

Multilevel multifidelity Monte Carlo methods for assessing uncertainty in coastal flooding

Note from the Authors

In this document, we have included the relevant revised sections of the manuscript in framed boxes. Note all references to figures, tables and equations in these boxes, unless otherwise stated, are to the figures and equations in the new manuscript and not to those in this response document.

Reviewer 1 comments

We would like to thank Reviewer 1 for their helpful and thoughtful comments. Below we outline how we have addressed each of their concerns and suggestions.

1. The MLMF method introduced by the authors seems to promise a nice way to generate good accuracy within a reasonable runtime for probabilistic hazards. This could be very useful for the nat haz community. However, as a traditionally educated hydrodynamic modeler and coastal engineer, it was very difficult for me to understand the details of the method presented. It would be helpful if the explanation could be extended or revised for understanding by a wider array of possible users. This could include something like "for each SFINCS run we calculate the mean and variance in water levels", "for each XBeach run we calculate mean and variance of water levels", "we do something with each of these means and variances to come up with a total mean and variance". With the explanation given so far, I have difficulty understanding how data from the various levels are used, how the various "samples" are used (there should be a better term for "samples"), and even how the two different models' results are used together. In general, the method should be accessible to hydrodynamic modelers to use, and understandable by such.

We will extensively re-write the methodology section in an attempt to make it more accessible to a broad range of coastal engineers and hydrodynamic modellers. As an initial step, we will move the description of the generic methodology for multifidelity estimators and the MLMC method to an appendix and focus solely on the combination MLMF in the main body of the text. We will also improve the introduction to the section with the following prose description of MLMF:

As discussed in Section 1, Monte-Carlo type methods can be used to assess uncertainty by estimating the expectations of functions of an input random variable. In our model scenario, the input random variable is some source of uncertainty, such as the friction coefficient, and the function involves running our numerical model and computing values such as the water elevation height at specific locations, from the model output. These estimates could be calculated using the standard Monte Carlo approach, but this is computationally expensive due to the need to run large numbers of model simulations to obtain an appropriate accuracy (see Eq. 1 and the discussion below it). The computational cost of running the model can be reduced by either coarsening the grid resolution or using a less complex model, or, in the case of this work, making use of both approaches by using the multilevel multifidelity Monte Carlo method (MLMF). Using a coarse grid and/or simpler model gives an estimate which is cheap to compute but (more) incorrect and thus has an error. This error can be corrected by estimating the difference between the low and high fidelity models and/or the different resolutions, and adding these on to the cheaply computed expectation. Key to the approach is the observation that estimating the difference requires fewer simulations than computing the full estimate, because the variance of the correction is (hopefully) smaller than the variance of the outputs. For the different grid resolutions, the correction is done by the telescoping sum of the multilevel Monte Carlo method (MLMC), while for the different fidelity models, the correction is done by control variate formulae. The challenge is composing these approaches so that we can do both, which is what MLMF seeks to do.

We will also introduce a hypothetical scenario to better illustrate MLMF in practical terms and use it to extensively describe certain core concepts such as estimators. Following the reviewer's suggestion, we will also change 'samples' to 'simulations'. The proposed new Section 2 is included as an addendum to this document.

2. Fig 1. I don't understand why the arrow at the left says "Level 1". Also, the term "number of samples" is confusing. Could be easier to understand as "number of runs" or "number of scenarios executed".

We acknowledge that the arrow with the levels is confusing. It was meant to indicate the mesh getting finer but the figure actually is clearer and simpler without the arrow as you suggest so we will remove it. Also following your suggestion we will change number of samples to number of simulations. We have included below the modified figure:

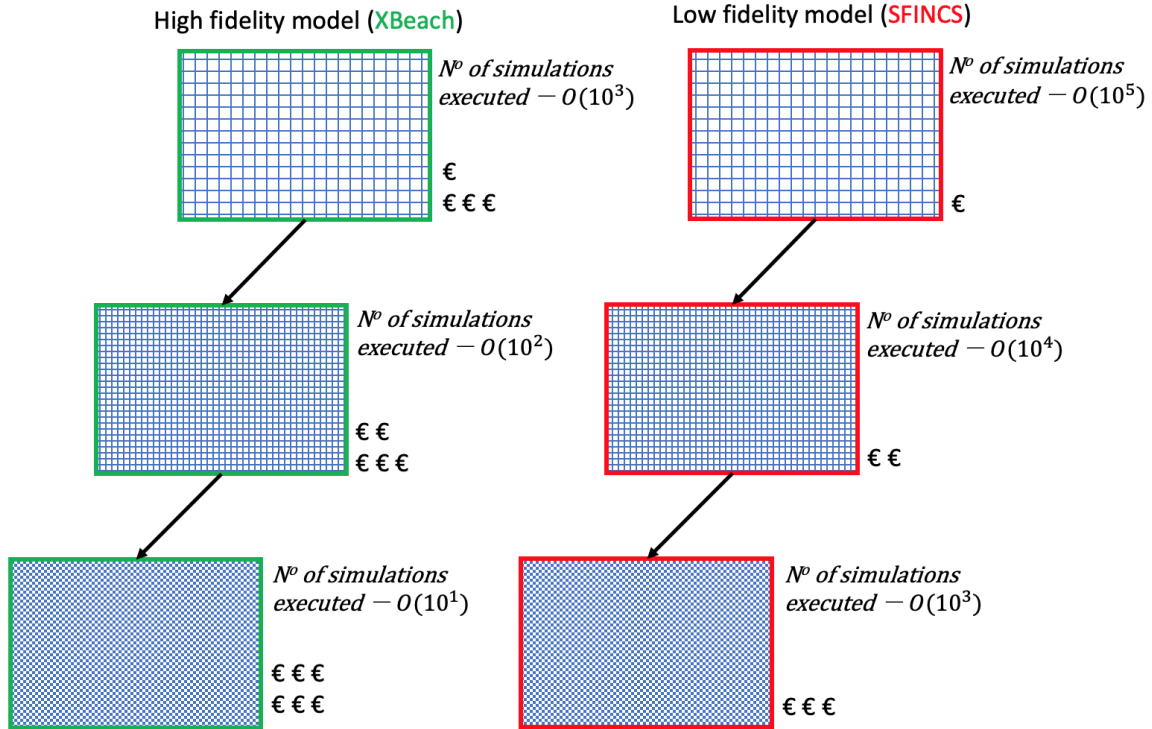


Figure 1: Example illustration of how MLMF's multifidelity multilevel approach using SFINCS and XBeach models on different grid resolutions results in computational cost savings. Note the € symbol indicates the order of magnitude of the computational cost for a single simulation with this model at this grid resolution i.e. €€ indicates $O(10^2)$ seconds for a single simulation. The orders of time and number of scenarios are approximately those for the Myrtle Beach test case in Section 3.3.

3. Line 163. What does := mean?

:= means equal by definition, or is defined to be, but we will make this line simpler by changing it to the following

set L equal to $L + 1$

4. Please explain clearly the difference between Eq. 3 and Eq. 4.

The first equation is the generic formula and the second equation is the actual expression of the estimator. We acknowledge this is confusing and therefore will completely remove the first equation (Eq 2.) and keep only the second equation (Eq 3.). Note, we have answered this query assuming you are referring to Eq 2 and 3, rather than 3 and 4.

5. It is difficult to tell whether sections 2.1, 2.2, and 2.3 are all just re-explaining what Geraci et al. 2015 showed, or whether this is new material. Please clarify this in the text.

Section 2.1 is indeed largely re-explaining what Geraci et al. 2015 showed; we will add the following text at the beginning of this sub-section:

In this section, we describe the theory for the standard MLMF approach, following Geraci et al. (2015) throughout.

Section 2.2 is a novel technique which we have developed as a part of this work and therefore we will add the following sentence at the beginning of this section:

To resolve this, in this work we develop our own novel technique to find the cumulative distribution function (CDF) from the MLMF outputs, using a modified version of the inverse transform sampling method from [Gregory and Cotter \(2017\)](#).

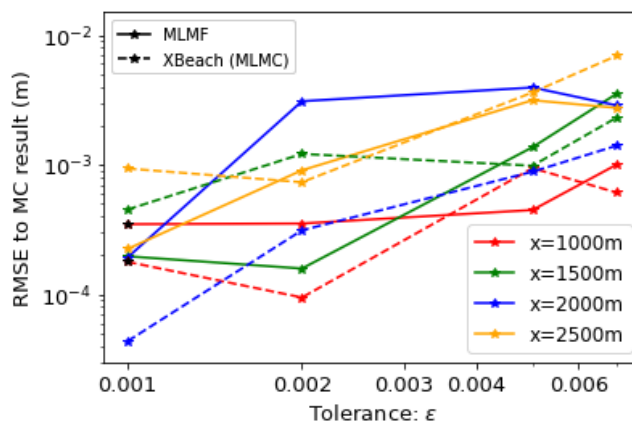
Section 2.3 is also new material and therefore we will add the following sentence at the beginning of this section:

In this work, we construct our own Python MLMF wrapper around both SFINCS and XBeach to implement the MLMF algorithm.

6. Why doesn't RMSE have a unit? Does RMSE mean relative RMSE, normalized to be unitless? This should be stated clearly.

We will add the units for the RMSE to the figures (see below for example which is a recreation of the updated figure 7). We will also add the following line to the error norm tables (2 and 4) to emphasise that these errors are unitless.

Note the error norms are unitless because the CDFs are unitless.



References

- Geraci, G., Eldred, M., Iaccarino, G., 2015. A multifidelity control variate approach for the multilevel monte carlo technique. Center for Turbulence Research Annual Research Briefs , 169–181.
- Gregory, A., Cotter, C.J., 2017. On the calibration of multilevel Monte Carlo ensemble forecasts. Quarterly Journal of the Royal Meteorological Society 143, 1929–1935. doi:[10.1002/qj.3052](https://doi.org/10.1002/qj.3052).

2 Methodology: Applying the multilevel multifidelity Monte Carlo method (MLMF) to assess uncertainty in coastal flooding

As discussed in Section 1, Monte-Carlo type methods can be used to assess uncertainty by estimating the expectations of functions of an input random variable. In our model scenario, the input random variable is some source of uncertainty, such as the friction coefficient, and the function involves running our numerical model and computing values such as the water elevation height at specific locations, from the model output. These estimates could be calculated using the standard Monte Carlo approach, but this is computationally expensive due to the need to run large numbers of model simulations to obtain an appropriate accuracy (see Eq. 1 and the discussion below it). The computational cost of running the model can be reduced by either coarsening the grid resolution or using a less complex model, or, in the case of this work, making use of both approaches by using the multilevel multifidelity Monte Carlo method (MLMF).

Using a coarse grid and/or simpler model gives an estimate which is cheap to compute but (more) incorrect and thus has an error. This error can be corrected by estimating the difference between the low and high fidelity models and/or the different resolutions, and adding these on to the cheaply computed expectation. Key to the approach is the observation that estimating the difference requires fewer simulations than computing the full estimate, because the variance of the correction is (hopefully) smaller than the variance of the outputs. For the different grid resolutions, the correction is done by the telescoping sum of the multilevel Monte Carlo method (MLMC), while for the different fidelity models, the correction is done by control variate formulae. The challenge is composing these approaches so that we can do both, which is what MLMF seeks to do.

The theory for MLMF is the focus of Section 2.1, whilst details on the control variate multifidelity approaches and MLMC can be found in Appendix A and B respectively. As described in Geraci et al. (2015), the standard MLMF approach cannot estimate the probability of an output variable exceeding a certain value. The latter is often also of significant interest for flooding problems and thus in Section 2.2, we present novel theory to extend MLMF for the estimation of probabilities. The implementation and application of the MLMF method in this work is then described in Section 2.3 and we conclude this methodology section with a brief remark on different methods to assess uncertainty in Section 4.

2.1 Multilevel multifidelity Monte Carlo method (MLMF)

MLMF seeks to improve the efficiency of uncertainty analyses by running fewer simulations at the more expensive finer resolutions than at the cheaper coarser resolutions and by running fewer high fidelity model simulations than low fidelity ones (see Figure 1). In this section, we describe the theory for the standard MLMF approach, following Geraci et al. (2015) throughout. A pictorial representation of this algorithm is shown in Figure 2 and a full statement of the algorithm is included at the end of the section.

To fix ideas, we consider a hypothetical scenario, where the variable of interest is the water elevation height at a given location after a given time and the uncertain parameter is the friction coefficient which we assume follows a normal distribution. The desired grid resolution in our model is $\Delta x = 5000/2^{10} (\approx 5)$ m. Note that this hypothetical scenario is similar to the example

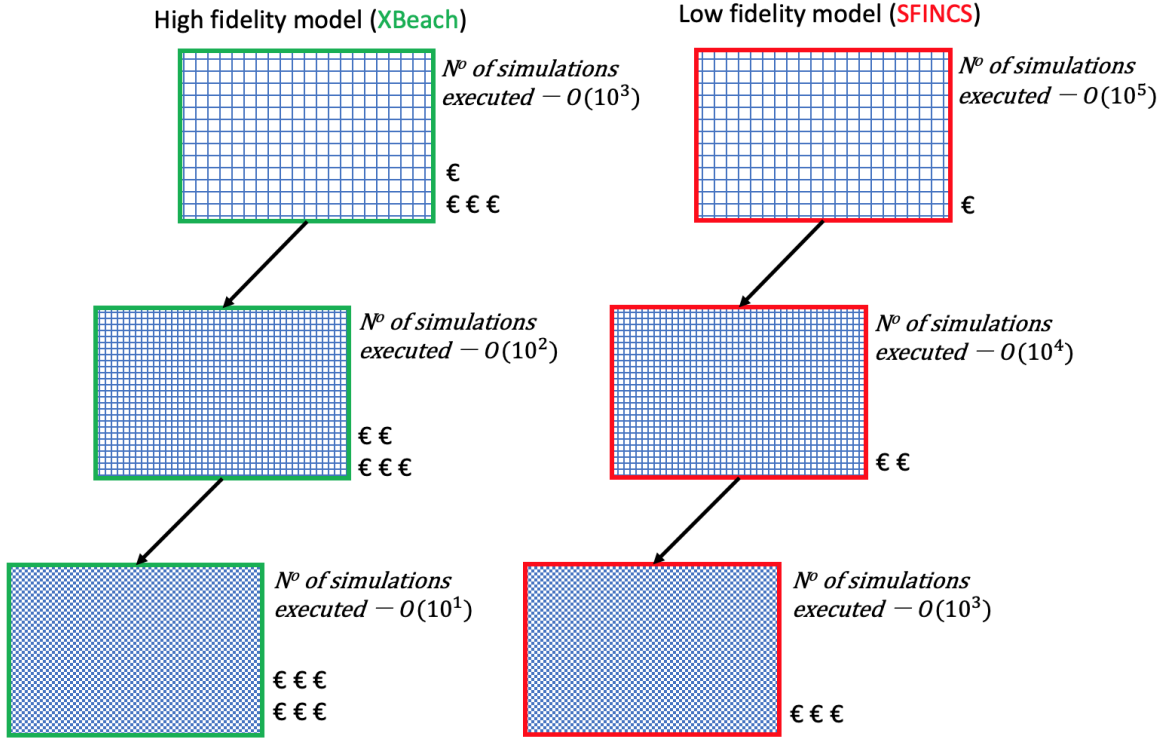


Figure 1. Example illustration of how MLMF’s multifidelity multilevel approach using SFINCS and XBeach models on different grid resolutions results in computational cost savings. Note the € symbol indicates the order of magnitude of the computational cost for a single simulation with this model at this grid resolution *i.e.* €€ indicates $O(10^2)$ seconds for a single simulation. The orders of time and number of scenarios are approximately those for the Myrtle Beach test case in Section 3.3.

used as the first test case in this work. Moreover, throughout this work, we use HF and LF to denote the high fidelity XBeach model and low fidelity SFINCS model respectively.

We denote the MLMF estimator for the water depth at the finest grid resolution L as $\hat{Q}_{M_L}^{HF,CV}$. Here the finest grid resolution is the grid resolution we would like to evaluate our model at; for our hypothetical scenario the finest grid resolution is $\Delta x = M/2^L = 5000/2^{10} (\approx 5)$ m. Note that following standard notation, $\hat{\cdot}$ denotes that $\hat{Q}_{M_L}^{HF,CV}$ is an estimator. An estimator represents the rule for calculating an estimate of a variable of interest given data. In our hypothetical scenario, the estimator is the rule, the variable of interest is the water elevation height, the data is our model runs and the estimate is then the numerical approximation of the mean water elevation that we obtain using our model runs. For MLMF, the rule for the estimator is a combination of the multilevel MLMC estimator (B3) with the multifidelity control variate (A1). The multilevel part of the

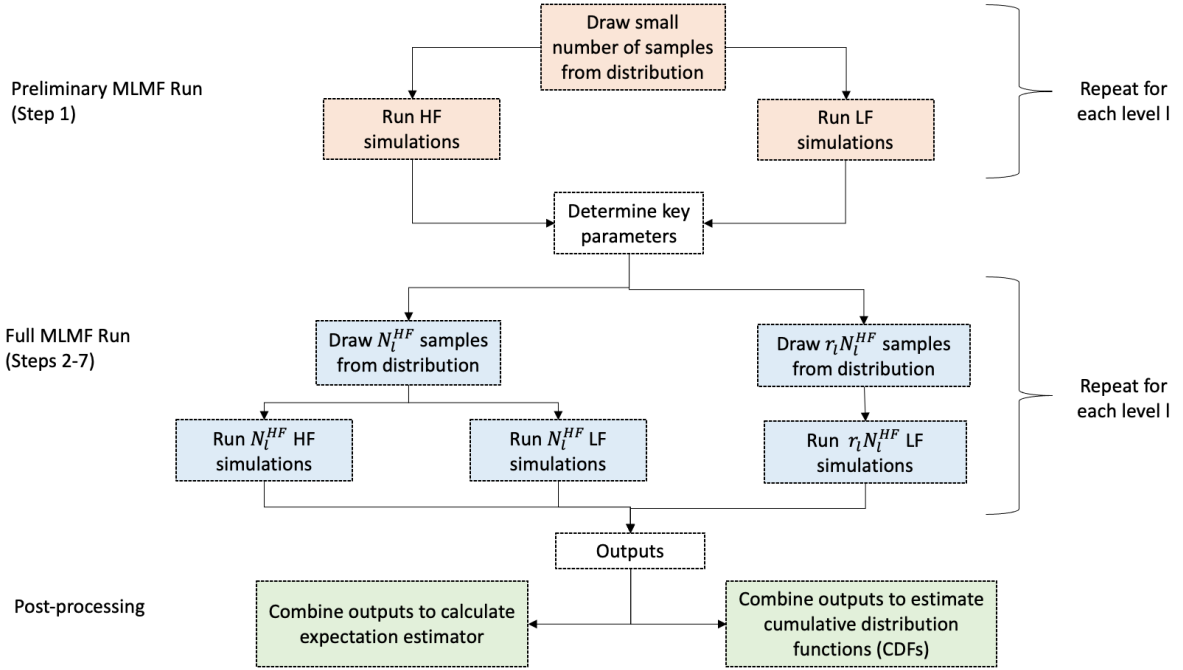


Figure 2. Flow chart of multi-model approach to MLMF using HF (XBeach) and LF (SFINCS).

125 estimator uses linearity of expectations (see (B1)) to construct the following telescoping sum

$$\hat{Q}_{M_L}^{HF,CV} = \hat{Q}_{M_{l_\mu}}^{HF,CV} + \sum_{l=l_\mu+1}^L \left[\hat{Q}_{M_l}^{HF,CV} - \hat{Q}_{M_{l-1}}^{HF,CV} \right], \quad (2)$$

where M_l denotes different resolutions at which the estimator is evaluated, with l_μ being the coarsest resolution. In our hypothetical scenario, the estimator is evaluated at resolutions of $[5000/2^4, 5000/2^5, 5000/2^6, 5000/2^7, 5000/2^8, 5000/2^9, 5000/2^{10}]$ m. Eq. (2) finds the multilevel multifidelity estimate of water elevation at the finest resolution by calculating the multifidelity estimate at the coarsest resolution ($5000/2^4$), adding to this the difference between the multifidelity estimates at the coarsest resolution ($5000/2^4$) and the slightly finer resolution ($5000/2^5$) etc., up to and including the second finest and finest resolutions pair of $5000/2^9$ and $5000/2^{10}$. By the linearity of expectations, the sum of these differences is an estimate for the expected value of the water elevation on the finest resolution that is as accurate as simply calculating a standard Monte Carlo estimate on the finest resolution. The advantage is that calculating the estimate using this approach is less computationally expensive than using the standard Monte Carlo approach because the width of the distribution of the model outputs at each resolution X_l is much larger than the width of the distribution of the *difference* between the outputs ($X_l - X_{l-1}$). Figure 3 illustrates this for two resolutions of the hypothetical scenario computed using XBeach. The narrower the distribution (*i.e.* the smaller the variance) the fewer samples are needed to estimate its mean (see Figure 9 for example). Note that the distribution of

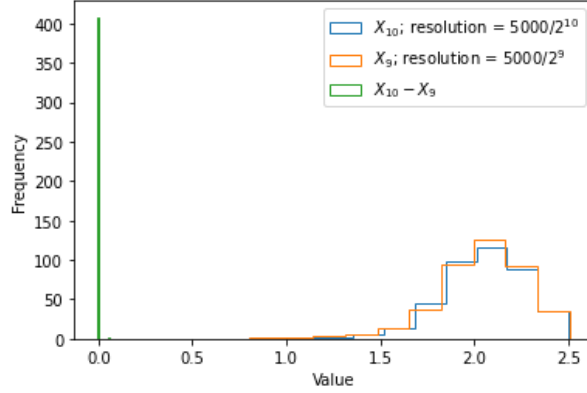


Figure 3. Distribution of outputs generated by XBeach for the hypothetical scenario at the finest resolution and at the second finest resolution considered, as well as the distribution of the difference between these output values. Note that the distribution for the difference between the output values is much narrower, meaning fewer samples are required to get a good estimate of the mean. For reference, in this hypothetical scenario the variance at resolution $\Delta x = 5000/2^9$ m is 0.0587; the variance at resolution $\Delta x = 5000/2^{10}$ m is 0.0580; and the variance of the difference between outputs at $[5000/2^9, 5000/2^{10}]$ is $9.82e-06$.

the difference is very narrow in this example; for more complex cases it may be wider, although it should still remain narrower than the distribution of the individual outputs.

Combining multilevel estimators with the multifidelity control variate (A1), the full rule for the MLMF estimator is

$$\hat{Q}_{M_L}^{HF,CV} = \sum_{l=l_\mu}^L \left(\hat{Y}_{M_l}^{HF} + \alpha_l \left(\hat{Y}_{M_l}^{LF} - \hat{E} [Y_{M_l}^{LF}] \right) \right), \quad (3)$$

where

$$\hat{Y}_{M_l}^* = \begin{cases} N_{l_\mu}^{-1} \sum_{i=1}^{N_{l_\mu}} X_{l_\mu}^{(i)} & l = l_\mu, \\ N_l^{-1} \sum_{i=1}^{N_l} \left(X_l^{(i)} - X_{l-1}^{(i)} \right) & l > l_\mu, \end{cases} \quad (4)$$

where the superscript * here can indicate results from either XBeach (HF) or SFINCS (LF). In our hypothetical scenario, X_l is the water elevation height from the model run using a grid resolution of $\Delta x = 5000/2^l$ m, with X_{l-1} being the same but for a grid resolution of $\Delta x = 5000/2^{l-1}$ m. For each difference pair, (i) denotes that the value sampled from the normal distribution for the uncertain friction coefficient is the same for both the finer resolution X_l^i simulation and the coarser resolution X_{l-1}^{i-1} simulation. Thus, $\hat{Y}_{M_l}^*$ is the mean of the difference between two model runs conducted at different resolutions with the same random number (*i.e.* value sampled from the distribution) used for friction for each pair. Moreover, for each (i) , the same random number is used for the XBeach model pair and the SFINCS model pair, *i.e.* the same random numbers are used to construct both $\hat{Y}_{M_l}^{HF}$ and $\hat{Y}_{M_l}^{LF}$. Note constructing estimators like this means that the coarsest level, l_μ is left without a pair and therefore $\hat{Y}_{M_{l_\mu}}^*$ is just the mean of the model runs conducted at the coarsest resolution. Note further that, although not strictly

necessary, here we choose to run both SFINCS and XBeach at the same resolutions, as it seems sensible to assume that this
 155 will maximise correlation between the outputs at each level.

The other terms in (3) come from the multifidelity estimator. The notation $\hat{E}[\cdot]$ denotes the estimator for the expected value
 – statistically speaking we cannot know the actual expected value (\mathbb{E}) of $Y_{M_l}^{LF}$ because this would require knowing the exact
 distribution of $Y_{M_l}^{LF}$. Thus, the best we can do is calculate an estimate of the expected value using data from SFINCS runs at
 different resolutions, *i.e.* use an estimator. This subtlety is discussed in more detail in Appendix A. Finally α_l is a coefficient
 160 which weights the SFINCS model outputs and is defined as

$$\alpha_l = -\rho_l \sqrt{\frac{\text{Var}(\hat{Y}_{M_l}^{HF})}{\text{Var}(\hat{Y}_{M_l}^{LF})}}, \quad (5)$$

where ρ_l is the Pearson's correlation coefficient. In our hypothetical scenario, ρ_l is the correlation between the water elevation
 calculated by XBeach and that calculated by SFINCS at each resolution l . We refer the reader to Appendix A for more details
 on multifidelity estimators.

165 To make calculating the variance of the water depth estimator simpler, we follow standard practice throughout and indepen-
 dently sample the values for the friction coefficient for each \hat{Y}_{M_l} . Hence the variance of the MLMF estimator is

$$\text{Var} \left[\hat{Q}_{M_L}^{HF,CV} \right] = \sum_{l=l_\mu}^L (N_l^{HF})^{-1} \text{Var} \left[\hat{Y}_l^{HF} \right] \left(1 - \frac{r_l}{1+r_l} \rho_l^2 \right), \quad (6)$$

using independence. Here N_l^{HF} is the number of XBeach (HF) simulations required at level l to compute $\hat{Y}_{M_l}^{HF}$ (which is
 also the number of SFINCS (LF) simulations required to compute $\hat{Y}_{M_l}^{LF}$), and r_l is the factor of extra SFINCS simulations
 170 required to compute $\hat{E} [Y_{M_l}^{LF}]$. Note that, throughout this work and for simplicity, we refer to N_l^{HF} as the number of XBeach
 simulations required, because the total number of SFINCS simulations required is the combined quantity $(1+r_l)N_l^{HF}$, and
 not just N_l^{HF} .

Because $\rho_l^2 < 1$ by definition of a correlation coefficient, equation (6) shows that the greater the correlation between the
 two models, the greater the reduction in the variance of the estimator. We thus seek to maximise this correlation. Geraci et al.
 175 (2017) show that, because the multifidelity control variate is unbiased, correlation can be artificially increased by modifying
 the estimator \hat{Y}_l^{LF} using

$$\hat{Y}_l^{LF} = \gamma_l \hat{X}_l^{LF} - \hat{X}_{l-1}^{LF}, \quad (7)$$

where the modification factor γ_l adds an extra degree of freedom to maximise the correlation. Therefore, instead of (3), we use

$$180 \quad \hat{Q}_{M_L}^{HF,CV} = \sum_{l=l_\mu}^L \left(\hat{Y}_{M_l}^{HF} + \alpha_l \left(\hat{Y}_l^{LF} - \hat{E} \left[\hat{Y}_l^{LF} \right] \right) \right), \quad (8)$$

and the new correlation coefficient $\hat{\rho}_l^2$ is dependent on γ_l and is equal to

$$\hat{\rho}_l^2 = \rho_l^2 \frac{\text{Cov}^2 \left(\hat{Y}_l^{HF}, \hat{Y}_l^{LF} \right) \text{Var} \left[\hat{Y}_l^{LF} \right]}{\text{Cov}^2 \left(\hat{Y}_l^{HF}, \hat{Y}_l^{LF} \right) \text{Var} \left[\hat{Y}_l^{LF} \right]}, \quad (9)$$

where we correct a typographical error in the formula given in Geraci et al. (2017). By differentiating (9) with respect to γ_l , we find the correlation is maximised when

$$185 \quad \gamma_l = \frac{\text{Cov}(\hat{Y}_l^{HF}, X_{l-1}^{LF}) \text{Cov}(X_l^{LF}, X_{l-1}^{LF}) - \text{Var}[X_{l-1}^{LF}] \text{Cov}(\hat{Y}_l^{HF}, X_l^{LF})}{\text{Var}[X_l^{LF}] \text{Cov}(\hat{Y}_l^{HF}, X_{l-1}^{LF}) - \text{Cov}(\hat{Y}_l^{HF}, X_l^{LF}) \text{Cov}(X_l^{LF}, X_{l-1}^{LF})}. \quad (10)$$

Note that when using the modified estimator (7) the formulae previously stated in this section remain the same but \hat{Y}_l^{LF} and ρ_l are replaced with \hat{Y}_l^{LF} and $\hat{\rho}_l$, respectively, in all formulae.

Finally, using (A3) and (B6), the overall cost of the MLMF algorithm (*i.e.* finding the water elevation at grid resolution L) is

$$190 \quad C = \sum_{l=l_\mu}^L N_l^{HF} (C_l^{HF} + C_l^{LF} (1 + r_l)). \quad (11)$$

In order to obtain the optimum values for N_l^{HF} and r_l in (6), we minimise this cost with respect to the variance constraint

$$\text{Var}[\hat{Q}_{M_L}^{HF,CV}] < \epsilon^2/2. \quad (12)$$

which results in the following optimum formula for the factor of extra SFINCS simulations

$$r_l = -1 + \sqrt{\frac{\hat{\rho}_l^2}{1 - \hat{\rho}_l^2} \omega_l}, \quad (13)$$

195 where $\omega_l = C_l^{HF}/C_l^{LF}$ is the ratio of the cost of running XBeach and SFINCS, and the following optimum formula for the number of XBeach simulations

$$N_l^{HF} = \frac{2}{\epsilon^2} \left[\sum_{k=l_\mu}^L \left(\frac{\text{Var}[\hat{Y}_k^{HF}] C_k^{HF}}{1 - \hat{\rho}_l^2} \right)^{1/2} \Lambda_k(r_k) \right] \sqrt{(1 - \hat{\rho}_l^2) \frac{\text{Var}[\hat{Y}_l^{HF}]}{C_l^{HF}}}, \quad (14)$$

where

$$\Lambda_k(r_k) = 1 - \frac{r_k}{1 + r_k} \hat{\rho}_k^2, \quad (15)$$

200 and as in (B7), ϵ should be viewed as a user-defined accuracy tolerance.

Calculating (14) requires estimates of the variance and cost. Therefore we run 50 initial simulations for each model at each resolution (see Step 1 of Algorithm 1) and use the kurtosis to check whether this provides a good enough estimate of the variance. Following Giles (2008), if the kurtosis is less than 100, then we consider our estimate of the variance to be good enough. Note further that if we are interested in the value of the variable of interest at multiple locations, N_l^{HF} must
 205 be calculated separately for each location. In the algorithm, we run $\max N_l$ over all locations and then when calculating the estimator (3) at each location, subsample the optimum number for that specific location from the full output.

2.1.1 MLMF algorithm

Given the theory outlined above, the MLMF algorithm used in this study is summarised in Algorithm 1.

Algorithm 1 Multilevel Multifidelity Monte Carlo method.

- 1: Estimate the variance and cost of the MLMF estimator, as well as the correlation and cost ratio between the HF and LF models at user-specified levels using an initial estimate for the number of simulations. The same set of random numbers must be used for the HF and LF models
 - 2: Start with $L = l_\mu$
 - 3: Define optimal N_l^{HF} using (14) and r_l using (13) with increased correlation factor (9) when required
 - 4: If the optimal N_l^{HF} is greater than the number of simulations of the HF and LF models from Step 1, evaluate the extra simulations required
 - 5: If the optimal $r_l N_l^{HF}$ is greater than the number of simulations of the LF model after Step 4, evaluate the extra simulations of LF required
 - 6: If the algorithm has not converged and $L < L_{\max}$, set L equal to $L + 1$ and return to Step 3
 - 7: If algorithm converged, or $L \geq L_{\max}$, STOP
-

2.2 Cumulative distribution functions

210 In this section so far, we have described the standard MLMF framework outlined in Geraci et al. (2015), the objective of which is to find the expectation of the output variable of interest. However, the probability of a variable exceeding a certain value is often of significant value in the study of natural hazards. This probability is complicated to estimate because MLMF computes very few values at the finest resolution from which we could build the distribution.

To resolve this, in this work we develop our own novel technique to find the cumulative distribution function (CDF) from 215 the MLMF outputs, using a modified version of the inverse transform sampling method from Gregory and Cotter (2017). The output of a cumulative distribution function, $\mathbb{P}(X \leq x)$, is some value between 0 and 1. Returning to the hypothetical example used throughout this section, X is the water elevation as a variable and x is its value. For evaluating uncertainty, we would like to know the value of x which the water elevation at a given location after a given time is below for 25% of cases, 50% of cases etc. In other words, we are interested in the inverse cumulative distribution function $F^{-1}(u)$, where $u \sim \mathcal{U}[0, 1]$ 220 and $F(x) = \mathbb{P}(X \leq x)$. If F is strictly increasing and absolutely continuous, then $x \equiv F^{-1}(u)$ is unique. A simple consistent estimate for x can then be found by sorting the values such that $X^1 < X^2 < \dots < X^N$ and then

$$\hat{F}^{-1}(u) = X^{\lceil N \times u \rceil}. \quad (16)$$

In other words, suppose in our hypothetical scenario we have 100 values for the water elevation at a given location after a given time. Then this expression simply says that the value x which the water elevation does not exceed 25% of the time, is 225 the 25th largest value. Gregory and Cotter (2017) show that this estimate is consistent because it converges in probability to x

as $N \rightarrow \infty$. Note that here converges in probability means that the probability of $X^{\lceil N \times u \rceil}$ being more than a small distance ϵ from x tends to zero as $N \rightarrow \infty$. In Gregory and Cotter (2017), they then use a formula to approximate $F_L^{-1}(u)$ from the MLMC outputs. In this work, we modify that formula to make it applicable for MLMF outputs so that the inverse cumulative distribution function for MLMF is approximated by

$$\begin{aligned}
F_L^{-1}(u) \approx & R^{HF}(X)_{l_\mu}^{\lceil N_{l_\mu}^{HF} \times u \rceil} + \alpha_{l_\mu} \left(\hat{R}^{LF}(X)_{l_\mu}^{\lceil N_{l_\mu}^{HF} \times u \rceil} - \hat{E} \left[\hat{Y}_l^{LF} \right] \right) \\
& + \sum_{l=l_\mu+1}^L \left(R^{HF}(X)_l^{\lceil N_l^{HF} \times u \rceil} - R^{HF}(X)_{l-1}^{\lceil N_{l-1}^{HF} \times u \rceil} \right) \\
230 \quad & + \sum_{l=l_\mu+1}^L \alpha_l \left(\hat{R}^{LF}(X)_l^{\lceil N_l^{HF} \times u \rceil} - \hat{R}^{LF}(X)_{l-1}^{\lceil N_{l-1}^{HF} \times u \rceil} - \hat{E} \left[\hat{Y}_l^{LF} \right] \right), \tag{17}
\end{aligned}$$

where $R^{HF}(X)_l^i$ and $\hat{R}^{LF}(X)_l^i$ represent the i^{th} order statistic of X_l on each level l of XBeach and modified correlation SFINCS (see 7), respectively. In other words, suppose that in our hypothetical scenario we want to know the value x which the water elevation does not exceed 25% of the time. We then pick the lower quartile value (*i.e.* the value not exceeded 25% of the time at each resolution for both models) and add them together following the rule of the MLMF estimator. Note that, unlike
235 with (B1), there cannot be exact cancellation because using this method means the approximations at each level are no longer unbiased.

2.3 Implementation

In this work, we construct our own Python MLMF wrapper around both SFINCS and XBeach to implement the MLMF algorithm. This wrapper can be shared on distributed cores of an HPC cluster to increase efficiency. Given the use of distributed
240 cores, any times quoted in this work are the total simulation times multiplied by the number of cores used. The different steps performed when running the models in the wrapper are illustrated in the flow chart of Figure 2. Note, in particular, that in this wrapper, the models are run and post-processed separately, meaning there is no issue with different input or output formats. Therefore, our MLMF wrapper is model-independent meaning it can be easily applied to other models and applications in further work.

245 For the models themselves, we use XBeach version 1.23.5526 from the XBeachX release and use the surfbeat mode to simulate the waves approaching the beach (Roelvink et al., 2018). SFINCS is not yet released in the public domain, but we use a version similar to that used in Leijnse et al. (2021).