# Understanding uncertainty in a SWAN wave model using a Bayesian Emulator

Jonathan Hardwick, Helen C. M. Smith, and Peter Challenor

*Abstract*—**Numerical simulation is used widely in the marine renewable energy sector. Wave and flow models are used to understand and predict the conditions experienced at offshore energy sites. Like all numerical simulations, wave models have uncertainties in their output caused by uncertainty about the various input data (which may themselves be model outputs), and uncertainty about how well the model simulates the real world. Understanding these uncertainties is important in order to hold confidence in the models accuracy. Classical Monte Carlo uncertainty analysis requires a large number of model runs which is impossible in large complex models if the computational run time is more than a few seconds. By substituting a much more computationally efficient mathematical model, known as an emulator, for the complex simulation then processing time can be decreased to a level where uncertainty analysis can be undertaken. A simple 'toy' wave model has been produced using SWAN. By using a Bayesian methodology on output from a small number of correctly designed model runs, a mathematical emulator is constructed to provide a statistical approximation of output from the model. Importantly this emulator provides not just an approximation of the output but a full probability distribution describing how close the emulator output is to the model. As this emulator provides results in a fraction of a second (compared to several seconds for the toy simulator and considerably longer for actual wave models) it can be run many thousands of times as is required for a Monte Carlo analysis. This paper describes the methodology used to construct an emulator of a simulation and provides method and results using the emulator to undertake uncertainty quantification. The methods described here can be scaled up and employed on large wave models, flow models or any deterministic numerical simulator.**

*Index Terms*—**Bayesian, Emulation, SWAN, Uncertainty Quantification, Wave Modelling**

## I. Introduction

MARINE renewable energy projects are heavily reliant on numerical simulations. Computational modelling is used across many facets of MRE from initial scoping studies and resource assessments through to ongoing O&M and decommissioning. Each model is computed to simulate a real world process and therefore will contain uncertainties as to how well the output from the models reflect that process. The sources and effects of uncertainties in are often overlooked, once a model has been validated against some trusted data then it is often considered acceptable without any deeper analysis. For the development of commercial MRE to continue, potential sites need to be identified and the resource quantified. Deploying current and wave measuring equipment is time consuming, costly and limited to the scope of the deployment. It is therefore essential to undertake computational modelling to determine the conditions at the site and to determine whether a site is suitable for development. As the MRE industry expands and more sites are investigated the reliance on accurate modelling is likely to increase.

Several computational codes have been developed over the past few decades to simulate ocean waves (SWAN, WAVEWATCH, WAM, MIKE 21) and considerable research has been undertaken in modelling wave resources at a number of different sites and under various conditions [1]–[3]. Spectral wave models, such as SWAN, take some boundary values and initial conditions, undertake some processes developed from mathematical equations and physical laws and provide an output. Conventionally they are then calibrated, verified and validated using empirical measurements and existing research (as in [4], [5]); if the model output is close to the measured data then the model can be said to be accurate and used to inform projects. Sensitivity studies have been carried out to categorise which inputs most effect the output and to tune some of the many physical constants and numerical schemes employed in the models. Sensitivity studies can be used to determine the conditions in which the model functions best and where the sources of inaccuracy arise however previous studies often stop short of giving a full understanding of any sources of inaccuracy. Understanding uncertainty is important in order to gain a more complete understanding of how well a simulation represent reality, particularly if project success and financial investment is reliant on modelled data. If this is well understood then the results of the model can prove extremely useful to many facets of MRE. Whether the outputs from a model are an accurate and complete picture of the real world or whether they just present the right direction for further investigation is essential to know before you can utilise the data on a project.

There will always be uncertainty in any model prediction; no simulation will perfectly represent real life. Uncertainty about each input value and uncertainty in the numerical calculations and assumptions undertaken by the model each have an impact on how well the model represents reality. There is little consistency in the handling uncertainty in computer models. While some studies provide a thorough and robust analysis of the uncertainties in their numerical simulations [6], [7] others limit the discussion to inaccuracies in the

Jonathan Hardwick and Helen C. M. Smith are are with the Renewable Energy Group, University of Exeter, Penryn Campus, Penryn, TR10 8PP, UK (e-mail: j.p.hardwick@exeter.ac.uk [Jonathan Hardwick]).

Peter Challenor is with Department of Mathematics, University of Exeter, Streatham Campus, Exeter EX4 4QJ, UK.

input data limiting the output reliability. As traditional Monte Carlo methods for uncertainty quantification are highly computationally intensive it is common for developers of large models to use a limited ensemble of input parameters to develop a measure of uncertainty. The limitations of this method depend largely on the particular model and the sample size [8], [9].

Another approach to uncertainty quantification is through the development of an emulator using Bayesian statistical methods [10]–[13]. An emulator is a statistical regression model used to simulate the output from a more complex numerical model. This emulator can provide output in a tiny fraction of the time of a complex model. The MUCM (managing uncertainty in complex models) consortium produced a considerable body of work developing the method for UQ across a number of disciplines, the list of publications and reports is too numerous to include here and the reader is directed to the MUCM web pages[1].

This paper demonstrates the methodology of creating a Bayesian emulator to undertake UQ on a simple 'toy' SWAN model. While the toy model is an uncomplicated simulator that can be resolved in a few seconds the principles discussed can be scaled up to any deterministic simulator, it is a particularly useful tool when working with large complex simulators where resolving the thousands (or millions) of runs necessary for traditional Monte Carlo type analyses are impossible.

The term *simulator* is used throughout to refer to a complex model, usually SWAN. *Emulator* refers to a statistical ragression model which emulates the simulator. The term *model* has different meanings depending on the context.

## II. Principles of Model Emulation

Numerical simulators (or computer models) utilize mathematical understanding about the real-world combined with the processing power of modern computers to simulate quantities which would be difficult (or in some cases impossible) to obtain by observation or measurement. A deterministic computer model, no matter how complex, takes some inputs and undertakes mathematical calculations in order to obtain some desired output. For example a simulator of the ocean waves around a particular MRE site would take as inputs a whole range of quantities that would impact on the waves across the site, for example: wind, current and bathymetry. Its outputs would then include a range of details about the wave conditions, for example: wave spectra, parameters or spatial quantities. Any simulator can be regarded as a function $f$, with the simulator inputs comprising the function's argument $x$ and the simulator output(s) comprising the function value $y$. Formally any simulator can be described as: $y = f(x)$, noting that $x$ and $y$ may be either scalar in the case of a single input or output, or vectors containing the values of multiple input and output parameters. For the purposes of model emulation the specific numerical schemes and calculations that occur within the simulator can be ignored and the model considered a black-box.

If a far simpler surrogate function can be found, $\hat{f}$, which is a sufficiently good statistical approximation of $f$ then, the parameters provided by $\hat{f}(x)$ will be close to those from the original function. Moreover if $\hat{f}$ can be computed much faster than $f$ then it can be used to provide the data necessary for Monte Carlo type analyses. An emulator is a statistical regression model created to provide a close approximation of the output of a simulator. Furthermore, in the fully Bayesian approach discussed here, the emulator will provide a probability distribution predicting the output of the simulator. Three key criteria are essential when constructing a Bayesian emulator:

- The emulator must have zero deviation from the simulator at the design points $\hat{f}(x_i) = f(x_i) = y$
- In the space between the design points the mean value of $\hat{f}$ should be a plausible interpolation of the simulator output.
- The simulator to be emulated is a smooth continuous function. Care should be taken if the simulator switches between numerical schemes or has other 'tipping points' where the output may not be continuous.

Several forms of statistical regression emulators have been developed [10], this work will focus on the development and implementation of the Gaussian Process (GP) emulator. A GP is a function for which the probability distribution for all outputs are normally distributed. Hence a GP emulator is a regression model which assumes that uncertainty in each of the simulator outputs are normally distributed, this assumption is well justified in many cases and is discussed in detail in [14] The behaviour of the GP emulator is described by mean and co-variance functions.

### A. Prior Distribution

In the Bayesian framework probability models are built by first defining a prior distribution function from existing knowledge about the system, the model is later completed by adding information from observations, in the case of an emulator these observations comprise runs of the simulator. For a GP the prior mean and co-variance functions comprise the initial stage of building the emulator. The prior distribution $\pi(.)$ depends on hyper-parameters which are initially unknown and solved when the model is completed. The form of the hyper-parameters and relationship between them will depend on the knowledge of the model and the choice of prior functions. A popular choice of prior model is to use a linear mean function and Gaussian form co-variance function shown in Equation 1 and 2 for a model with $p$ inputs.

$$h(\cdot)^T \beta \tag{1}$$

$$c(x, x') = \exp\left\{ -\sum_{i=1}^{p} 0.5 \left[ \frac{(x_i - x_i')}{\delta_i} \right]^2 \right\} \tag{2}$$

[1]MUCM maintains a list of publications and technical reports at www.mucm.ac.uk

Where $h$ is a vector of $p+1$ known basis function. $\beta$, $\sigma^2$ and $\delta$ are hyper-parameters. The prior emulator is hence a Gaussian Process dependent on the unknown hyper-parameters (as in Equation 3).

$$\eta(\cdot) \mid \beta, \sigma^2, \delta \sim \boldsymbol{GP}\left(\mu, \sigma^2 c(\cdot,\cdot)\right) \tag{3}$$

### B. Design

In order to create an emulator, output from a simulator must be provided. A small number of carefully selected simulator runs provide the information required to build the emulator. The process of deciding the points at which the simulator should be resolved is known as the design. This is an important part of constructing the emulator. For relatively simple models the design may just be a case of running the model at a few equally spaced input values, however for large and complex simulators with a high number of input dimensions it is a complicated task requiring careful consideration. Deciding where the best points to run the simulator will depend on the type of model (and possibly on expert knowledge about its outputs). One common method is to select the design points with Latin Hyper-cube (LHC) sampling, this is a sampling method that ensures that input values are spread across the full range of the input space enabling a set of design points providing even coverage.

### C. Posterior Distribution

Updating the prior distribution with information from the the simulator output at the design points completes the posterior distribution $\pi^*(.)$. This involves finding values for the hyper-parameters. There are different methods to estimate or calculate the hyper-parameters depending on the choice of prior functions.

### D. Validation

Once an emulator has been constructed validation must be undertaken to ensure that it is a sufficiently good approximation of the simulator. Without sufficient validation of the emulator any results produce will not be usable. [15] provide a a detailed description of several validation methods. In order to complete the validation a further set of simulator runs must be completed. These runs are then used to facilitate statistical tests on the emulator to determine its validity. The examples in this study are validated by calculating the standard error and Mahalanobis distance. The standard error is computed as the difference in the output from the simulator and emulator mean divided by the posterior standard deviation at the validation points. The Mahalanobis distance is a measure of the distance between a point and the mean of a distribution. The example emulators created in this work are considered valid if all the validation points are within two standard deviations of the simulator output and that the Mahalanobis distance is close to the mean of the reference distribution.

## III. Toy Simulator

A small 'toy' SWAN model has been constructed to provide examples of model emulation. The simulator comprises a simple rectangular grid with constant water depth and allows for variation of the wind input and boundary conditions. Output is comprised of a number of wave parameters provided at a single location. For the examples specified here the only output of consideration is significant wave height. While this model can be evaluated fairly quickly on a simple home computer the principles can be applied to much larger and more complex simulations.

The toy simulator contains 100 x 20 nodes on a rectangular grid simulating a 10km x 2km rectangular pool with a constant 10m depth. It is evaluated as a stationary SWAN simulation, so SWAN attempts to resolve a solution in which the wave spectrum is stationary. A constant wind is applied at a 45 degree angle. Code was written to facilitate the toy model to be run several times with variation of two input parameters: wind speed and significant wave height at the boundary. All other inputs and model parameters are ignored for the process of this example. The toy simulator takes between approximately 5 and 30 seconds to evaluate depending on the input. Figure 1 is a diagram of the toy simulator set up showing the wind and boundary inputs.

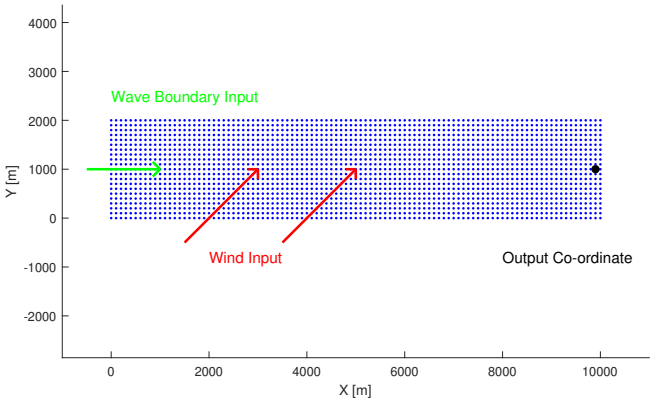


Figure 1. Diagram of toy simulator domain.

### A. Example 1: Single input and output

For the process of this first example, it is assumed that the simulator output: significant wave height is dependent on only a single input parameter: the input wind speed, all other parameters are fixed and ignored. An emulator is then constructed with this single input variable.

As there is just a single input and there is no expectation about the shape of the output, the design of the training sample is chosen to be set of equidistant wind speed values ranging from $5 - 20ms^{-1}$. The simulator output is shown in Figure 2. Five of these runs were selected as training points for the emulator shown in Table I. The points are scaled so that they lie in the range $[0, 1]$.

Knowledge about the system (specifically that stronger winds are associated with larger waves)

Table I
DESIGN POINTS FOR SINGLE INPUT TOY SIMULATOR

| | | | | |
|---|---|---|---|---|
| **Input** | Wind Speed $[ms^{-1}]$ | 5 | 9 | 13 |
| **Input** | Scaled Input | 0 | 0.29 | 0.57 |
| **Output** | $H_{m0}$ $[m]$ | 0.40 | 0.59 | 0.87 |
| **Input** | Wind Speed $[ms^{-1}]$ | 17 | 19 | |
| **Input** | Scaled Input | 0.86 | 1.0 | |
| **Output** | $H_{m0}$ $[m]$ | 1.08 | 1.17 | |

suggests that a linear mean function is an appropriate choice of prior. This linear mean function $\mu$ and a Gaussian correlation form co-variance function $\sigma^2 c(x, x')$ are chosen to construct the emulator. Which for the 1-dimensional case are given by Equation 4 and Equation 5 leading to the join prior distribution $\pi(\beta, \sigma^2, \delta)$ with unknown hyper-parameters $[\beta, \sigma^2, \delta]$.

$$\mu = \beta_1 x + \beta_2 \tag{4}$$

$$c\left(x, x'\right) = \exp\left\{-\frac{(x - x')}{\delta}\right\} \tag{5}$$

To develop the posterior model, values for the hyper-parameters need to be found. This can be a complex and procedure and choices about the form of the hyper-parameters can have a large effect on the effectiveness of the emulator. As this model uses a linear mean function with a weak prior information about $\beta$ and $\sigma^2$ (shown in Equation 6) analytical solutions for $\beta$ and $\sigma^2$ can be found conditional on the training sample output and the scaling parameter $\delta$.

$$\pi\left(\beta, \sigma^2, \delta\right) \propto \sigma^{-2} \pi_\delta\left(\delta\right) \tag{6}$$

The correlation length scale parameter $\delta$ is fixed for simplicity and estimated by maximising the likelihood function $\pi_\delta(\delta)$ with the simulator output at the design points. Hyper-parameters $[\beta, \sigma^2]$, can be found analytically once $\delta$ is calculated to complete the posterior function. The complete emulator is shown in Figure 3. There is zero code uncertainty at the design points as the emulator has been constructed from the simulator output at these points, as the emulator interpolates or extrapolates further away from the design points the variance from the mean function increases.
The emulator can be validated using some of the additional output values from the initial runs. Three further design points are selected and used to validate the emulator. (Table II)

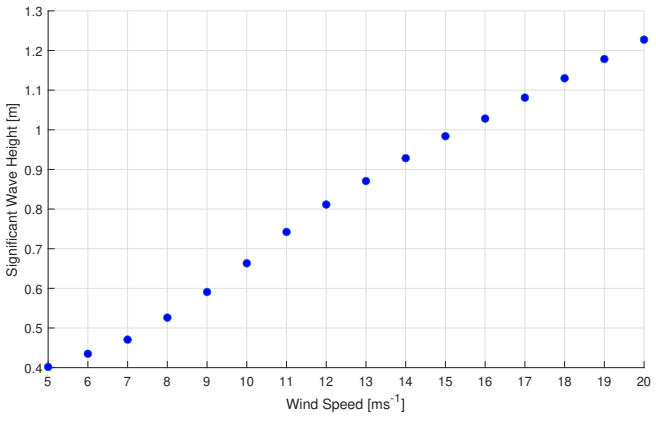


Figure 2. Output from the toy SWAN model with only one variable input, wind speed.

Table II
FURTHER DESIGN POINTS FOR VALIDATION

| | | | | |
|---|---|---|---|---|
| **Input** | Wind Speed $[ms^{-1}]$ | 7 | 12 | 15 |
| **Output** | $H_{m0}$ $[m]$ | 0.471 | 0.812 | 0.984 |

Two tests taken from the methods described in [15] are applied to the emulator; firstly the individual error for each of the validation points is calculated and shown in Figure 4. It can be seen that all three points are within two standard deviations of the emulator mean. Secondly the Mahalanobis distance is calculated as 1.19, which is a reasonably good match to the reference value of 3. As both tests have been passed the emulator can be considered an acceptable statistical approximation of the simulator within the design space. To further improve the emulator, the validation points are incorporated into the design points and the emulator is re-constructed, shown in Figure 5.

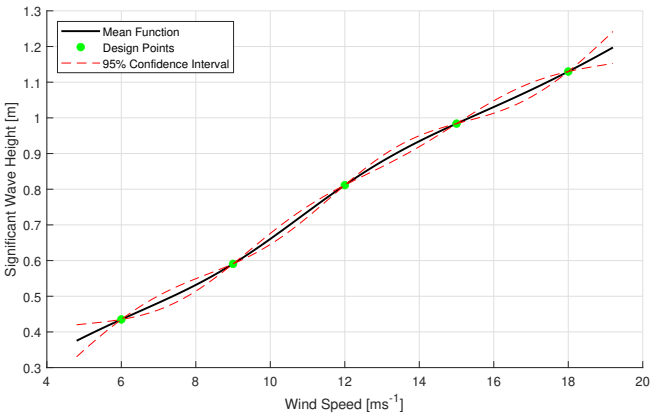


Figure 3. The first iteration of the 1D-emulator showing the posterior mean function and 95% confidence interval.

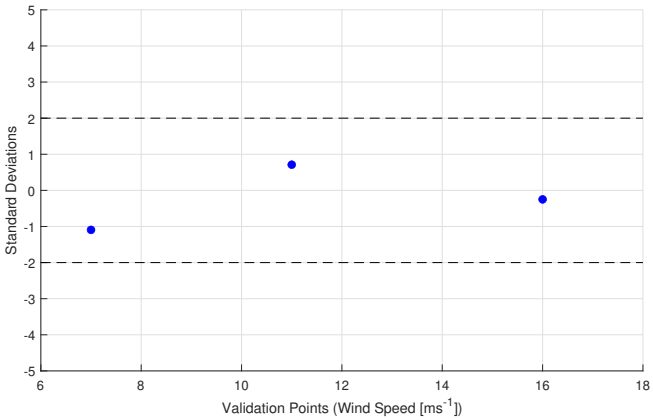


Figure 4. Standard error of the emulators prediction at the validation points.

## B. Example 2: Multiple input emulator

One advantage of GP emulators is that that can be easily scaled up to incorporate multi-dimensional

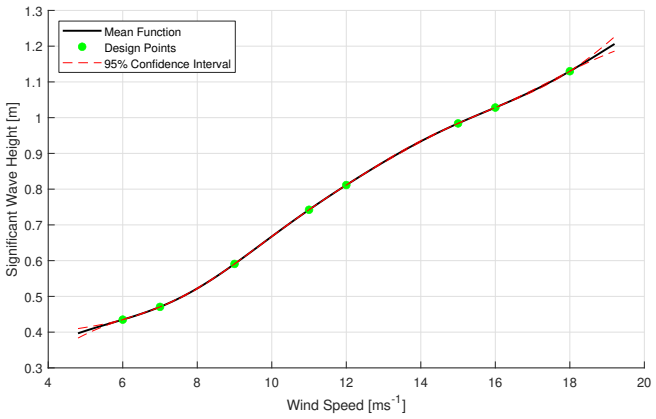


Figure 5. Final version of the validated 1-dimensional emulator.

inputs. In this example the SWAN model is now set-up to provide the significant wave height output as a condition of two independent inputs- wind speed and wave height on the boundary. The first problem which must be overcome is deciding the design space for the simulator. If, like with the single input example, the simulator were run even steps along in each dimension the number of runs required would rapidly become impossible, even just 10 runs in each dimension requires 100 simulator runs. By drawing from a a Latin Hypercube (LHC) it was possible to acquire a sample of design points that are well distributed throughout the input space. This was applied to generate 20 sets of initial conditions, shown in Figure 6. 12 points are used in the initial design, with the remaining 8 points used for validation listed in Table III.

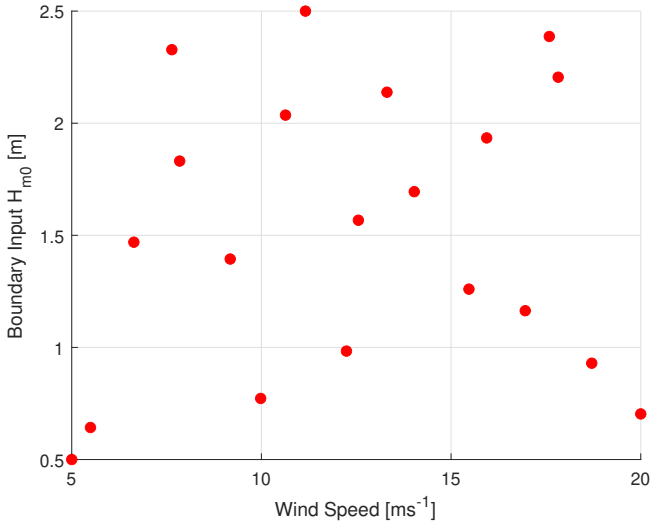


Figure 6. The 20 design points for the 2-dimensional input of the emulator, chosen using LHC sampling.

As with the 1-dimensional example a linear mean function $\mu$ Equation 7 and Gaussian form correlation function $\sigma^2 c(x, x')$ Equation 8 are selected, this time accounting for the 2-dimensional input, where $x_1$ and $x_2$ represent the two input variables. This form can be used for any number of independent inputs.

$$\mu = \beta_1 x_1 + \beta_2 x_2 + \beta_3 \tag{7}$$

Table III
DESIGN POINTS FOR 2D SIMULATOR

| | | | | | |
|---|---|---|---|---|---|
| **Emulator Design** | | | | | |
| **Input** | Wind Speed $[ms^{-1}]$ | 5.0 | 14.03 | 9.98 | 15.47 |
| | | 12.56 | 17.82 | 7.85 | 11.16 |
| | | 9.18 | 20.0 | 16.96 | 6.63 |
| **Input** | Boundary $H_{m0}$ $[m]$ | 0.5 | 1.69 | 0.77 | 1.26 |
| | | 1.57 | 2.21 | 1.83 | 2.5 |
| | | 1.39 | 0.73 | 1.16 | 1.47 |
| **Output** | $H_{m0}$ $[m]$ | 0.43 | 1.53 | 0.81 | 1.42 |
| | | 1.38 | 2.01 | 1.29 | 1.80 |
| | | 1.04 | 1.37 | 1.46 | 1.04 |
| **Validation** | | | | | |
| **Input** | Wind Speed $[ms^{-1}]$ | 12.24 | 7.64 | 15.94 | 17.59 |
| | | 5.49 | 13.31 | 10.64 | 18.71 |
| **Input** | Boundary $H_{m0}$ $[m]$ | 0.98 | 2.33 | 1.93 | 2.39 |
| | | 0.64 | 2.14 | 2.04 | 0.93 |
| **Output** | $H_{m0}$ $[m]$ | 1.08 | 1.59 | 1.74 | 2.08 |
| | | 0.52 | 1.72 | 1.52 | 1.43 |

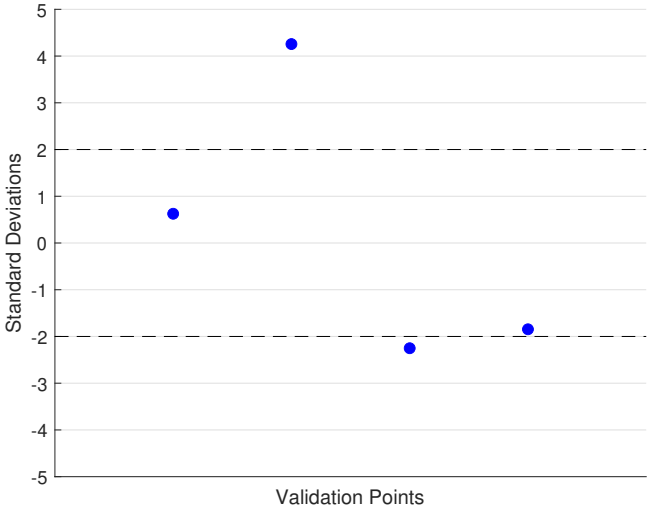


Figure 7. Standard error of predicted values from first iteration of 2D emulator at first four of validation points

$$c(x, x') = \exp\left\{-\sum_{i=1}^{2} 0.5\left[\frac{(x_i - x_i')}{\delta_i}\right]^2\right\} \tag{8}$$

The same procedure used in *Example 1* is followed to build the emulator. The emulator is also being constructed with a linear mean and with the assumption of a weak prior relationship between the hyper-parameters. To obtain the posterior distribution, the correlation lengths $\delta = [\delta_1, \delta_2]$ are fixed and estimated by maximising the likelihood function based on the design points. Then the estimated value is substituted in to give the other hyper-parameters.

To validate the emulator four of the simulator runs are allocated as validation points. The emulator is tested against these four points to determine the standard error and Mahalanobis distance. The standard error from the validation points is shown in Figure 7, where it can be seen that the error is outside 2 standard deviations for two of the four points. Additionally the Mahalanobis distance is 65.99 which is not a close fit to the theoretical mean of 4. The emulator therefore fails both validation tests.
The emulator can be improved by re-building it using

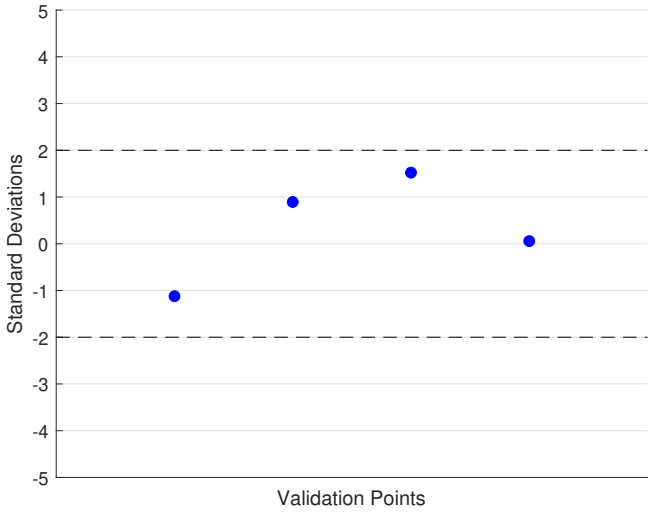


Figure 8. Standard error of predicted values from second iteration of 2D emulator at final four validation points.

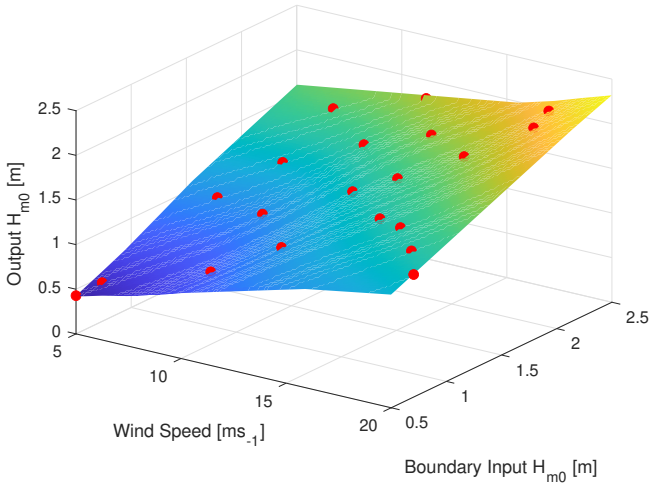


Figure 9. Plot of finalised emulator prediction showing design points (red).

extra design points. The 4 validation points are added to the 12 design points to create a new design space of 16 points. The emulator is reconstructed with the exact same method from this larger design. The new emulator is the subjected to the validation tests using the final 4 simulator runs. The standard error is shown in Figure 8 where it can be seen that all the validation points fall within 2 standard deviations. Furthermore the Mahalanobis distance is now 4.85, a close fit to the reference value of 4. The emulator is now declared a valid approximation of the simulator. Finally the emulator is further improved by re-building in once again including the final 4 validation points in the design. The emulator output is shown in Figure 9 showing the design points.

## IV. Uncertainty Quantification

The aim of uncertainty quantification is to assess the effect that uncertainty in the inputs has on the simulator output. Each of the model inputs are subject to uncertainties. Thus the inputs can be each be considered as a probability distribution rather than a fixed value. The (unknown) simulator inputs are denoted as $X$ and the joint uncertainty distribution is $\omega(x)$. The objective is to quantify the effect in the model output $f(X)$ arising from uncertainty in $X$ by calculating the mean $E[f(X)]$ and variance $Var[f(X)]$ with respect to $\omega(x)$. The traditional Monte Carlo method is to draw a large number of samples from $X$ and run the simulator to compute a corresponding number of outputs $f(X)$, from these the mean and variance can be computed. This requires many thousands (or even millions) of runs of a simulator which is not possible if the computational time takes more than even a few seconds. A computationally efficient emulator is able to provide output extremely quickly, as long as it is an effective approximation for the simulator then it can be used to provide the data for uncertainty quantification. The emulator mean and variance are denoted as $E^*$ and $Var^*$ so approximation of the simulator mean and variance with respect to $\omega(x)$ provided by the emulator are denoted $E^*[E[f(x)]]$ and $E^*[Var[f(x)]]$. Using an emulator rather than the simulator also inserts a new measure of uncertainty, related to how well the emulator approximates the simulator, $Var^*[E[f(x)]]$.

### A. Example: Multiple input emulator

Using the emulator constructed in *Example 2* a demonstration of the principles of uncertainty quantification is shown. For the purposes of the example a particular set of input values is chosen with mean values as follows:

- Wind Speed: $8.5ms^{-1}$
- Boundary Wave height: $1.9m$

If the expert knowledge about the inputs suggests that the wind and wave inputs are both normally distributed then the two inputs are described by Equation 9 and 10. 10,000 values drawn from the joint distribution are shown in Figure 10.

$$X_1 \sim \mathcal{N}(8.5, 0.75) \tag{9}$$

$$X_2 \sim \mathcal{N}(1.9, 0.1) \tag{10}$$

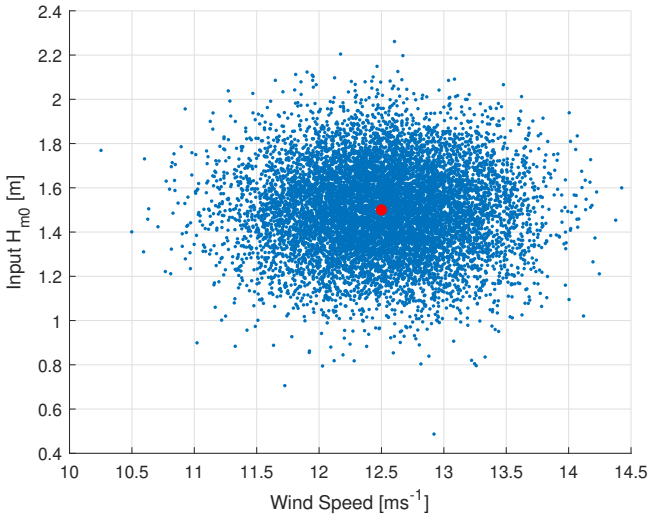


Figure 10. 10,000 sample input points and mean value(red).

The results from the emulator uncertainty calculations are given in Table IV.

Table IV
EMULATOR UNCERTAINTY OUTPUT

| | |
|---|---|
| $E^*[E[f(x)]]$ | 1.377 |
| $E^*[Var[f(x)]]$ | 0.021 |
| $Var^*[E[f(x)]]$ | 0.000089 |

These results say that the mean output of the simulator due to uncertainties about the inputs is a $H_{m0}$ value of 1.377 $m$. The variance in the output due to the uncertainty in the inputs is 0.021. Figure 11 shows a histogram comprised of 10,000 samples drawn from the Gaussian distribution given by Equation 11. The uncertainty about the estimate of $E[f(x)]$ given by $Var^*[E[f(x)]]$ is very small, suggesting that the emulator is a very good fit to the toy model simulator.

$$\mathcal{N}(E^*[E[f(x)]], E^*[Var[f(x)]]) \tag{11}$$

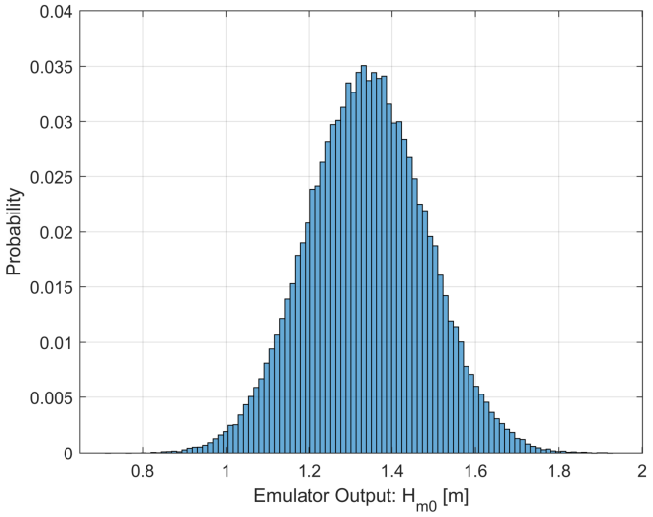


Figure 11. Histogram of emulator estimation showing probability of range of values.

## V. CONCLUSIONS

The method described in this study is a powerful tool that can be applied to a wide number of computational simulations. While the design of the emulator may be more complicated and greater care needs to be taken in selecting the mean and covariance functions the methods demonstrated here can be scaled up and utilized across many applications. Understanding uncertainty in simulations is an important and often overlooked facet of modelling and methods such as these can be employed to the benefit of the industry.

## VI. ACKNOWLEDGMENTS

This work was funded as part of the Tidal Stream Industry Energiser Project (TIGER), a European Union INTERREG V A France (Channel) England Research and Innovation Programme, which is co-financed by the European Regional Development Fund (ERDF).

## REFERENCES

[1] J. C. C. van Nieuwkoop, H. C. M. Smith, G. H. Smith, and L. Johanning, "Wave resource assessment along the Cornish coast (UK) from a 23-year hindcast dataset validated against buoy measurements," *Renewable Energy*, vol. 58, pp. 1–14, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0960148113001420http://ac.els-cdn.com/S0960148113001420/1-s2.0-S0960148113001420-main.pdf?_tid=acad56f2-580d-11e6-9065-00000aab0f27&acdnat=1470072823_624f3db507add1ad41f9ac12c9f4f39dhttp://dx.doi.org/10.

[2] H. C. Smith, D. Haverson, G. H. Smith, C. S. Cornish, and D. Baldock, "Assessment of Wave and Current Resource at the Wave Hub Site," Tech. Rep., 2011. [Online]. Available: https://www.wavehub.co.uk/downloads/Resource_Info/Assessement_wave_and_current_resource_University_of_Exeter_-_Marine_Energy_Matters_2011.pdf

[3] H. C. M. Smith, D. Haverson, and G. H. Smith, "A wave energy resource assessment case study: Review, analysis and lessons learnt," *Renewable Energy*, vol. 60, pp. 510–521, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0960148113002620http://ac.els-cdn.com/S0960148113002620/1-s2.0-S0960148113002620-main.pdf?_tid=8e064222-580d-11e6-ac9d-00000aacb35d&acdnat=1470072772_249c8b830cdd0b4b467391bb3b17fa44

[4] R. C. Ris, L. H. Holthuijsen, and N. Booij, "A third-generation wave model for coastal regions - 2. Verification," *Journal of Geophysical Research-Oceans*, vol. 104, no. C4, pp. 7667–7681, 1999. [Online]. Available: http://onlinelibrary.wiley.com/store/10.1029/1998JC900123/asset/jgrc7637.pdf?v=1&t=ird777m6&s=28701ac3632b64e8c83dadd1161d2ac94a7a8083

[5] S. Ambühl, J. P. Kofoed, and J. D. Sørensen, "Quantification of wave model uncertainties used for probabilistic reliability assessments of wave energy converters," *Journal of Ocean and Wind Energy*, vol. 2, no. 2, pp. 98–106, 2015.

[6] J. Rougier, S. Guillas, A. Maute, and A. D. Richmond, "Expert knowledge and multivariate emulation: The thermosphere-ionosphere electrodynamics general circulation model (TIE-GCM)," *Technometrics*, vol. 51, no. 4, pp. 414–424, 11 2009.

[7] M. Brito, G. Griffiths, J. Ferguson, D. Hopkin, R. Mills, R. Pederson, and E. Macneil, "A behavioral probabilistic risk assessment framework for managing autonomous underwater vehicle deployments," *Journal of Atmospheric and Oceanic Technology*, vol. 29, no. 11, pp. 1689–1703, 11 2012.

[8] A. M. Clayton, A. C. Lorenc, and D. M. Barker, "Operational implementation of a hybrid ensemble/4D-Var global data assimilation system at the Met Office," *Quarterly Journal of the Royal Meteorological Society*, vol. 139, no. 675, pp. 1445–1461, 7 2013.

[9] A. Nikishova, A. Kalyuzhnaya, A. Boukhanovsky, and A. Hoekstra, "Uncertainty quantification and sensitivity analysis applied to the wind wave model SWAN," *Environmental Modelling and Software*, vol. 95, pp. 344–357, 2017.

[10] A. O'Hagan, "Bayesian analysis of computer code outputs: A tutorial," *Reliability Engineering and System Safety*, vol. 91, no. 10-11, pp. 1290–1300, 2006. [Online]. Available: http://ac.els-cdn.com/S0951832005002383/1-s2.0-S0951832005002383-main.pdf?_tid=8b5dd62a-580d-11e6-94b3-00000aacb35f&acdnat=1470072767_75e4e4f674c31dc5298d0a8bde72e819

[11] M. C. Kennedy and A. O'Hagan, "Bayesian calibration of computer models," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 63, no. 3, pp. 425–464, 2001. [Online]. Available: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/1467-9868.00294

[12] S. Conti and A. O'Hagan, "Bayesian emulation of complex multi-output and dynamic computer models," *Journal of statistical planning and inference*, vol. 140, no. 3, pp. 640–651, 2010.

[13] J. E. Oakley and A. O'Hagan, "Probabilistic sensitivity analysis of complex models: a Bayesian approach," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 66, no. 3, pp. 751–769, 2004. [Online]. Available: http://dx.doi.org/10.1111/j.1467-9868.2004.05304.xhttp://onlinelibrary.wiley.com/store/10.1111/j.1467-9868.2004.05304.x/asset/j.1467-9868.2004.05304.x.pdf?v=1&t=ighvw03e&s=3158df30f49ffad4cd4315b5e9d525b251466c57

[14] S. Conti, J. P. Gosling, J. E. Oakley, and A. O'Hagan, "Gaussian process emulation of dynamic computer codes," *Biometrika*, vol. 96, no. 3, pp. 663–676, 4 2009. [Online]. Available: http://www.jstor.org/stable/27798855

[15] L. S. Bastos and A. O'hagan, "Diagnostics for gaussian process emulators," *Technometrics*, vol. 51, no. 4, pp. 425–438, 11 2009.