the FRF of a structure subject to vibration can be conducted using well established techniques such as the finite element method coupled with dynamic analysis (see for example[12–19]). Nevertheless, the corresponding dynamic stiffness matrix $D(\omega)$ had to be inverted 4000 times. This implies a potential problem of computational efficiency, since the execution of the simulator might become too resource-consuming for systems with several millions of degrees of freedom. In such context, emulation might be a sensible choice. In the following section, the basic theory on the implementation of emulators will be briefly exposed, in order to overcome the problem.



(a) Low-frequency range (b) Medium-frequency range (c) High-frequency range

Figure 1. Simulator of the FRF of a cantilever plate subject to vibration, in different segments of the frequency range.

III. Implementation of the Emulator

Suppose that n points, namely $\mathbf{x}_1, \ldots, \mathbf{x}_n$, have been chosen in the input domain of the simulator $\eta(\cdot)$. Each of these will be referred to as a *design point*. The set $\{\mathbf{y}_1 = \eta(\mathbf{x}_1), \ldots, \mathbf{y}_n = \eta(\mathbf{x}_n)\}$, resulting from the evaluation of $\eta(\cdot)$ in each of the design points, will be called *training set*. Following O'Hagan,[1] an emulator should satisfy some minimal criteria:

1. Since true value of the output for each design point is known, the emulator should produce the corresponding output with no uncertainty.

2. At any $\mathbf{x}$ that is not a design point, the probability distribution provided by the emulator should produce a mean value that constitutes a plausible interpolation/extrapolation of the training data. The probability distribution around this mean value should also express the uncertainty about how the emulator might interpolate/extrapolate.

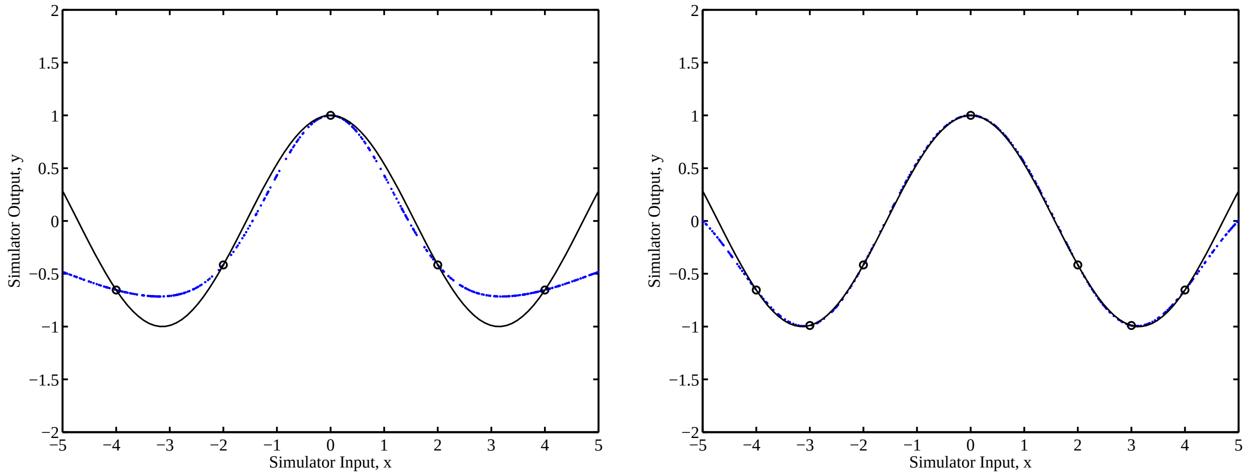
Naturally, it is also desirable that emulation results as efficient as other available techniques, if it is to be worthy.

To have a better understanding of what do the above criteria mean, take a simple one-dimensional simulator and treat it as if it were computer intensive. Figure 2(a) shows the case when five training runs (the circles) are used. The mean of the emulator (the dots) approximates the real values of the simulator (the solid line) at several untried inputs throughout the input domain. On the other hand, it returns the exact value of the simulator at each of the training points. Note how the approximation improves when more training runs are used, as shown in Figure 2(b). Additionally, Figure 3 shows upper and lower bounds of two standard deviations for the mean of the emulator. As the number of training runs increases, there is a reduction of the uncertainty in the value that the mean of the emulator might have. Note how uncertainty is equal to zero in each of the training runs, as it would be expected, since the emulator returns the true value of the simulator in these points. Observe however that for both cases, uncertainty increases very rapidly when extrapolating the training set.

If $\eta(\cdot)$ is an expensive simulator, it can only be evaluated at a limited number of inputs. From the perspective of Bayesian Statistics, $\eta(\cdot)$ is regarded as a random variable in the sense that it is unknown until the simulator is run. Thus, a characterization of the relationship between the input and the unknown output must first be selected . Let us elaborate on how subjective information about this relationship is combined with objective information provided by the training set to predict the output of the simulator at untried inputs.

Begin by assuming that the random function $\eta(\cdot)$ deviates from the mean of its distribution in the following way

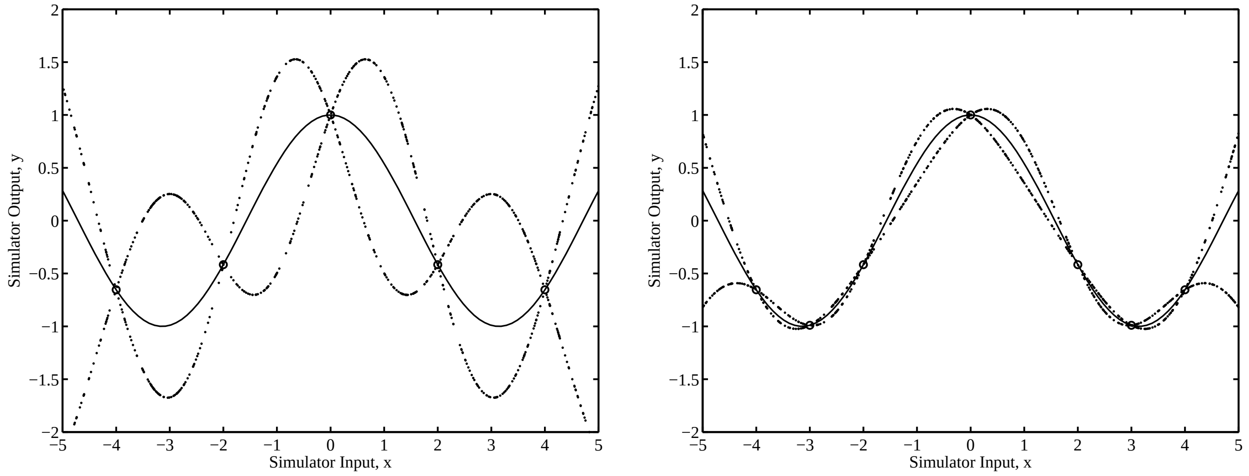
$$\eta(\mathbf{x}) = \sum_{j=1}^{n} \beta_j h_j(\mathbf{x}) + Z(\mathbf{x}) \tag{6}$$



(a) Interpolation/extrapolation with 5 design points.

(b) Interpolation/extrapolation with 7 design points.

Figure 2. Two approximations of the simulator y $= \cos(\mathbf{x})$, when a different number of design points is used. The solid line is the true output of the simulator. The circles represent the training runs, and the dots are the mean of the emulator, which provides the approximation.



(a) Uncertainty with 5 design points.

(b) Uncertainty with 7 design points.

Figure 3. Reduction in the uncertainty about the true value of the simple simulator y $= \cos(\mathbf{x})$ at untried inputs, due to the increase in the number of design points used to build the emulator.

where for all j, $h_j(\mathbf{x})$ is a known function and β_j is an unknown coefficient. The function $Z(\cdot)$ is assumed to be a stochastic process with mean zero and covariance function $Cov(Z(\mathbf{x}), Z(\mathbf{x}^{'}))$ given by some function of **x** and $\mathbf{x}^{'}$. An advantageous choice for $Z(\cdot)$ is the *Gaussian stochastic process*. As noted by Kennedy and O'Hagan,[20] it is used in practice for much the same reasons that the normal distribution repeatedly appears in Statistics: it is convenient, flexible and quite often realistic.

Definition. Gaussian stochastic process: Let $\mathcal{X} \subseteq \mathbb{R}^d$. Then $Z(\mathbf{x})$ for $\mathbf{x} \in \mathcal{X}$ is a Gaussian stochastic process if for any $L \geq 1$ and any choice $\{\mathbf{x}_1, \ldots, \mathbf{x}_L\} \subseteq \mathcal{X}$, the vector $[Z(\mathbf{x}_1), \ldots, Z(\mathbf{x}_L)]^T$ has a multivariate normal distribution.

Given the above definition, $\eta(\cdot) \sim N(m(\cdot), \sigma^2 C(\cdot,\cdot))$ means that $\eta(\cdot)$ has a Gaussian process distribution with mean function $m(\cdot)$ and covariance $\sigma^2 C(\cdot,\cdot)$. If a linear structure to model $m(\cdot)$ is assumed, then

$$m(\cdot) = \mathbf{h}(\cdot)^T \boldsymbol{\beta} \tag{7}$$

where $\mathbf{h}(\cdot)$ is a vector of known functions and $\boldsymbol{\beta}$ is a vector of unknown coefficients. That way, the interpretation of Eq. (6) becomes clearer, that is, the random function $\eta(\cdot)$ deviates from the mean of its distribution following a Gaussian stochastic process.

A very important assumption must now be done: $\eta(\cdot)$ will be treated as a smooth, continuous function of its inputs. It follows that if $\mathbf{x}$ and $\mathbf{x}^{'}$ are close together, then the values of $Z(\mathbf{x})$ and $Z(\mathbf{x}^{'})$ should also be close. It is therefore reasonable to assume that $\eta(\mathbf{x})$ and $\eta(\mathbf{x}^{'})$ are highly correlated when $\mathbf{x}$ and $\mathbf{x}^{'}$ are close and viceversa. This assumption implies that each element of the training set provides considerable information about $\eta(\cdot)$ for inputs close to the corresponding design points. Hence, the uncertainty about the value of untried inputs decreases as the number of design points increases because the maximum distance from any design point decreases (recall criterion 2 above). O'Hagan[1] points out that the use of this extra information is the feature that accounts for greater efficiency of the use of emulators over Monte Carlo methods.

The discussion on how to determine valid covariance functions can become very technical and further details can be consulted in Satner *et al.*[4] A popular choice of a covariance function is the one we will adopt hereafter, namely

$$Cov(\eta(\mathbf{x}), \eta(\mathbf{x}^{'})) = \sigma^2 e^{-(\mathbf{x}-\mathbf{x}^{'})^T \mathbf{B}(\mathbf{x}-\mathbf{x}^{'})} \tag{8}$$

where $\mathbf{B}$ is a positive definite diagonal matrix. Observe that $C(\mathbf{x}, \mathbf{x}) = 1$ and that it decreases as the distance between two points increases, which is clearly a desired property for a correlation function.

As a consequence of the above discussion, the prior knowledge about $\eta(\cdot)$, given $\boldsymbol{\beta}$ and σ^2, is represented as having a Gaussian process distribution with mean function expressed by Eq. (7) and covariance expressed by Eq. (8). The latter is symbolized by

$$\eta(\cdot)|\boldsymbol{\beta}, \sigma^2 \sim N(\mathbf{h}(\cdot)^T \boldsymbol{\beta}, \sigma^2 C(\cdot, \cdot)) \tag{9}$$

The subjective information about the input and the unknown outputs is contained in this probability distribution. The next step is to update this prior belief by adding objective information. Suppose there are n design points that produce the vector of observations $\mathbf{y} = [\mathbf{y}_1 = \eta(\mathbf{x}_1), \ldots, \mathbf{y}_n = \eta(\mathbf{x}_n)]^T$ and the output of $\eta(\cdot)$ at an untried input $\mathbf{x}$ is to be estimated. It can be shown that

$$\eta(\cdot)|\mathbf{y}, \boldsymbol{\beta}, \sigma^2 \sim N(m^*(\cdot), \sigma^2 C^*(\cdot, \cdot)) \tag{10}$$

However, Haylock and O'Hagan[9] point out that knowing the value of $\boldsymbol{\beta}$ or σ^2 beforehand is usually unrealistic. Carrying out the suitable integration, they remove the conditioning on $\boldsymbol{\beta}$ to obtain

$$\eta(\cdot)|\mathbf{y}, \sigma^2 \sim N(m^{**}(\cdot), \sigma^2 C^{**}(\cdot, \cdot)) \tag{11}$$

To consult the complete expressions of $m^*(\cdot)$, $C^*(\cdot, \cdot)$, $m^{**}(\cdot)$ and $C^{**}(\cdot, \cdot)$, as well as the procedure to obtain the posterior distribution (11) from the prior distriburtion (9), consult the Appendix. There, it can be seen that $m^{**}(\cdot)$ does not include any terms involving $\eta(\cdot)$. Thus, it provides a fast approximation of $\eta(\mathbf{x})$ for any $\mathbf{x}$ (recall again criterion 2 above). The conditioning on σ^2 can also be eliminated. Haylock and O'Hagan[9] also show that

$$\frac{\eta(\mathbf{x}) - m^{**}(\mathbf{x})}{\widehat{\sigma}\sqrt{\frac{(n-q-2)C^{**}(\mathbf{x})}{n-q}}} \sim t_{n-q} \tag{12}$$

which is a t-distribution with $n - q$ degrees of freedom (not to be confused with the degrees of freedom in a finite element method context). The expression for $\widehat{\sigma}$ and the meaning of q are also contained in the Appendix.

In light of the above discussion, the algorithm to approximate the simulator defined by Eq. (5) is the following.

Algorithm 1: Emulation

1. Select n initial frequency values $\omega_1, \ldots, \omega_n$.
2. Obtain the vector of observations $\mathbf{y} = [\mathbf{y}_1 = \eta(\omega_1), \ldots, \mathbf{y}_n = \eta(\omega_n)]^T$.
3. Update the prior distribution (9), which contains subjective information, by adding the objective information $\mathbf{y}$. This will enable the calculation of $m^{**}(\cdot)$, the mean of the updated posterior distribution given the data $\mathbf{y}$. As already mentioned, such mean constitutes an approximation of $\eta(\omega)$ for any ω.

IV. Further Implementation Details

IV.A. Selection of the Training Set

When dealing with expensive computer codes, the selection of an initial design $D = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ to run the code at must be done carefully. It would be ideal to extract the most information about $\eta(\cdot)$ out of the minimum number of evaluations possible. The following is a brief overview of existing methods. For the next discussion, let Δ_η denote the domain of $\eta(\cdot)$.

One strategy for selecting a set of inputs to evaluate the code is to choose D such that its elements are evenly spread throughout Δ_η. In that case, random sampling from the distribution of the inputs could be a suitable strategy. Nevertheless, this scheme might have a disadvantage. Consider a case where the output is influenced by only a few components of each input vector. McKay et al.[21] proposed *Latin hypercube sampling* as a solution to this problem. Latin hypercube sampling can be viewed as an extension of Latin square designs to higher dimensions. Each dimension is guaranteed to be fully represented. Furthermore, it offers the advantage of being a computationally cheap method.

The notion of points spread evenly throughout Δ_η can have many interpretations, though. Johnson *et al.*[22] provide a criterion for quantifying this property. Given an arbitrary design D, they call it a *maximin design* if no two points are too close together, that is, if the minimum distance between any two points is maximized. In an analogous way, they define *minimax designs*.

Both Latin hypercube sampling and minimax and maximin designs rely on the idea of a set of inputs being evenly spread throughout Δ_η. Another criterion is what Shewry and Wynn[23] call *maximum entropy sampling*. Its objective is to maximize the gain in information for prediction at unsampled sites.

The choice of the initial design D is an active research area. A copious amount of literature on the subject is available and the overview presented here is by no means exhaustive. There exist a number of different strategies such as criterion based designs, combined designs and designs based on optimization procedures. A more complete account can be consulted in Satner et al.[4]

IV.B. Selection of the Smoothness Parameters

It was previously assumed that $\eta(\cdot)$ is a smooth and continuous function. Additionally, the correlation function between any two inputs, $\mathbf{x}$ and $\mathbf{x}^{'}$, was defined. A crucial component of such correlation function is the diagonal matrix $\mathbf{B}$, which contains what are known as *smoothness parameters*. Intuitively, these parameters specify how far an untried input needs to go from a design point before the uncertainty becomes appreciable. In other words, $\mathbf{B}$ determines how close the inputs $\mathbf{x}$ and $\mathbf{x}^{'}$ need to be such that the covariance between $\eta(\mathbf{x})$ and $\eta(\mathbf{x}^{'})$ takes a particular value.

There are at least two available techniques to estimate the smoothness parameters from the vector of observations $\mathbf{y} = [\mathbf{y}_1 = \eta(\mathbf{x}_1), \ldots, \mathbf{y}_n = \eta(\mathbf{x}_n)]^T$. The first one is called cross-validation and proceeds as follows. Define $\mathbf{y}_{-j}$ as the vector that excludes the j-th observation from $\mathbf{y}$. Let $\rho(\cdot)$ be a distance function and let the value of $\mathbf{B}$ be given.

Algorithm 2: Cross-Validation

For $j = 1, \ldots, n$

1. Calculate $m^{**}_{-j}(\cdot) \equiv E[\eta(\cdot)|\mathbf{y}_{-j}]$.
2. Calculate $d_j \equiv \rho(m^{**}_{-j}(\cdot), \eta(\mathbf{x}_j))$.
3. Find $\mathbf{b}$ that minimizes $\sum_{j=1}^{n} d_j$, where $\mathbf{b}$ is the diagonal of $\mathbf{B}$ and thus the vector of smoothness parameters.

Another technique to estimate the smoothness parameters is to derive the density function $f(\mathbf{B}|\mathbf{y})$ and obtain a maximum likelihood estimator of $\mathbf{b}$. The details can be consulted in Haylock.[24]

V. Applications

V.A. Spring-Mass System

Consider the damped three-degree-of-freedom spring-mass system shown in Figure 4. The mass of each block is 1 kg and the stiffness of each spring is 1 N/m. The viscous damping constant of the damper associated with each block is 0.2 Ns/m.

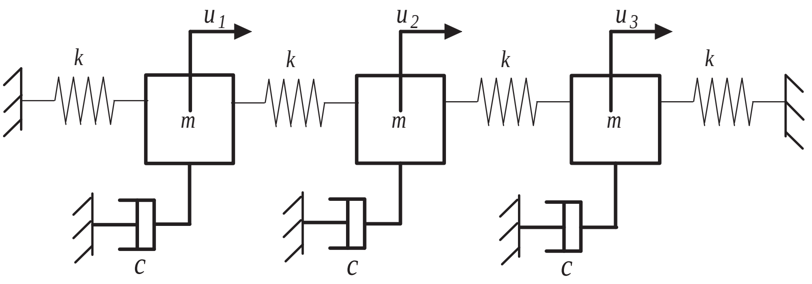


Figure 4. Three-degree-of-freedom damped spring-mass system; $m = 1$ kg, $k = 1$ N/m, $c = 0.2$ Ns/m

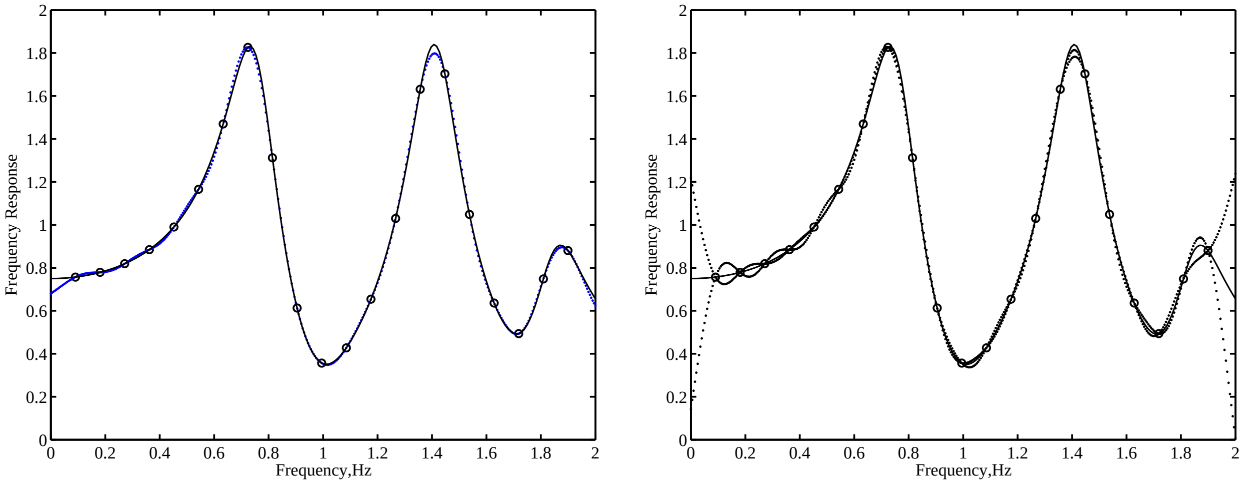
Suppose the dynamic response when only the first mass is subjected to unit initial displacement is to be obtained. For this simple case, the FRF has the following form

$$\phi_k(\omega) = \left[\sum_{j=1}^{3} \frac{\widetilde{\mathbf{x}}_j^T \bar{\mathbf{f}}(i\omega)}{-\omega^2 + 2i\omega\zeta_j\omega_j + \omega_j^2} \widetilde{\mathbf{x}}_j \right]_k \tag{13}$$

where $k = 1, \ldots, 3$ and for all j, ω_j are the natural frequencies, $\widetilde{\mathbf{x}}_j$ are the normal modes and ζ_j are the damping ratios. Since $\phi_k(\cdot)$ is a complex-valued function, the simulator will be

$$\eta(\omega) = |\phi_k(\omega)| = \sum_{j=1}^{3} \frac{\widetilde{\mathbf{x}}_j^T \bar{\mathbf{f}}(i\omega)}{\sqrt{(\omega_j^2 - \omega_j)^2 + (2\omega\omega_j\zeta_j)^2}} \widetilde{x}_{kj} \tag{14}$$

For k fixed, $\eta(\cdot)$ is a single variable function. That simplifies the task of choosing an initial design, since the output can only be sensitive to the only dimension of the input. Hence, a design that is uniformly spread throughout the input domain will suffice. Twenty-one equally-spaced design points were selected. The smoothness parameters were then calculated. Algorithm 1 was then applied to construct the corresponding emulator for the FRF. The results for $k = 1$ are shown in Figure 5. As before, the circles represent the training runs, the dots represent the mean emulator and the uncertainty bounds respectively, and the solid line is the true value of the simulator. It is important to bear in mind that this is a purely illustrative exercise. The knowledge of the true value of the simulator at a limited number of design points is assumed. Note how closely does the approximation provided by the mean of the emulator follows the simulator.



(a) Mean of the emulator.

(b) Probability bounds: two standard deviations.

Figure 5. Emulation of $\eta(\omega) = |\phi_1(\omega)|$ for a damped spring-mass system with three degrees of freedom. The displacement vector is $[1, 0, 0]$. The solid line is the true output of the simulator. The circles represent the training runs. The dots are, respectively, the mean of the emulator and the probability bounds for that mean.

V.B. Cantilever Plate with Simulated Data

The second simulator measures the frequency response of a cantilever steel plate with a slot. Suppose the plate is excited by a unit harmonic force and the response is calculated at six different nodes (see Figure 6).

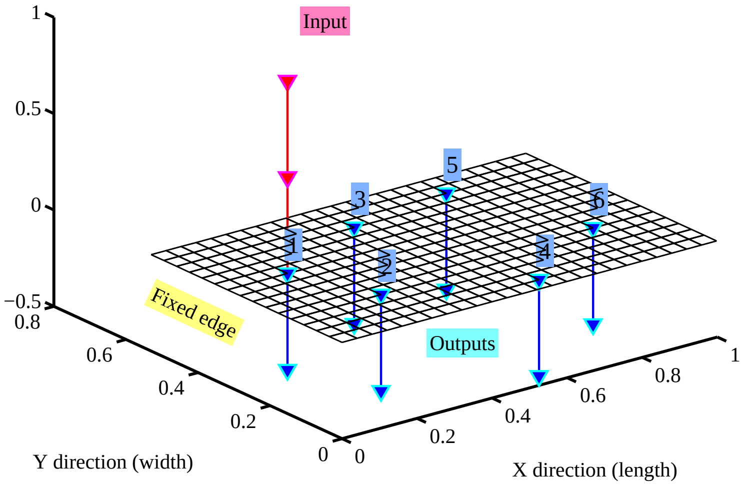
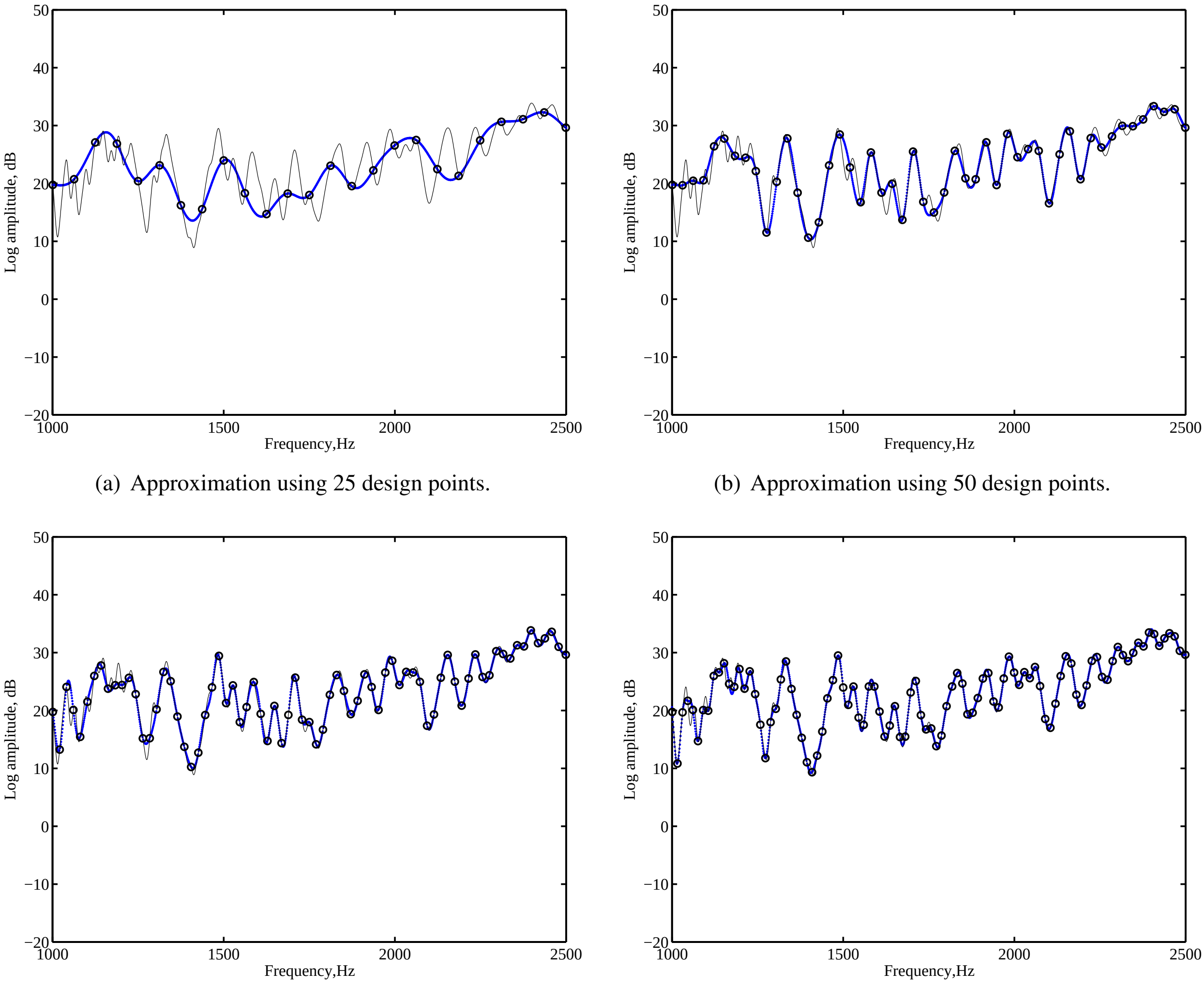


Figure 6. Finite Element model of a steel cantilever plate: 25 × 15 elements, 416 nodes, 1200 degrees of freedom.

The standard four-noded thin plate bending element (resulting in 12 degrees of freedom per element) is assumed. The plate is divided into 25 $\times$ 15 elements for the numerical calculations. The resulting system has 1200 degrees of freedom. The simulator of this FRF was proposed by Adhikari.[25] Figure 7 shows a comparison between selecting a different number of design points to approximate the response of node 1 to vibration in the medium-frequency range. Note how the approximation is improved, the more design points are used.



(a) Approximation using 25 design points.

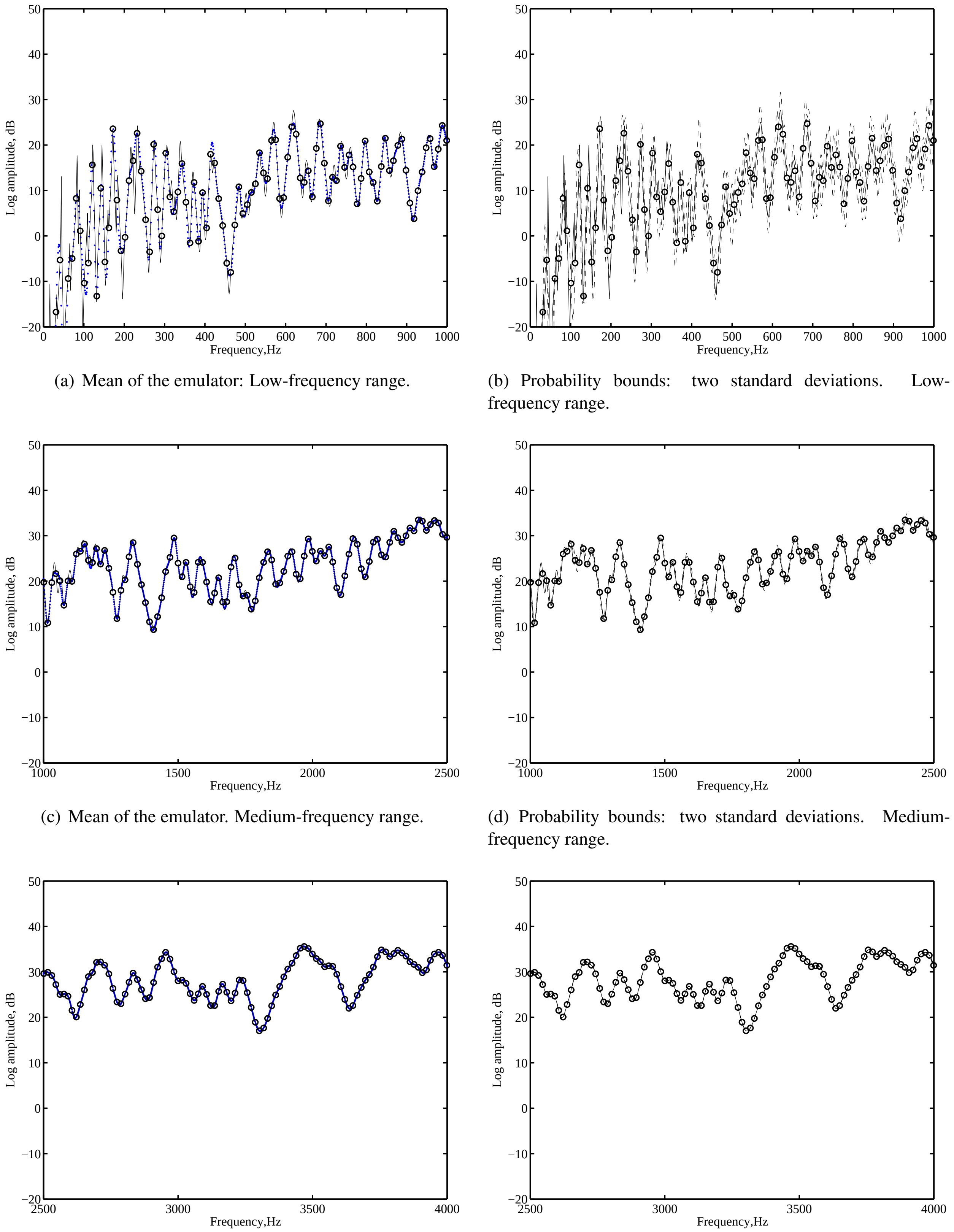
(b) Approximation using 50 design points.

(c) Approximation using 75 design points.

(d) Approximation using 100 design points.

Figure 7. Improvement in the approximation (dots) of the true output (line) of the simulator in the medium-frequency range, when an increasing number of training runs (circles) is used.

Emulation was then performed for the response of node 1 to vibration in the low, medium and high frequency ranges. For each of them, 100 equally-spaced design points were selected and the smoothness parameters calculated. The mean of the emulator and the corresponding probability bounds across the frequency range are shown in Figure 8.



(a) Mean of the emulator: Low-frequency range.

(b) Probability bounds: two standard deviations. Low-frequency range.

(c) Mean of the emulator. Medium-frequency range.

(d) Probability bounds: two standard deviations. Medium-frequency range.

(e) Mean of the emulator. High-frequency range.

(f) Probability bounds: two standard deviations. High-frequency range.

Figure 8. Emulation of the response of node 1 to vibration across the frequency range. The true output of the simulator is represented by the solid line. The training runs are symbolized by circles. The dots represent the mean of the emulator in the figures on the left column, and probability bounds for that mean in the figures on the right column.

V.C. Cantilever Plate with Experimental Data

Consider now a setup in which there is no simulator available, due perhaps to the lack of knowledge of the physics of the system. Again, the aim is to approximate the response function of a plate subject to vibration. The experiment described was carried out by Adhikari et. al.[11] A rectangular steel plate with uniform thickness was used. The physical and geometrical properties of the plate are shown in Table 1.

Plate Properties	Numerical values
Length	998 mm
Width	530 mm
Thickness	3.0 mm
Mass density	7860 kg/m^3
Young's modulus	2.0×10^5 MPa
Poisson's ratio	0.3
Total weight	12.47 kg

Table 1. Material and geometric properties of the cantilever plate considered for the experiment

The plate was clamped along one edge using a clamping device. The clamping device was attached to the top of a heavy concrete block and the whole assembly was placed on a steel table. The plate had a mass of approximately 12.47 kg and special care was been taken to ensure its stability and to minimize vibration transmission. The plate was divided into 375 elements (25 along the length and 15 along the width). Taking one corner of the cantilevered edge as the origin, co-ordinates were assigned to all of the nodes. Accelerometers were attached to these nodes. This approach allowed easy correlation to a finite element model. The test rig is shown in Figure 9.

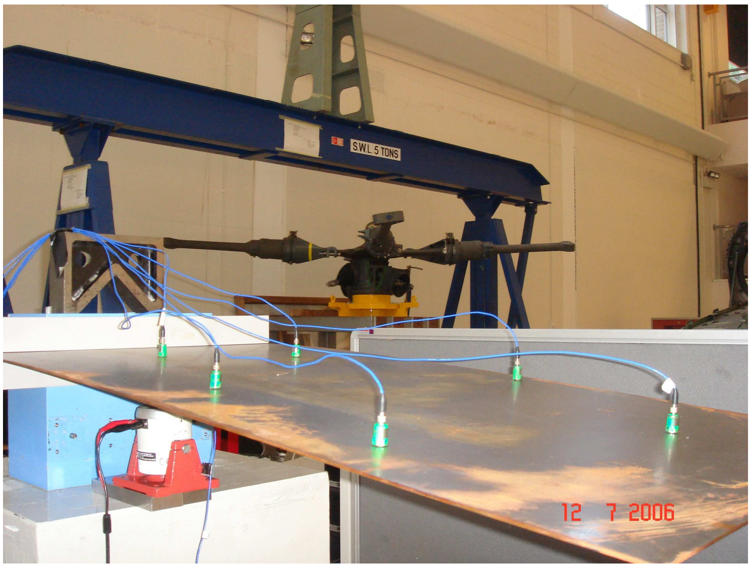
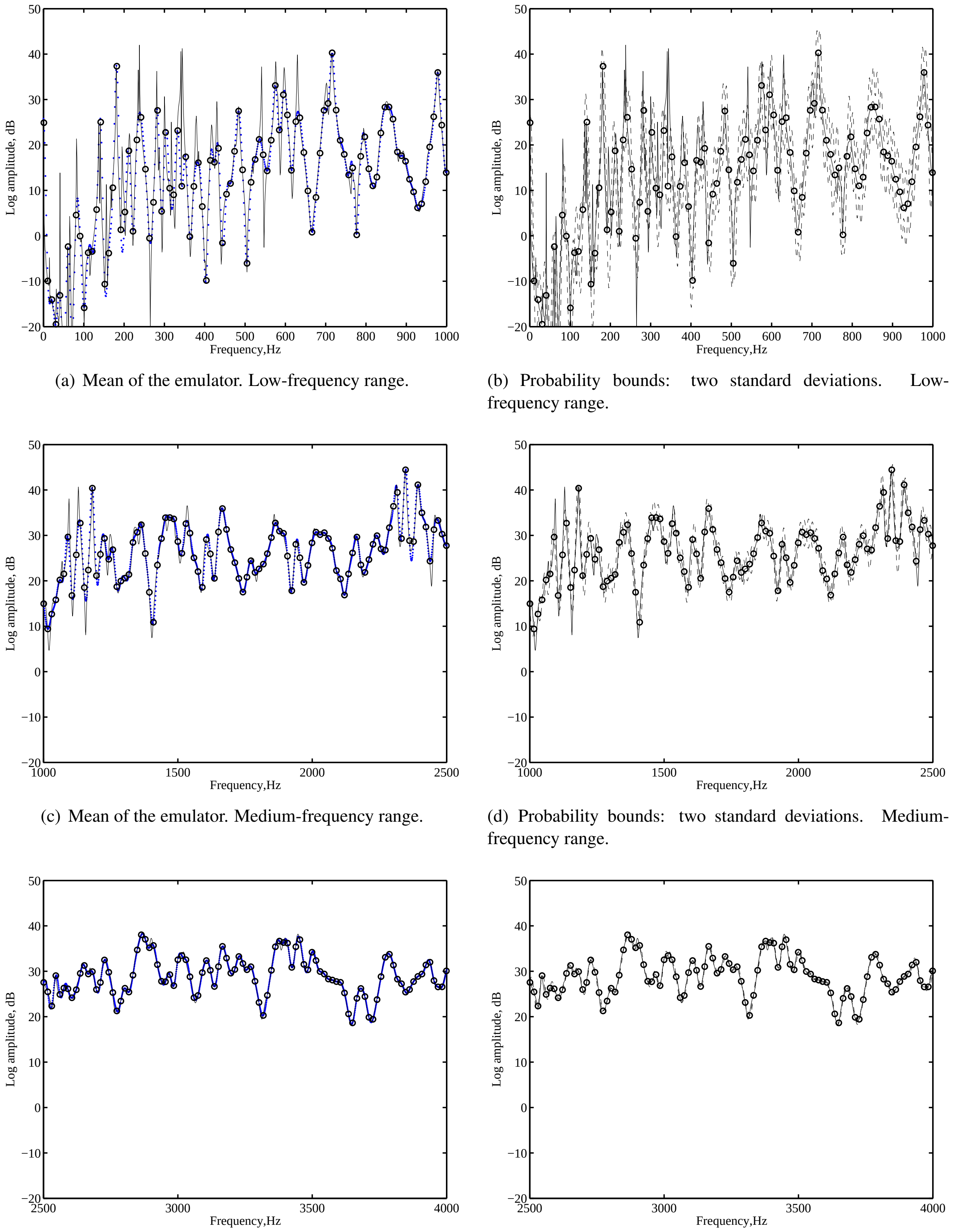


Figure 9. Experimental setup.

In an experimental context like the one described, there are only real measurements available. To take advantage of emulation, these measurements should be generated using a space-filling design (or any of the strategies explained in Subsection IV.A). Those measurements will then act as the training set and Algorithm 1 can be applied the same way as in the previous cases. This will reduce the number of experimental runs necessary to obtain an empirical FRF. The results of doing so are shown in Figure 10.



(a) Mean of the emulator. Low-frequency range.

(b) Probability bounds: two standard deviations. Low-frequency range.

(c) Mean of the emulator. Medium-frequency range.

(d) Probability bounds: two standard deviations. Medium-frequency range.

(e) Mean of the emulator. High-frequency range.

(f) Probability bounds: two standard deviations. High-frequency range.

Figure 10. Emulation of the the response of node 1 to vibration across the frequency range, where the initial design is based on actual measurements. The experimental results are represented by the solid line. The measurements, used as the training runs, are symbolized by circles. The dots represent the mean of the emulator in the figures on the left column, and probability bounds for that mean in the figures on the right column.

VI. Conclusions

Computer codes for Structural Dynamics can become very expensive to run, as the number of degrees of freedom in the system increases. This can pose considerable difficulty for real-life problems where numerical models might involve millions of degrees of freedom. To tackle this difficulty, the use of emulators has been proposed, since they provide a fast approximation to the output of the original code using in only a few training runs. The general theory behind the construction of emulators was reviewed and applied to different cases with the following objectives in mind:

a) *Computational cost.* Can the output of a computer code be approximated using only a few trial runs?

b) *Efficiency.* Can the number of floating point operations in an expensive code be dramatically reduced but still produce a satisfactory output?

c) *Interpolation of experimental data.* Can experimental data be confidently interpolated to cope with the lack of a mathematical/computer model?

With regards to the first objective, an emulator for a simple spring-mass system with three degrees of freedom was constructed to illustrate how the mean of the emulator gives a plausible approximation to the FRF. For the second objective, a bigger system with 1200 degrees of freedom was considered. The corresponding dynamic response was approximated using an emulator. The advantage of doing so was the improvement in computational efficiency with respect to the original model, in which the dynamical stiffness matrix had to be inverted 4000 times. Adopting the emulator approach, the matrix had to be inverted only 300 times, equal to the number of training runs necessary to approximate the output of the original code. This accounts for only 7.5% of the original computer effort. The results were particularly appealing for the medium and high-frequency ranges. This is encouraging since in systems with millions of degrees of freedom, these are the computationally demanding ranges of vibration. Finally, a frequency response function obtained via experimental methods was emulated. Real data was used as the set of training runs necessary to construct an emulator and the real experimental output was compared with the corresponding approximation. This scheme might be useful when the physics of a system are not well known and there is no mathematical or computational model upon which to construct an simulator.

Appendix

A.1. Updating of the Prior Distribution

In this section the process of updating the prior distribution (9) to obtain the posterior distribution (11) is outlined. Define $\mathbf{H} = [\mathbf{h}(\mathbf{x}_1), \ldots, \mathbf{h}(\mathbf{x}_n)]^T$ and $\mathbf{A} \in \mathbb{R}^{n\times n}$ with $\mathbf{A}_{\ell j} = C(x_\ell, x_j)\ \forall \ell, j \in \{1, \ldots, n\}$. Hence

$$\mathbf{y}|\boldsymbol{\beta}, \sigma^2 \sim N(\mathbf{H}\boldsymbol{\beta}, \sigma^2\mathbf{A}) \tag{15}$$

To incorporate the objective information $\mathbf{y}$ and obtain the distribution of $\eta(\cdot)|\mathbf{y}, \boldsymbol{\beta}, \sigma^2$, use of the following result, given in Krzanowski.[26]

Theorem. Let $\mathbf{z} \in \mathbb{R}^n$ be a random vector such that $\mathbf{z} \sim N(\boldsymbol{\mu}, \Sigma)$. Partition $\mathbf{z}$ as $\begin{pmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \end{pmatrix}$, where $\mathbf{z}_1 \in \mathbb{R}^p$ and $\mathbf{z}_2 \in \mathbb{R}^{n-p}$. Consequently, partition $\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{pmatrix}$ and $\Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}$, so that $\mathbf{E}[\mathbf{z}_j] = \boldsymbol{\mu}_j$ and $Cov(\mathbf{z}_j, \mathbf{z}_k) = \Sigma_{jk}$. Then, $\mathbf{z}_1|\mathbf{z}_2 \sim N(\widetilde{\boldsymbol{\mu}}, \widetilde{\Sigma})$, where $\widetilde{\boldsymbol{\mu}} = \boldsymbol{\mu}_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{z}_2 - \boldsymbol{\mu}_2)$ and $\widetilde{\Sigma} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$.

From this result, it follows that

$$\eta(\cdot)|\mathbf{y}, \boldsymbol{\beta}, \sigma^2 \sim N(m^*(\cdot), \sigma^2 C^*(\cdot, \cdot)) \tag{16}$$

where

$$m^*(x) = \mathbf{h}(x)^T \boldsymbol{\beta} + \mathbf{t}(x)\mathbf{A}^{-1}(\mathbf{y} - \mathbf{H}\boldsymbol{\beta}) \tag{17}$$

$$C^*(\mathbf{x}, \mathbf{x}^{'}) = C(\mathbf{x}, \mathbf{x}^{'}) - \mathbf{t}(\mathbf{x})^T A^{-1} \mathbf{t}(\mathbf{x}^{'}) \tag{18}$$

$$\mathbf{t}(\mathbf{x}) = [C(\mathbf{x}, \mathbf{x}_1), \ldots, C(\mathbf{x}, \mathbf{x}_n)]^T \tag{19}$$

Removing the conditioning on $\boldsymbol{\beta}$ using standard integration techniques, obtain the posterior distribution

$$\eta(\cdot)|\mathbf{y}, \sigma^2 \sim N(m^{**}(\cdot), \sigma^2 C^{**}(\cdot, \cdot)) \tag{20}$$

where

$$m^{**}(\mathbf{x}) = \mathbf{h}(\mathbf{x})^T \widehat{\boldsymbol{\beta}} + \mathbf{t}(\mathbf{x})\mathbf{A}^{-1}(\mathbf{y} - \mathbf{H}\widehat{\boldsymbol{\beta}}) \tag{21}$$

$$C^{**}(\mathbf{x}, \mathbf{x}^{'}) = C^*(\mathbf{x}, \mathbf{x}^{'}) + (\mathbf{h}(\mathbf{x})^T - \mathbf{t}(\mathbf{x})^T \mathbf{A}^{-1}\mathbf{H})(\mathbf{H}^T \mathbf{A}^{-1}\mathbf{H})^{-1}(\mathbf{h}(\mathbf{x})^T - \mathbf{t}(\mathbf{x})^T \mathbf{A}^{-1}\mathbf{H})^T \tag{22}$$

$$\widehat{\boldsymbol{\beta}} = (\mathbf{H}^T \mathbf{A}^{-1}\mathbf{H})^{-1}\mathbf{H}^T \mathbf{A}^{-1}\mathbf{y} \tag{23}$$

Regarding Eq. (12), q is the rank of $\mathbf{H}$. The term $\widehat{\sigma}$ in the expression is equal to

$$\widehat{\sigma}^2 = \frac{\mathbf{y}^T(\mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{H}(\mathbf{H}^T \mathbf{A}^{-1}\mathbf{H})\mathbf{H}^T \mathbf{A}^{-1})\mathbf{y}}{n - q - 2} \tag{24}$$

Acknowledgments

FADO gratefully acknowledges the support of the Engineering and Physical Sciences Research Council (EPSRC) for the award of a studentship through an Ideas Factory grant. SA gratefully acknowledges the support of EPSRC through the award of an Advanced Research Fellowship.

References

[1] O'Hagan, A., "Bayesian analysis of computer code outputs: A tutorial," *Reliability Engineering & System Safety*, Vol. 91, No. 10-11, 2006, pp. 1290–1300.

[2] Goldstein, M., "Uncertainty analysis for complex physical models," 2007, 30th Research Student's Conference in Probability and Statistics in Public Resource, March 27-29, 2007, Durham, UK.

[3] Thomke, S., Holzner, M., and Gholami, T., "The crash in the machine," *Scientific American*, Vol. 280, No. 3, 1999, pp. 92–97.

[4] Satner, T., Williams, B., and Notz, W., *The Design and Analysis of Computer Experiments*, Springer Series in Statistics, London, UK, 2003.

[5] Kennedy, M. C., Anderson, C. W., Conti, S., and O'Hagan, A., "Case studies in Gaussian process modelling of computer codes," *Reliability Engineering and System Safety*, Vol. 91, No. 10-11, 2006, pp. 1301–1309.

[6] Challenor, P., Hankin, R., and Marsh, R., *Avoiding Dangerous Climate Change*, Cambridge University Press, Cambridge, UK, 2006.

[7] Rougier, J., "Probabilistic inference for future climate using an ensemble of climate model evaluations," *Climatic Change*, Vol. 81, 2007, pp. 247–264.

[8] Bates, R., Kennet, R., Steinberg, D., and Wynn, H., "Achieving robust design from computer simulations," *Quality Technology & Quantitative Management*, Vol. 3, 2006, pp. 161–177.

[9] Haylock, R. and O'Hagan, A., *Bayesian Statistics 5*, Oxford University Press, Oxford, UK, 1996.

[10] Oakley, J. and O'Hagan, A., "Probabilistic Sensitivity Analysis of Complex Models: A Bayesian Approach," *Journal of the Royal Statistical Society B*, Vol. 66, 2004, pp. 751–769.

[11] Adhikari, S., Friswell, M. I., and Lonkar, K., "Uncertainty in structural dynamics: Experimental case studies on beams and plates," *Proceedings of the Computational Methods in Structural Dynamics and Earthquake Engineering (COMPDYN)*, Crete, Greece, June 2007.

[12] Bathe, K.-J., *Finite Element Procedures*, Prentice Hall Inc., Englewood Cliffs, New Jersey, USA, 1995.

[13] Géradin, M. and Rixen, D., *Mechanical Vibrations*, John Wiely & Sons, New York, NY, 2nd ed., 1997, Translation of: Théorie des Vibrations.

[14] Hughes, T. J. R., *The Finite Element Method : Linear Static and Dynamic Finite Element Analysis*, Dover Publications, New York, USA, 2000.

[15] Maia, N. M. M. and Silva, J. M. M., editors, *Theoretical and Experimental Modal Analysis*, Engineering Dynamics Series, Research Studies Press, Taunton, England, 1997, Series Editor, J. B. Robetrs.

[16] Meirovitch, L., *Principles and Techniques of Vibrations*, Prentice-Hall International, Inc., New Jersey, 1997.

[17] Newland, D. E., *Mechanical Vibration Analysis and Computation*, Longman, Harlow and John Wiley, New York, 1989.

[18] Zienkiewicz, O. C. and Taylor, R. L., *The Finite Element Method*, McGraw-Hill, London, 4th ed., 1991.

[19] Adhikari, S., "Modal analysis of linear asymmetric non-conservative systems," *ASCE Journal of Engineering Mechanics*, Vol. 125, No. 12, December 1999, pp. 1372–1379.

[20] Kennedy, M. C. and O'Hagan, A., "Bayesian calibration of computer models," *Journal of the Royal Statistical Society Series B-Statistical Methodology*, Vol. 63, 2001, pp. 425–450, 3.

[21] McKay, M., Conover, W., and Beckman, R., "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics*, Vol. 21, 1979, pp. 239–245.

[22] Johnson, M. E., Moore, L. M., and Ylvisaker, D., "Minimax and maximin distance designs," *Journal of Statistical Planning and Inference*, Vol. 26, No. 2, 1990, pp. 131–148.

[23] Shewry, M. and Wynn, H., "Maximum Entropy Sampling," *J. Appl. Statist.*, Vol. 14, 1987, pp. 165–170.

[24] Haylock, R., *Bayesian Inference About Outputs of Computationally Expensive Algorithms with Uncertainty on the Inputs*, Ph.D. thesis, University of Nottingham, Nottingham, UK, 1966.

[25] Adhikari, S., "Matrix variate distributions for probabilistic structural mechanics," *AIAA Journal*, Vol. 45, No. 7, July 2007, pp. 1748–1762.

[26] Krzanowski, W., *Principles of Multivariate Analysis*, Oxford University Press, Oxford, UK, 2000.