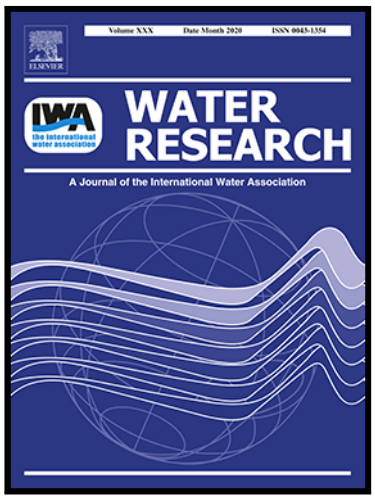

Gaussian Process Emulation of Spatiotemporal Outputs of a 2D Inland Flood Model

James Donnelly , Soroush Abolfathi , Jonathan Pearson , Omid Chatrabgoun , Alireza Daneshkhah

PII: S0043-1354(22)01046-6
DOI: https://doi.org/10.1016/j.watres.2022.119100
Reference: WR 119100

To appear in: *Water Research*

Received date: 13 April 2022
Revised date: 16 August 2022
Accepted date: 9 September 2022

Please cite this article as: James Donnelly , Soroush Abolfathi , Jonathan Pearson , Omid Chatrabgoun , Alireza Daneshkhah , Gaussian Process Emulation of Spatiotemporal Outputs of a 2D Inland Flood Model, *Water Research* (2022), doi: https://doi.org/10.1016/j.watres.2022.119100

Highlights

- Dimensionality reduction extends the class of simulators a gaussian process can model
- The emulator model accurately reproduces spatiotemporally varying inundation
- Predictive uncertainty quantified utilising the gaussian process framework
- Consistencies observed between emulator risk estimates and existing flood risks maps
- Gaussian process model observed to outperform alternative approaches to emulation

Gaussian Process Emulation of Spatiotemporal Outputs of a 2D Inland Flood Model

James Donnelly[1,2*], Soroush Abolfathi[2], Jonathan Pearson[2], Omid Chatrabgoun[3], Alireza Daneshkhah[1]

[1]Centre for Computational Science and Mathematical Modelling, Coventry University, CV1 5FB, Coventry, United Kingdom

[2]School of Engineering, University of Warwick, CV4 7AL, Coventry, United Kingdom

[3]School of Computing, Electronics and Mathematics, Coventry University, CV1 5FB, Coventry, United Kingdom

[*]Corresponding author E-mail address: donnel39@coventry.ac.uk

Abstract

The computational limitations of complex numerical models have led to adoption of statistical emulators across a variety of problems in science and engineering disciplines to circumvent the high computational costs associated with numerical simulations. In flood modelling, many hydraulic and hydrodynamic numerical models, especially when operating at high spatiotemporal resolutions, have prohibitively high computational costs for tasks requiring the instantaneous generation of very large numbers of simulation results. This study examines the appropriateness and robustness of Gaussian Process (GP) models to emulate the results from a hydraulic inundation model. The developed GPs produce real-time predictions based on the simulation output from LISFLOOD-FP numerical model. An efficient dimensionality reduction scheme is developed to tackle the high dimensionality of the output space and is combined with the GPs to investigate the predictive performance of the proposed emulator for estimation of the inundation depth. The developed GP-

based framework is capable of robust and straightforward quantification of the uncertainty associated with the predictions, without requiring additional model evaluations and simulations. Further, this study explores the computational advantages of using a GP-based emulator over alternative methodologies such as neural networks, by undertaking a comparative analysis. For the case study data presented in this paper, the GP model was found to accurately reproduce water depths and inundation extent by classification and produce computational speedups of approximately 10,000 times compared with the original simulator, and 80 times for a neural network-based emulator.

Keywords: Flood prediction, LISFLOOD-FP, Gaussian Process Emulator, Spatiotemporal outputs, Dimensionality reduction, Extreme event simulation

1. Introduction

Numerical modelling and simulations of flooding scenarios is a vital part of flood risk management, enabling timely planning and response to extreme climatic events. A range of complex numerical modelling techniques have been developed in the past decade to accurately simulate the flow-structure interactions and the corresponding hydrodynamic structure. These models differ on factors such as scale, spatial resolution (e.g., global, continental, basin), numerical techniques, complexity as measured by the number of dimensions considered to model the flow (1D, 2D and 3D), the underlying processes which lead to flooding events (i.e., inland and coastal flood), and the model outputs. Hence, there are significant differences between the capabilities, accuracy, and robustness of the available flood models. Much research has been conducted into reviewing the current hydrodynamic modelling strategies (Teng, et al., 2017; Bulti & Abebe, 2020), and assessing the performance of existing widely used flood models (Fewtrell et al., 2010; Zischg et al., 2017).

A primary drawback of the existing numerical simulations for flood modelling is the associated computational costs to generate results. Despite active research to enhance the computational capabilities of these models, accurate flood modelling at high spatiotemporal resolutions could still be prohibitively expensive. In most flood modelling studies, it is also desirable to understand the influence of varying the inputs on the outputs and simulate potential flooding scenarios. In-depth understanding of the uncertainty associated with the model outputs, and the model parameterisation is essential to establish a robust flood modelling framework. There are several approached to address these questions including sensitivity analysis (Borgonovo & Plischke, 2015; Oakley & O'Hagan, 2004; Daneshkhah & Bedford, 2013), uncertainty quantification (Soize, 2017; Ghanem et al., 2017), and model calibration (Lee et al., 2019; Kennedy & O'Hagan, 2001). However, these techniques generally require very large numbers of model runs, indicating that performing such tasks can become infeasible with even modest model computation times and so alternative modelling approaches are needed for robust and timely analysis of numerical models.

One approach to mitigate the high costs associated with numerical flood modelling tools is to utilise parallel and GPU computing techniques (Sharif, et al., 2020; Morales-Hernandez, et al., 2021; Neal, et al., 2018; Abolfathi et al., 2018; Yeganeh-Bakhtiary et al., 2020). At high temporal and spatial resolutions, although parallel CPU and GPU implementations can significantly reduce the computation time, a single simulation of such models can take many hours to complete (Shaw, et al., 2021). Despite the computational limitations, in many cases, numerical modelling is the most reliable tool for simulation and prediction of flooding and inundation especially when no alternative data-driven approaches (Van Steenbergen, et al., 2012; Abolfathi et al., 2016) can be utilised due to sparse empirical observations.

The adoption of machine learning (ML) approaches (Mosavi, et al., 2018; Yang & Chang, 2020; Lin, et al., 2020) in flood modelling could alleviate the high computational costs associated with numerical modelling, allowing a wider range of scenarios to be considered in a reasonable amount of time and

with more realistic computational resources. The increasing dependence on numerical models for applications in science and engineering and their associated computational costs has led to the development and adoption of a new class of ML-based models, known as surrogate models or probabilistic 'emulators'.

An emulator is a statistical model that approximates the outputs of a, usually complex model, where the model can be considered as a deterministic input-output computer simulator (Sarri et al., 2012). The emulator considered in this paper is originated from model output (O'Hagan, 2006), where working with the convenient simulators representing complex mathematical models is computationally very expensive. An emulator should approximate the output of a simulator to a sufficient level of accuracy with a significant reduction in associated computational cost, and the required training datapoints. In recent years, several studies focused on the use of emulators for a range of numerical modelling outputs including Tsunami models (Sarri et al., 2012; Salmanidou et al., 2017), flood models (Kabir et al., 2020), and climate projection models (Tran et al., 2019). A variety of modelling approaches have been examined for statistical emulation including Artificial Neural Networks (García-Alba et al., 2018; Wang et al., 2019), Polynomial Chaos Expansions (Massoud, 2018; Moreno-Rodenas, et al., 2018) and Gaussian Processes (Conti et al., 2009; Chang et al., 2015; Longobardi et al., 2020; Yang et al., 2018).

Kabir et al. (2020) proposed a deep convolutional neural network (CNN) model to emulate flood simulation results from a numerical model to generate rapid predictions of inundation levels. Benchmarked against a support vector regression (SVR) model, Kabir et al. (2020) found the CNN model outperforms the SRV with respect to classification metrics such as precision and recall, and slightly underperforms with respect to regression metrics such as RMSE and Nash-Sutcliffe efficiency. However, when validating the model, only a very small subset of the locations in the domain were chosen to validate the model at which may not provide the most accurate assessment of the model performance. The proposed emulator was more than 20 times faster compared to the

underlying numerical model. However, while the computational improvements are significant relative to the original numerical model, the runtime of the emulator may still be prohibitively computationally expensive for tasks such as scenario modelling, sensitivity analysis, and uncertainty quantification. Kabir et al. (2020) further concluded that, in contrast to GP, the CNN-based approach can not readily quantify the uncertainty associated with the emulator predictions.

There are several ML-based techniques, which have led to a variety of hybrid approaches to forecasting flood inundation depth utilising the computationally expensive physics-based numerical models and ML methodologies. These approaches often aim to reconstruct, or simplify, the modelling problem to make it more tenable for ML approaches and have had success in reconstructing inundation values (Yan et al., 2021; Zhou et al., 2021). However, in this study, a novel GP-based methodology is developed to emulate the outputs of a 2D hydraulic flood model, making as few modelling simplifications as possible. The proposed model will require far less model evaluations, and consequently significant reduction in the computational resources needed and model's runtime. The numerical model used in this study is LISFLOOD-FP, originally developed by Bates & De Roo (2000) for flood inundation simulations. Due to the high number of model outputs to be considered, a dimensionality reduction scheme is developed on the output space to reduce the computational complexity of modelling with multi-output GP models. The proposed model has low computational costs and is capable of straightforward quantification of predictive uncertainty. The appropriateness and robustness of the developed multi-outputs GP model for predicting flood inundation depths is examined for a case study data in North Yorkshire, UK, and a direct comparison is made between the outlined GP methodology and CNN model proposed by Kabir et al. (2020).

2. Methods

2.1. Numerical Flood Modelling

Numerical flood models simulate the interactions of water flow with the surrounding environment using partial differential equations (PDEs) that govern the motion of the fluid flow. Many open-

source models exist for modelling flood in two dimensions (i.e. spatial coordinates) including LISFLOOD, LISFLOOD-FP, MIKE-FLOOD and TELEMAC-2D. These models have been successfully validated for modelling inland and coastal flooding scenarios (Pinos & Timbe, 2019; Dong, et al., 2018; Abolfathi, et al., 2018). This study adopts LISFLOOD-FP , a two-dimensional hydraulic model (hereafter, 2D flood model), capable of utilising high resolution topographic data for simulating flood inundation depths in challenging urban settings. The accuracy of LISFLOOD-FP for predicting inundation levels is validated by several studies (e.g., Shustikova et al., 2019; O'Loughlin et al., 2019).

2D flood models make the simplifying assumption that the depth of the water is shallow compared with the magnitude of the other spatial dimensions and as a result, vertical variations in flow and velocity structure can be ignored. The shallow water equations are commonly used as the governing equations of 2D flood models to determine the interactions between fluid flow and the environment. LISFLOOD-FP simplifies the problem of simulating 2D flow by decoupling the flows in x and y directions and treats the simulation of 2D flow as a series of calculations in one dimension through the cell face boundaries (Shustikova et al., 2019). The governing equations used for flow modelling in this study including the conservation of mass (Eq. 1), and momentum (Eq. 2) are represented as follows:

$$\frac{\partial Q}{\partial x} + \frac{\partial A}{\partial t} = q, \#(1)$$

$$S_0 - \frac{n^2 P^{\frac{4}{3}} Q^2}{A^{\frac{10}{3}}} - \frac{\partial h}{\partial x} = 0, \#(2)$$

Where Q is the volumetric flow rate in the channel, A denotes the cross-sectional area of the flow, q is the flow into the channel from other sources (e.g., precipitations, wastewater discharge), S_0 is the bed slope, n denotes the Manning's roughness coefficient, P is the wetted perimeter of the channel, and h stands for the flow depth. The numerical model adopted in this study discretises the governing equations over square-type Eulerian grids, by discretising the domain into a rectangular

grid of $H \times W$ cells. The model uses an adaptive numerical time stepping scheme to determine Δt. Fluid flow between two cells is described as a function between the cells over time:

$$\frac{dh^{i,j}}{\Delta t} = \frac{Q_x^{i-1,j} - Q_x^{i,j} + Q_y^{i,j-1} - Q_y^{i,j}}{\Delta x \Delta y} \#(3)$$

where $h^{i,j}$ is the water free surface height at cell (i,j), Δx and Δy are the cell size in the x and y direction, , and $Q_x^{i,j}$ and $Q_y^{i,j}$ are the volumetric flow rates between cells in the floodplain. The reader should refer to Bates & De Roo (2000) for a full explanation of the governing equations and numerical implementation.

Flood simulations with LISFLOOD-FP requires parameterisation with a topographic dataset, boundary conditions describing fluid flows into and out of the model domains, choice of numerical solvers, and other case-specific parameters such as friction coefficients. In this study, assuming D sources of time-varying boundary conditions, LISFLOOD-FP can be considered as a function, $\boldsymbol{f}(\boldsymbol{x}^{(t)})$, taking inputs, $\boldsymbol{x}^{(t)} \in \mathbb{R}^D$, describing the time-varying boundary conditions at time t. The outputs generated at each simulation timestep is a matrix of inundation depths $\boldsymbol{Y}^{(t)} = \boldsymbol{f}(\boldsymbol{x}^{(t)}) \in \mathbb{R}^{H \times W}$ for a rectangular modelling domain of $H \times W$ cells, where $Y_{ij}^{(t)}$ denotes the water level in cell (i,j) at time t. However, for mathematical convenience, the matrices, $Y^{(t)}$ are flattened into vectors $\boldsymbol{y}^{(t)} \in \mathbb{R}^{HW}$ as illustrated in Fig. (1). Therefore, a single flood simulation with T timesteps can be described by the dataset $\mathcal{D} = \{(\boldsymbol{X}, \boldsymbol{Y}) | \boldsymbol{X} \in \mathbb{R}^{T \times D}, Y \in \mathbb{R}^{T \times HW}\}$, where $\boldsymbol{X} = \{\boldsymbol{x}^{(t)}\}_{t=1}^{T}$ and $\boldsymbol{Y} = \{\boldsymbol{y}^{(t)}\}_{t=1}^{T}$ are matrices corresponding to the row-wise aggregation of each $\boldsymbol{x}^{(t)}$ and $\boldsymbol{y}^{(t)}$.

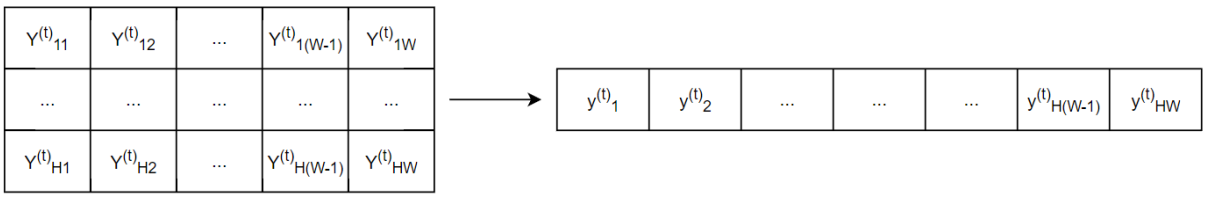

Fig. (1): Flattening water depth matrices $\boldsymbol{Y}^{(t)}$ into 1D vectors $\boldsymbol{y}^{(t)}$.

2.2. Gaussian Processes

A Gaussian Process is a stochastic process defined as a collection of random variables, any finite number of which have a joint Gaussian distribution (Rasmussen & Williams, 2006). In other words, a GP can be simply considered as a generalisation of the multivariate Gaussian distribution, retaining many convenient mathematical properties of the multivariate Gaussian distribution. However, instead of being defined over finite-length vectors and parameterised by a mean vector and covariance matrix, a GP is defined over functions and parameterised by mean and covariance functions. A GP is a (possibly infinite) distribution that is fully specified by its mean function $m(\boldsymbol{x}^{(i)})$ and its kernel-covariance function $k(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)})$, referred to as the kernel function. For a real process $f(\boldsymbol{x}^{(i)})$, such as a numerical model taking boundary conditions $\boldsymbol{x}^{(i)}$ at time $t = i$ as outlined in Section 2.1, the mean and kernel functions can be written as:

$$m(\boldsymbol{x}^{(i)}) = E[f(\boldsymbol{x}^{(i)})], \#(4)$$

$$cov(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}) = k(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}) = E\left[\left(f(\boldsymbol{x}^{(i)}) - m(\boldsymbol{x}^{(i)})\right)\left(f(\boldsymbol{x}^{(j)}) - m(\boldsymbol{x}^{(j)})\right)\right]. \#(5)$$

Leading to the prior distribution over functions,

$$f(\boldsymbol{x}^{(i)}) \sim GP\left(m(\boldsymbol{x}^{(i)}), k(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)})\right), \#(6)$$

which can be read as some function $f(.)$, at a given location in the input space $\boldsymbol{x}^{(i)}$, is distributed as a GP with mean function $m(.)$ and kernel function $k(.,.)$. A real kernel is a symmetric function which can approximately be considered to provide a metric of 'similarity' between two points $\boldsymbol{x}^{(i)}$ and $\boldsymbol{x}^{(j)}$. Eq. (6) outlines a prior distribution over target functions and encapsulates the prior beliefs about the space of functions from which the target function $f(.)$ could be drawn from. In practice, mean functions and kernel functions are replaced with finite length mean vectors, $\mu = \mathbf{0}$, and

covariance matrices, K, whose entries are $K_{ij} = k\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right)$. It should be noted that zero mean function assumption is common in ML applications, and this does not limit the mean of the posterior to zero (for further details see Rasmussen and Williams, 2006). The flexibility of GPs comes from the choices of kernel functions and their parameterisations. These choices will determine the properties of candidate functions such as smoothness and variability.

By combining the prior distribution with the observed dataset, the space of candidate functions is constrained to only contain functions which interpolate (exactly or approximately) the training datapoints, obtaining a posterior distribution over the function of interest. It is more common in practice to have a collection of noisy observations of some target function, $y^{(i)} = f\left(\boldsymbol{x}^{(i)}\right) + \varepsilon$, where ε is additive independently identically distributed Gaussian noise with variance σ_n^2. The prior on the noisy observations can be rewritten as:

$$K_{ij} = k\left(\boldsymbol{x}^{(i)}, x^{(j)}\right) + \delta_{ij}\sigma_n^2, \#(7)$$

where $\delta_{ij} = 1$ if $i = j$, otherwise $\delta_{ij} = 0$, and

$$cov(\boldsymbol{y}) = K(X, X) + \sigma_n^2 I, \#(8)$$

where $\boldsymbol{X} \in \mathbb{R}^{N \times D}$ is the design matrix consisting of N training instances, which here represents the numerical timesteps in flood simulations, in which each row corresponds to $\boldsymbol{x}^{(i)} \in \mathbb{R}^D$.

Therefore, given a set of observations $(\boldsymbol{X}, \boldsymbol{y})$, where $\boldsymbol{X}$ is the previously mentioned design matrix and $\boldsymbol{y} = \{y_i\}_{i=1}^N$ is a vector of noisy training outputs, a joint prior distribution over observed outputs, $\boldsymbol{y}$, and predicted outputs, $\boldsymbol{f}_*$ at a set of test inputs X_*, can be defined as:

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} K + \sigma_n^2 I & K_* \\ K_*^T & K_{**} \end{bmatrix}\right), \#(9)$$

where,

$$K \in \mathbb{R}^{N \times N} = k(X, X), \#(10)$$
$$K_* \in \mathbb{R}^{N \times N_*} = k(X, X_*),$$

$$K_*^T \in \mathbb{R}^{N_* \times N} = k(X_*, X),$$
$$K_{**} \in \mathbb{R}^{N_* \times N_*} = k(X_*, X_*)$$

By conditioning the prior distribution on the observed data, the posterior distribution of zero-mean Gaussian process over the test inputs is given as:

$$\boldsymbol{f}_* | \boldsymbol{y}, X, X_* \sim \mathcal{N}(\boldsymbol{\mu}_*, \Sigma_*) \quad (11)$$

where,

$$\boldsymbol{\mu}_* = K_*^T (K + \sigma_n^2 I)^{-1} \boldsymbol{y} \#(12)$$

$$\Sigma_* = K_{**} - K_*^T (K + \sigma_n^2 I)^{-1} K_* \#(13)$$

Eq. (11) shows that, within the GP framework, prediction is equivalent to the construction of a full multivariate Gaussian posterior distribution over function values at new unseen points in the input space. Hence, $\boldsymbol{\mu}_*$ can be considered as the mean prediction and Σ_* the variance associated with the mean prediction.

Learning the hyperparameters

In the GP modelling framework, assuming a zero-mean, training the model means choosing an appropriate kernel function and using the training data to optimise the kernel's hyperparameters. The default kernel used in GP regression models is the Squared Exponential (SE) kernel:

$$k\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right) = \sigma^2 \exp\left(-\frac{\left\|\boldsymbol{x}^{(i)} - \boldsymbol{x}^{(j)}\right\|}{2l^2}\right), \#(14)$$

where $\sigma^2$is the output variance, determining the average distance of the function from the mean value, and l is the lengthscale which determines how quickly the function varies and determines how the covariances decay with distance between points. The SE kernel function generalises well to many applications, yet it makes assumptions about smoothness which often are not realistic. In this study, a zero-mean function and the Matern$_{3/2}$ kernel function are used for the development of the

GP model. The Matern$_{3/2}$ kernel is a stationary kernel, operating only on the Euclidean distance between points, with origins in spatial statistics and is commonly used for modelling physical processes (Rasmussen & Williams, 2006). Furthermore, preliminary testing showed the Matern$_{3/2}$ (Eq. 15) to outperform the SE for the flood modelling context addressed in this study.

$$k\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right) = C_{\frac{3}{2}}(d) = \sigma^2\left(1 + \frac{\sqrt{3}\left\|\boldsymbol{x}^{(i)} - \boldsymbol{x}^{(j)}\right\|_2}{l}\right)\exp\left(-\frac{\sqrt{3}\left\|\boldsymbol{x}^{(i)} - \boldsymbol{x}^{(j)}\right\|_2}{l}\right) + \sigma_n^2\delta_{ij}, \#(15)$$

where $\delta_{ij} = 1$ if $i = j$, otherwise $\delta_{ij} = 0$. The kernel's hyperparameters, denoted by $\boldsymbol{\theta} = (\sigma^2, l, \sigma_n^2)$, are determined through Maximum likelihood estimation with respect to each hyperparameter. Where σ^2 is the variance parameter, l is the length-scale, and σ_n^2 is a Gaussian noise parameter.

To estimate kernel hyperparameters the log marginal likelihood is maximised with respect to the hyperparameters. Given that prior distribution over observations can be expressed as $\boldsymbol{y}\sim N(\mathbf{0}, K + +\sigma_n^2 I)$, the log marginal likelihood, explicitly conditioned on kernel hyperparameters, can be written as:

$$\log p(\boldsymbol{y}|X, \boldsymbol{\theta}) = -\frac{1}{2}\boldsymbol{y}^T(K + \sigma_n^2 I)^{-1}\boldsymbol{y} - \frac{1}{2}\log|K + \sigma_n^2 I| - \frac{n}{2}\log 2\pi\,. \#(16)$$

By taking partial derivatives with respect to each hyperparameters the marginal likelihood can be maximised as:

$$\frac{\partial}{\partial\theta_j}\log p(\boldsymbol{y}|X, \boldsymbol{\theta}) = \frac{1}{2}\boldsymbol{y}^T K_y^{-1}\frac{\partial K_y}{\partial\theta_j}K_y^{-1}\boldsymbol{y} - \frac{1}{2}tr\left(K_y^{-1}\frac{\partial K_y}{\partial\theta_j}\right), \#(17)$$

where $K_y = K + \sigma_n^2 I$ and tr is the trace operator. Performing this operation scales with $O(N^3)$ computational complexity from the inversion of K_y matrix.

2.3. Dimensionality Reduction

With the LISFLOOD-FP model, the outputs describe inundation across the study area for each timestep. When constructing a statistical emulator, a collection of K design floods $\{\mathcal{D}_i\}_{i=1}^{K}$, where $\mathcal{D}_i = (\boldsymbol{X}_i \in \mathbb{R}^{T_i \times D}, \boldsymbol{Y}_i \in \mathbb{R}^{T_i \times HW})$, consisting of T_i timesteps, where H is the number of cells in the model in the x-direction and W in the y-direction is used to train and validate the emulator. The input-output data matrices from each design flood are aggregated into new input-output matrices, $\boldsymbol{X} \in \mathbb{R}^{N \times D}$ and $\boldsymbol{Y} \in \mathbb{R}^{N \times HW}$, with a new number of aggregated timesteps $N = T_1 + \cdots + T_K$. For numerical simulators operating at high spatiotemporal resolutions, this can result in very large values of HW and N.

To overcome computational issues arising from modelling a very high-dimensional output using GPs, a dimensionality reduction (DR) scheme is first developed and applied to the outputs of LISFLOOD-FP simulations, making the construction of a GP-based statistical emulator more feasible. Principal Component Analysis (PCA) decomposition is implemented using a randomised implementation of Single Value Decomposition (SVD) (Halko et al., 2011; Feng et al., 2018). This transformation produces a lower-dimensional dataset of latent features, $\boldsymbol{Z} \in \mathbb{R}^{N \times D^*}$, to replace the original dataset $\boldsymbol{Y} \in \mathbb{R}^{N \times HW}$. The output features in the original dataset correspond to the water depths in cells of the output domain, and so it can be expected that there will exist significant linear spatial correlations in the data, a structure which PCA can effectively represent in a lower-dimensional space (Cheng et al., 2022). Furthermore, PCA relies on linear transformations allowing it to scale easily to very high-dimensional datasets and has a computationally efficient decoding process to reconstruct the original observations from the latent features.

2.4. PCA-GP development

For standard GP regression problems, only functions of the form $f: \mathbb{R}^D \to \mathbb{R}$ are considered, such that when performing GP regression, a scalar output, $y^{(i)}$, is regressed onto a vector valued input $\boldsymbol{x}^{(i)}$. This type of GP regression scales cubically with the number of training instances, $O(N^3)$, which

means the GP has the order of N^3 time complexity or express the runtime in terms of how quickly it grows relative to the size of the input, as the size of input gets larger. However, after the construction of the latent feature dataset, $\boldsymbol{Z}$, there are D^* outputs to be modelled and so construction of the emulator is equivalent to a vector valued regression problem in which the mapping $\boldsymbol{f}: \mathbb{R}^D \rightarrow \mathbb{R}^{D^*}$ is learned. Approaches for vector valued GP regression have been explored and methods such as the linear model of coregionalization (LMC) have been proposed (Alvarez et al., 2012). However, exact implementation of these methods would result in the computational complexity of $O((D^* \cdot N)^3)$.

Using the reduced dataset, $\boldsymbol{Z}$, a collection of independent single output GPs (SOGP) is utilised to model each latent feature independently given that applying PCA constructs a collection of orthogonal latent features, possibly removing correlations between output features. The exact implementation of this methodology will scale with $O(D^* \cdot N^3)$ computational complexity. As it is evident, the computational time of SOGP is at least $(D^*)^2$ time faster than the LCM method. Fig. (2) illustrates the computational complexity of alternative approaches to vector valued GP regression for a varying number of features, D^*, using a baseline of $N = 1000$ training instances. This figure implicitly highlights how infeasible a naive implementation of the original dataset would be for which $D^* \approx 800{,}000$.

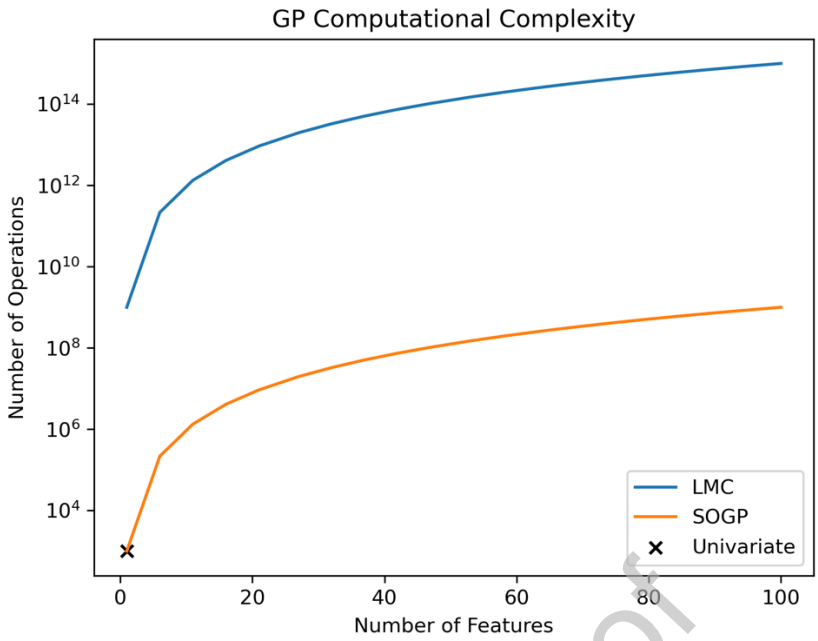

Fig. (2): Computational complexity analysis of the LCM model (Alvarez et al., 2012) and the SOGP model developed in this study in terms of the number of output features and training points. The univariate GP's complexity is also highlighted. A baseline of N=1000 training instances is used for the calculations.

Fig. (3) shows a graphical representation of the GP model with D input features and D^* independent output features, mapping each input vector to a scalar via a GP f_j for $j = 1, \ldots, D^*$. For each GP in this model, a zero-mean function and the Matern$_{3/2}$ kernel function are used with the hyperparameters for each determined as outlined in Section 2.2.

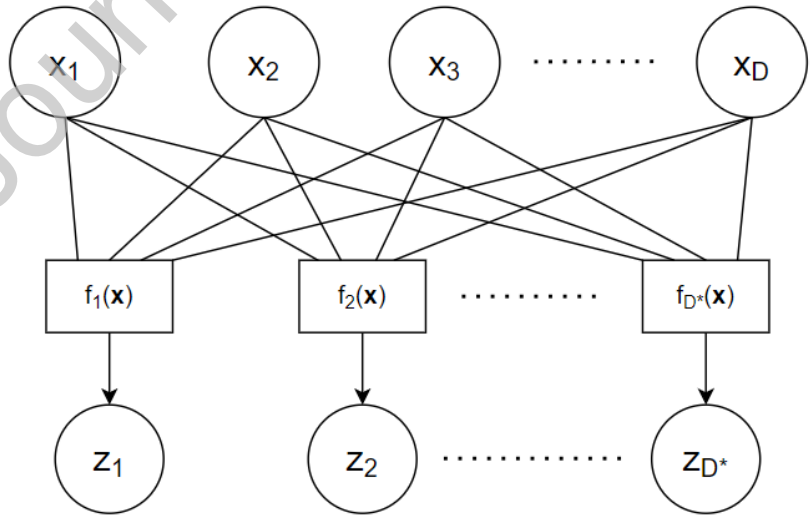

Fig. (3): Graphical Representation of GP model for D input features and D^* independent latent outputs.

2.5. Validation Metrics

When building statistical emulators, the primary concerns are predictive accuracy and time taken to generate new predictions. In this study, the modified versions of the root mean squared log-error (RMSLE) and root mean squared error (RMSE), defined by Eqs. (18) and (19), respectively, will be used for assessing regression performance.:

$$RMSLE = \sqrt{\frac{\sum_{\Omega}\left(\log y_{\Omega_i} - \log \hat{y}_{\Omega_i}\right)^2}{|\Omega|}}, \#(18)$$

$$RMSE = \sqrt{\frac{\sum_{\Omega}\left(y_{\Omega_i} - \hat{y}_{\Omega_i}\right)^2}{|\Omega|}}, \#(19)$$

where for each simulation, Ω is the set of cell indexes in which either LISFLOOD-FP or the GP emulator predict an inundation value greater than 0 at any time-step. This reduces the number of cells averaged over, providing a more realistic error metric because most of the cells will never see any inundation, causing the average error to be reduced. Furthermore, the log-error is used as well as ordinary RMSE, as it could give a more realistic weighting to the errors. Given that the relative error of prediction is more important than the absolute value, and the error should be weighted accordingly. For instance, in a deep section of the river, the emulator's prediction error may be large in an absolute sense, but this might not be the case when scaled against the true value. Whereas in the floodplain, where inundation is lower, a small absolute error could represent a significant difference in the interpretation of the results.

The model's ability to correctly classify wet/dry cells is also assessed. To convert regression predictions into a classification task, an inundation threshold (c) is applied as a binary classifier to determine if a cell is wet or dry:

$$y = \begin{cases} 1, y > c \\ 0, y \leq c \end{cases} \#(20)$$

Different values of c were trialled including 0.05m, 0.1m and 0.3m. A threshold of $c = 0.3$ is recommended following the flood risk assessment conducted by Aldridge et al. (2016), in which it was concluded that properties intersecting a flood depth of 0.3m would be determined as flooded. Having established a binary classification scheme, an assessment of classification performance is conducted using true positive rate (TPR), false positive rate (FPR), and F1 score, defined by Eqs. (21)-(23), respectively:

$$TPR = \frac{TP}{TP + FN}, \#(21)$$

$$FPR = \frac{FP}{FP + TN}, \#(22)$$

$$F1 = \frac{TP}{TP + \frac{1}{2}(FP + FN)}, \#(23)$$

where TP and TN are the number of true positives and negatives, respectively, FP and FN are the number of false positives and negatives, respectively.

3. Case Study

3.1. Site description and modelling domain

The GP modelling framework developed in this study is adopted for a case study of urban flood modelling in Tadcaster, UK. Tadcaster is a town situated between York and Leeds in the Northeast of England. The town is prone to flooding from the river Wharfe which is running through the centre of the town. In December 2015 (Fig. (4)), in the aftermath of Storm Frank, the town faced its worst flood in recorded history during which many of businesses and homes were evacuated and the town's critical infrastructure (e.g., bridge and roads) suffered major damages from the force of the flood waters. As a result of the extensive flood damages from the 2015 Storm Frank, a £10m flood defence scheme was proposed to enhance the flood resilience of the town for extreme climatic

events. However, the project execution and completion has been delayed until 2026 due to 'modelling inaccuracies' (The York Press, 2021). Hence, this study examines the performance of the proposed GP-based emulator for flood scenario modelling of Tadcaster with the aim of producing instantaneous inundation predictions and allowing for the quantification of uncertainty surrounding the model predictions. Fig. (5) illustrates the 3.6km^2 study domain that is considered for the case study in this paper.

Fig. (4): Tadcaster 2015 flood from Storm Frank where (a) Shows aerial imagery of the flood affected zone (source: Environment Agency), and (b) highlights an example of the extent of the damaged infrastructure (source: BBC, 2016)

Fig. (5): Case study location in Tadcaster and the modelling domain for (a) map overlay and (b) satellite imagery overlay.

3.2. Topographic data

LISFLOOD-FP was parameterised with a Digital Elevation Map (DEM) describing the topography of the modelling domain. The DEM was constructed using Digital Terrain Models (DTM) to capture the elevation of the underlying surface and then key geographical features and urban structures were reinserted to ensure accurate and realistic flow-routing in an urban environment. Flood modelling in urban environments requires high resolution spatial data, and therefore this study constructed the DEM with a 2m resolution, resulting in a two-dimensional Eulerian mesh of 876,204 cells. Fig. (6) compares elevation (in meters above the sea level) for the original DTM (Fig. (6a)) and the post-processed DEM used in the flood model (Fig. (6b)).

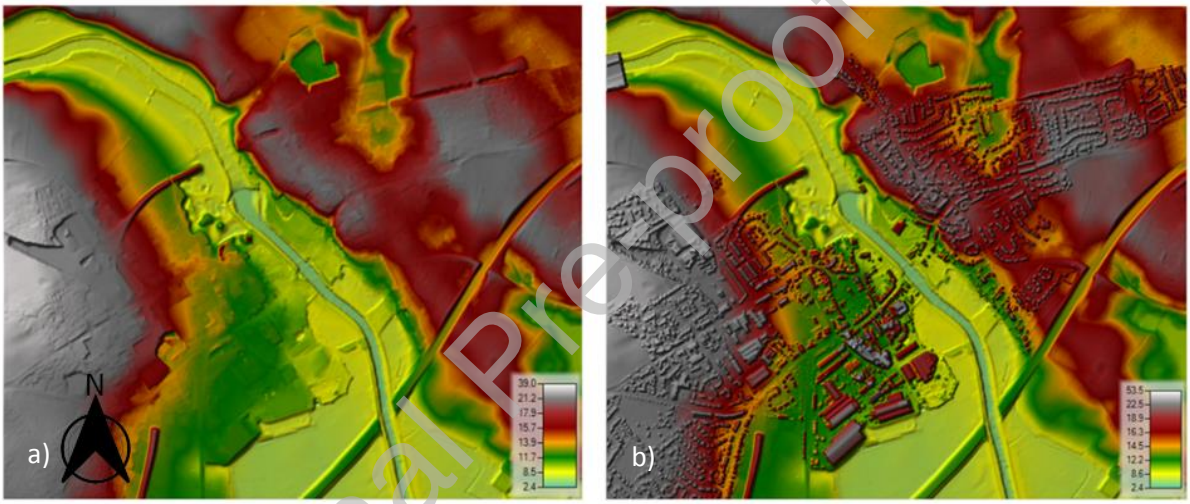

Fig. (6): The case study modelling domain elevation map for (a) the unprocessed DTM at 2m cell resolution, and (b) the post-processed DTM with buildings and urban infrastructures reinserted.

3.3. Boundary conditions

Time-varying point-source hydrographs, at upstream points at the North side of the domain, were used as the boundary conditions for each simulation scenario alongside a free boundary at the Southern downstream boundary. The hydrographs were determined using discharge data collected from a river monitoring station within the domain. From the empirical discharge data, a collection of candidate flood hydrographs was extracted as the basis for synthetic hydrographs. Then, by sampling peak-discharge values from a Pareto distribution fitted using historical Peaks-over-

threshold data, the hydrographs were re-scaled to match the peak-discharge values. This study focused on modelling extreme climatic events by oversampling from the upper tails of the peak-discharge distribution to investigate scenarios with more severe flooding. This study simulates 14 synthetic flood scenarios, described by datasets $\{\mathcal{D}_i\}_{i=1}^{14}$, to investigate the extent of inundation and flood water depth across the domain. For each hydrograph, a discharge value is recorded for the river Wharfe at 15-minute intervals, and input to the numerical flood model.

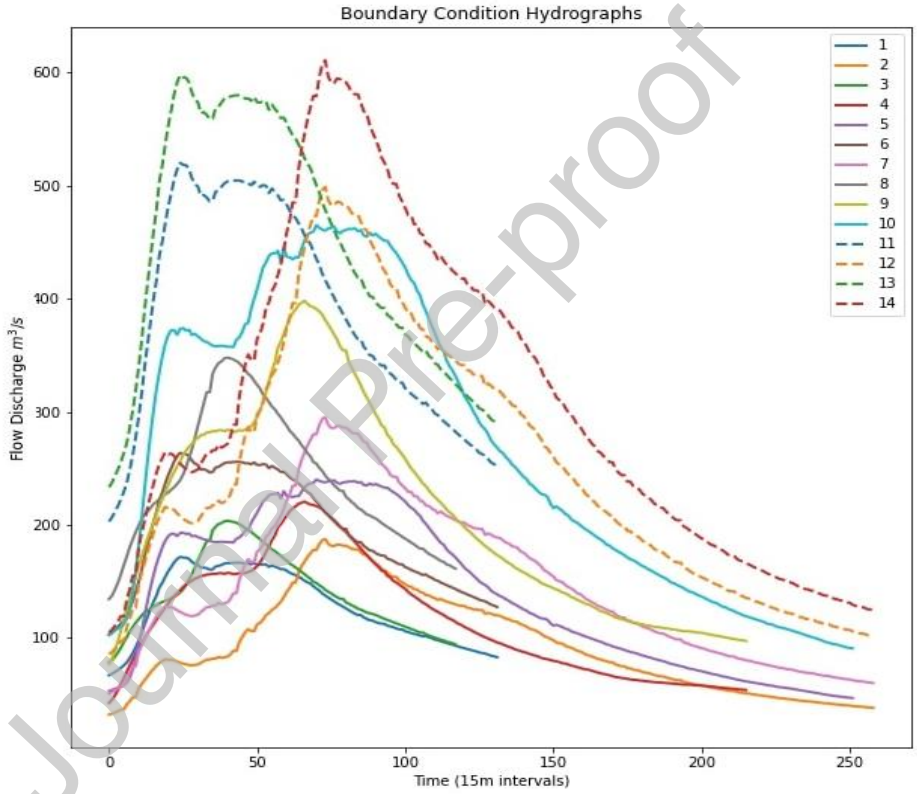

Fig. (7): 14 Synthetic hydrographs used as boundary to parameterise LISFLOOD-FP simulations

3.4. LISFLOOD-FP Inundation Data

Following implementation of the topographic features of the case study location, and the boundary conditions, the LISFLOOD-FP was used to simulate the flood for the synthetic hydrographs. A spatially varying manning's friction coefficient was applied with values determined based on the land

usage (Papaioannou et al., 2018). Fig. (8) presents the land cover map with the respective Manning's coefficient values.

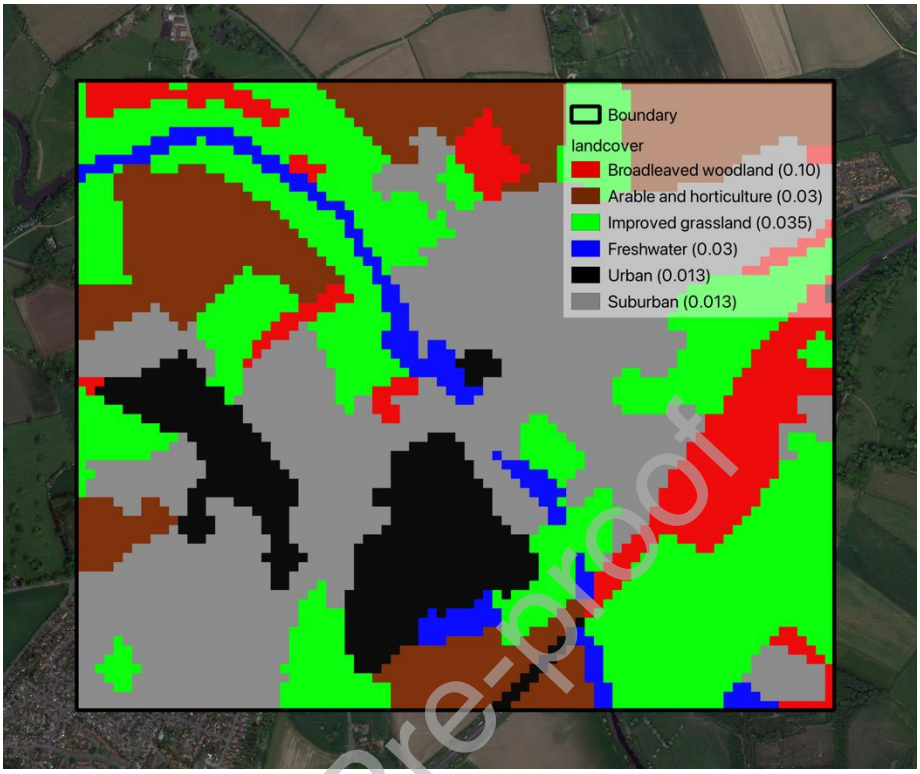

Fig. (8): Land usage and associated Manning's coefficients.

To minimize the potential uncertainty associated with the simulation results, an inundation threshold of 0.03m was applied, where an inundation less than 0.03m was assigned to 0. Following the completion of the simulations, the water depth inundation data was aggregated across simulations. The discharge value for time $t = i$ along with the values at $t = i - i, i - 2, \dots, i - 8$ is used as training input for the GP emulator. These input features were then scaled to have zero mean and unit variance, resulting in aggregated inputs $\boldsymbol{X} \in \mathbb{R}^{2624\times 9}$ and outputs $\boldsymbol{Y} \in \mathbb{R}^{2624\times 876204}$ for the construction of the emulator model, i.e. 2624 aggregated timesteps across all 14 simulations and $876{,}204$ features (cells) for each timestep.

3.5. Latent Feature Dataset

The dimensionality reduction approach outlined in Section 2.2 was then applied to $\boldsymbol{Y}$ to produce a dataset of latent variables $\boldsymbol{Z} \in \mathbb{R}^{2624 \times D^*}$. It was determined to set $D^* \coloneqq 6$ after experimentation found that the first six principal components were sufficient to explain 99% of the variance within the dataset. To assess the fit of this PCA decomposition, the RMSE between the original dataset and the reconstruction of the latent features back to the original features space was assessed (Eq. 24). The RMSE obtained was 0.018m meaning the reconstruction deviates from the original by 1.8cm on average.

$$RMSE\big(\boldsymbol{Y}, \phi^{-1} \circ \phi(\boldsymbol{Y})\big) \#(24)$$

Performing 10-fold cross-validation on the PCA transformation found the result obtained from Eq. (24) to be highly consistent across folds, showing the methodology generalises well to out-of-sample test instances. Fig. (9) illustrates the new dataset, $\boldsymbol{Z}$, plotting each of principal component values aggregated across all the hydrographs, the delineation between each of the 14 hydrographs is highlighted by the blue dashed lines. The unit of measurement on the y-axis can be ignored as the regression performance in the latent feature space is not interpreted.

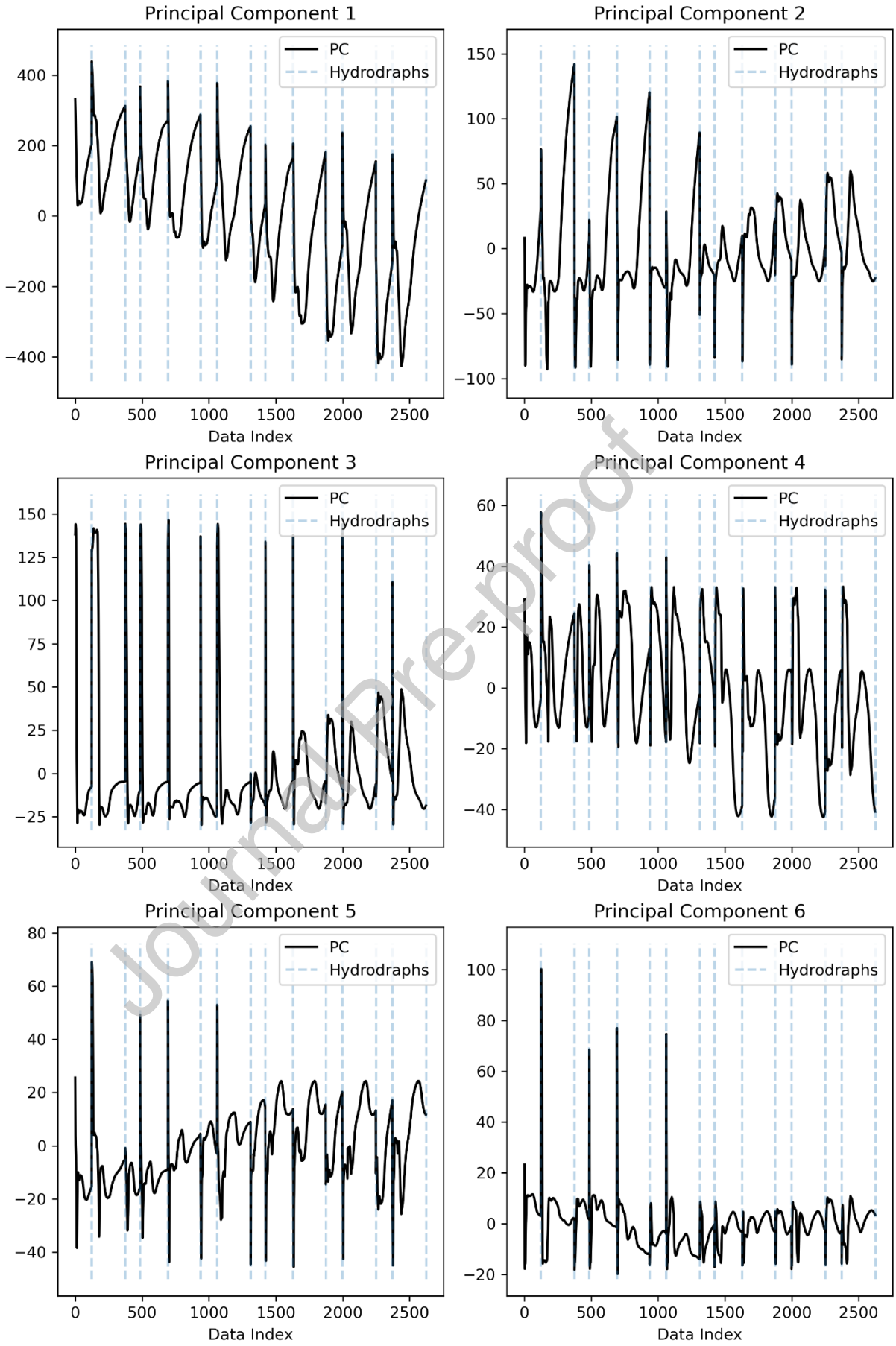

Fig. (9): Aggregated outputs in the latent feature space corresponding to the first 6 principal components of the original dataset. Blue dashed lines delineate between 14 input hydrographs.

Fig. (10) illustrates a visual representation of the complete methodology employed in this study to build and train the GP emulator for inland flood modelling, highlighting the sequential modelling process and data sources. Fig. (11) shows how the constructed emulator can then be employed to make predictions on unseen data as a replacement for the LISFLOOD-FP flood simulation model.

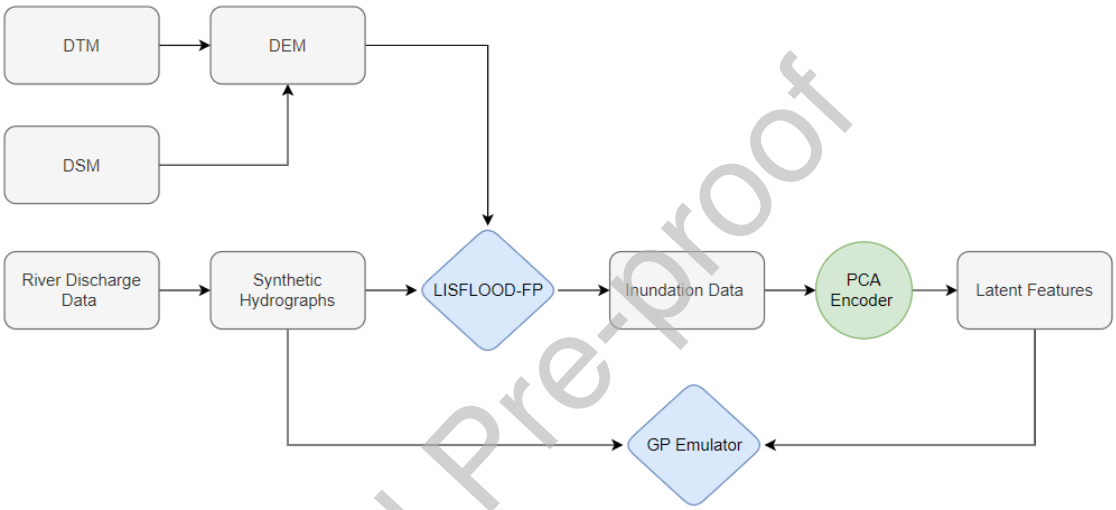

Fig. (10): Data sources and sequential modelling process to build and train the GP emulator outlined in this study

Fig. (11): Methodology to generate new predictions using GP emulator, replacing the original simulator with the emulator.

4. Results and Discussion

4.1. Cross Validation Results

To validate the GP model and assess out-of-sample performance, leave-one-out cross-validation (LOOCV) method was implemented whereby the GP-based emulator is trained on 13 synthetic hydrographs (Fig. (7)) and predictive performance is assessed on the remaining hydrograph, repeating this process for all 14 training examples. Fig. (12) illustrates the model's regression performance, showing the test RMSLE and RMSE values for each held-out hydrograph. The mean RMSLE and RMSE values are 0.11 and 0.21, respectively. As expected, the RMSLE values are lower than the RMSE, and are theoretically expected to be a better estimate of the model's error. Reasonable variance among the results from cross-validation testing is observed, with hydrographs 8 being somewhat of an outlier. However, overall,] consistency in performance across the test cases are evident.

While good regression performance is important, the ability to correctly identify areas as wet/dry is essential for flood risk management and produce interpretable forecasts. As such, classification was also considered utilising the scheme outlined in Eq. (20). Fig. (13) shows the cross-validation classification performance, differentiating the model performance by the threshold value c used. Table (1) provides a full breakdown of the mean (averaged across cross-validation folds) scores for each metric at each threshold. The scores remain relatively consistent across the threshold values with no value being clearly favourable. However, $c = 0.05$ appears marginally better than the other values. Overall, the GP model clearly shows effectiveness at predicting whether cells will be wet or dry.

Table (1): Cross-validation classification scores

	$\boldsymbol{c = 0.05}$	$\boldsymbol{c = 0.1}$	$\boldsymbol{c = 0.3}$
F1	0.940	0.941	0.937
Recall	0.968	0.958	0.937
FNR	0.0046	0.0058	0.0078

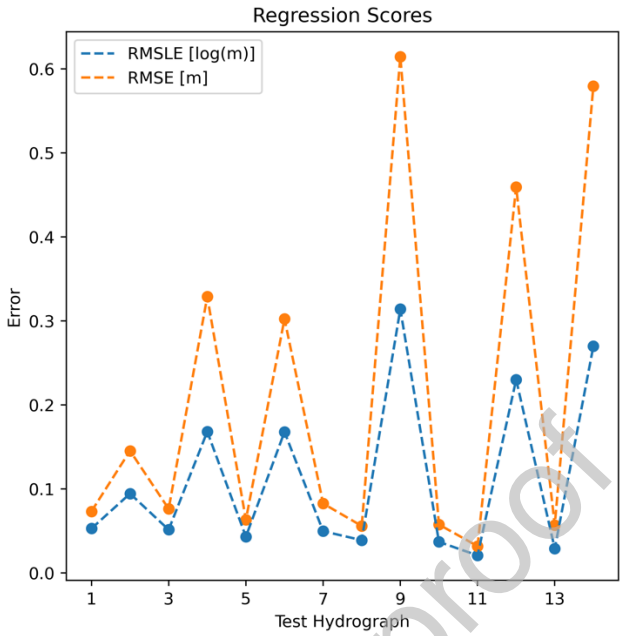

Fig. (12): Leave-one-out cross-validation scores for regression performance.

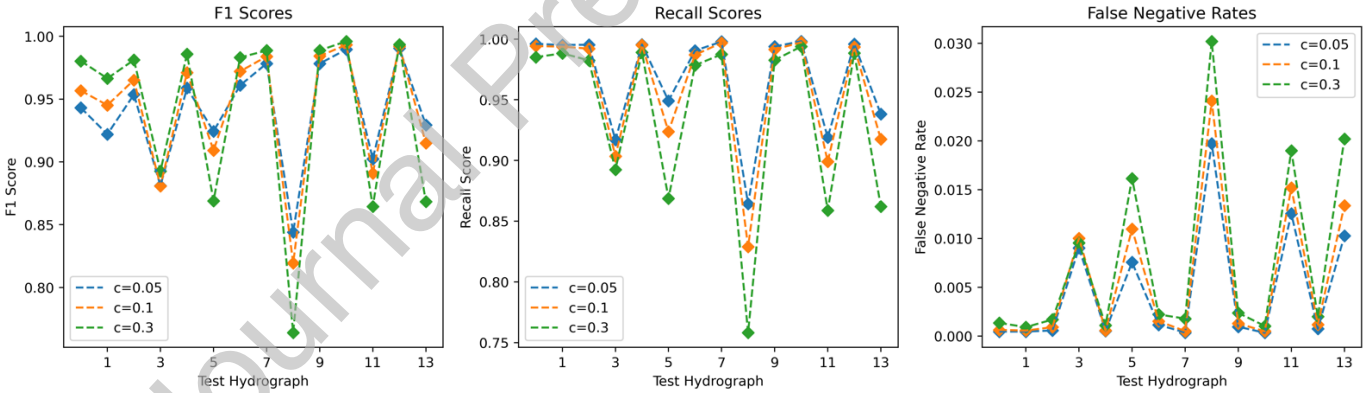

Fig. (13): Cross-validation classification scores for varying discretisation thresholds.

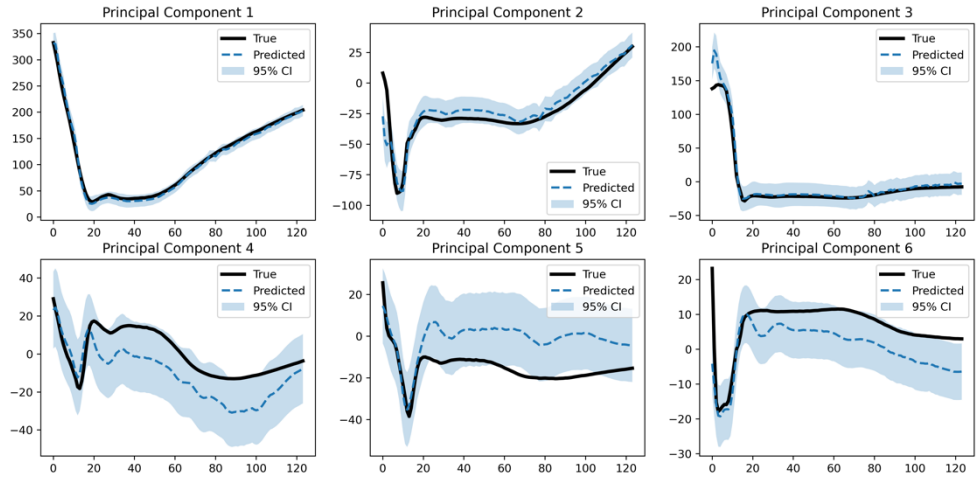

Fig. (14): Cross-validation predictions for hydrograph 1 latent feature predictions with 95% confidence intervals.

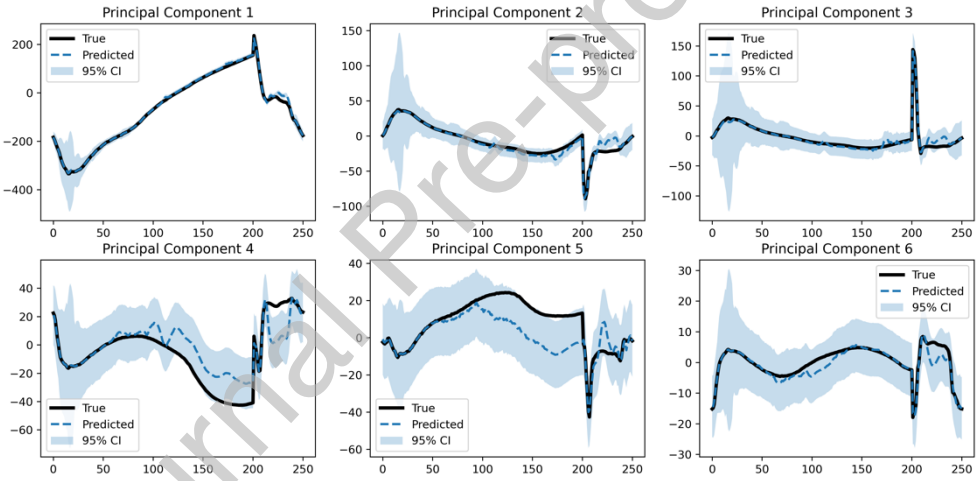

Fig. (15): Cross-validation predictions for hydrograph 12 latent feature predictions with 95% confidence intervals.

.

Figs. (14)-(15) illustrate regression performance in the latent feature space with 95% predictive intervals highlighted. A common feature of these plots is the ability of each GP to accurately predict the value of the first 3 principal components in each test case. The performance on the following 3 components is more varied, and the model expresses greater uncertainty surrounding the predictions for these features as illustrated by the significantly wider confidence intervals. However, the earlier principal components capture most of the variance within the dataset and so accurate

prediction of these is significantly more important than accurate prediction of the latter components.

Fig. (16): Hydrograph 7 Maximum Depth across flood events, with a threshold of 0.3m to highlight inundation, (a) Maximum Inundation calculated by LISFLOOD-FP, and (b) Maximum Inundation calculated by GP emulator.

Fig. (17): Hydrograph 9 Maximum Depth across flood events, with a threshold of 0.3m to highlight inundation, (a) Maximum inundation calculated by LISFLOOD-FP, and (b) Maximum inundation calculated by GP emulator.

Figs. (16)-(17) illustrate results of flood modelling for hydrographs 7 and 9 during the cross-validation process (the data from these simulations is held out for testing while the model trains on the other 13 hydrographs), respectively, in the original feature space after projecting predictions on

the latent feature space back to the high-dimensional space. The maximum water depth in each cell across the entire simulation is plotted, and then a water depth threshold of 0.3m is applied to the cells. If the maximum inundation across the flood event is greater than 0.3m the cells is highlighted and left blank otherwise.

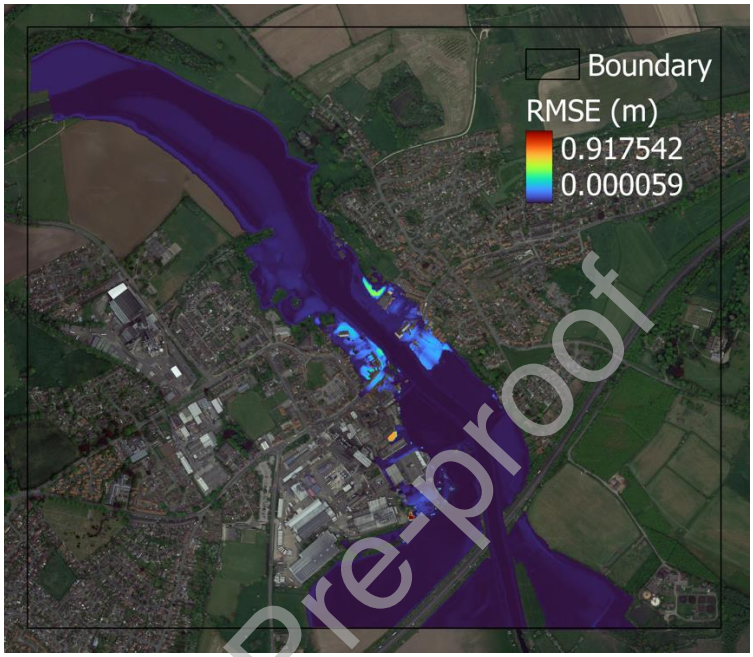

Fig. (18): Mean inundation error between LISFLOOD-FP and the GP emulator with inundation averaged across time for testing on hydrograph 9 during cross-validation.

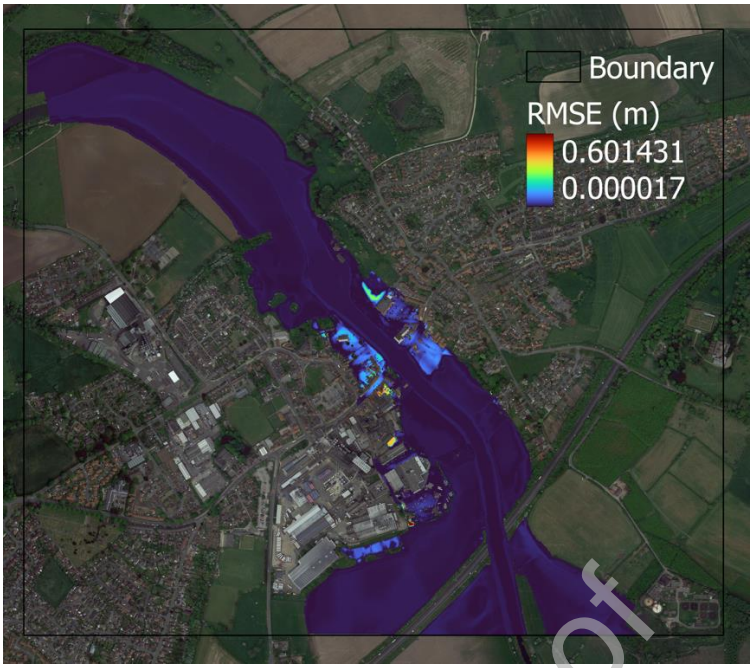

Fig. (19): Mean inundation error between LISFLOOD-FP and the GP emulator with inundation averaged across time for testing on hydrograph 14 during cross-validation.

Figs. (18-19) illustrates the difference between the average water depth produced from LISFLOOD-FP simulation and the predictions by the GP emulator. This allows the exploration of the average prediction error on a cell-by-cell basis. Similar results are observed in both scenarios with the GP model's error being approximately 0 for very large sections of the inundated areas, and smaller areas of higher errors was observed. In both cases, the areas of higher error correspond to the areas with more complex urban topography which are in closest proximity to the river reaches. To improve future model performance in similar scenarios, the training data could be chosen such that a wide range of inundation scenarios in these areas is observed.

4.2 Benchmarking

To robustly assess the performance of the proposed GP model against alternative methods of statistical emulation, the CNN model (Fig. (20)) outlined in Kabir et al. (2020) was reconstructed for this study to make a direct comparison of predictive performance between the two methodologies.

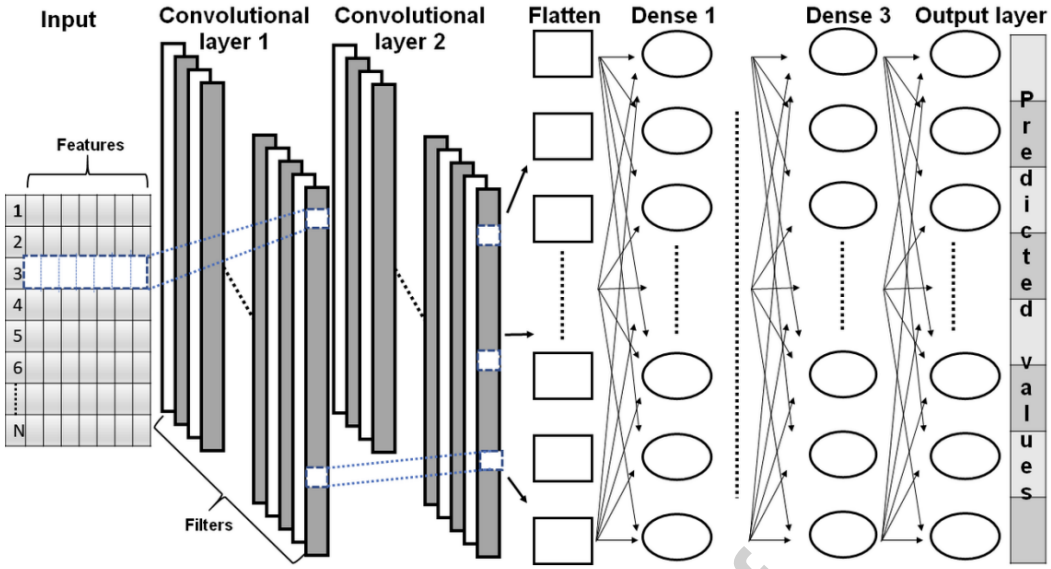

Fig. (20): The reconstructed CNN model proposed in Kabir et al., (2020) for the case study introduced in Section3.

Performing the same approach to cross-validation by training 14 different CNNs, each time holding one hydrograph out as validation data, a direct comparison of GP and CNN model performance can be made. Table (2) shows the cross-validation regression performance, and Table (3) describes the classification performance of each model.

Table (2): Cross-validation regression performance

	RMSLE	**RMSE**
GP	0.112	0.209
CNN	0.122	0.232

Table (3): Cross-validation classification performance

$c = 0.05$	**F1**	**Recall**	**FNR**
GP	0.940	0.967	0.0046
CNN	0.940	0.960	0.0051
$c = 0.10$			
GP	0.941	0.958	0.0058
CNN	0.940	0.955	0.0061
$c = 0.30$			
GP	0.937	0.936	0.0078
CNN	0.935	0.935	0.0081

Although performance of both models is quite similar, the results show that the GP model outperforms CNN with respect to both regression and classification. These results confirm the findings of previous studies in terms of the ability of deep learning methodologies to tackle these complex predictive tasks. However, this study proves the capability of the proposed GP methodology to outperform alternative emulations, which cannot be directly evaluated for uncertainty quantification. The proposed GP model is also far more robust for scenario modelling compared to alternative emulators when the computational costs is significant (See Section 4.3).

4.3 Model Runtimes

Table (4) outlines training time, run time, and model size (i.e., number of cells in modelling domain) for the original LISFLOOD-FP simulator, the GP-based emulator developed in this study, and the CNN emulator reconstructed for this study. The run-time is calculated as the time taken to generate new predictions for a simulation with 250 timesteps of 15-minutes. The results show a significant reduction in runtime when using the GP emulator proposed in this study over the CNN. However, for both methods of statistical emulation, very significant reductions in runtime are observed as compared with the original numerical model. The GP emulator is shown to be 80 times faster than the CNN model adopted for the considered case study.

Table (4): Computational time requirements

Model	Training Time (s)	Runtime (s)	No. Cells
LISFLOOD-FP	-	9,900	876,204
CNN Emulator (Current Study)	600	80	876,204
GP Emulator (Current Study)	226	1	876,204

Constructing emulators has considerable developmental cost, given that training data are generated from the original numerical model which can be costly. Although this computational cost is common for all surrogate models, the GP model can be efficiently trained with considerable limited training

datapoints which makes it a very cost-efficient emulator. Furthermore, modelling decisions need to be made during development of surrogate models (e.g., network structure for a CNN, or kernel for a GP). The training times in Table (4) only represent the time taken to estimate model hyperparameters. However, the overall development time associated with emulator construction is higher due to data collection, model training, and validation.

5. Application of GP Emulator for flood predictions

As an example of replacing the efficient GP-based emulator with the simulator, a flood was simulated using an estimated 1000-year return period synthetic hydrograph. Using the previously constructed peak discharge distribution (see Section 3.3), the estimated peak discharge was 750m^3/s. For reference, the estimated peak discharge in the 2015 flood was 480m^3/s. A candidate synthetic hydrograph was scaled to match this estimated peak discharge value, as shown in Fig. (21).

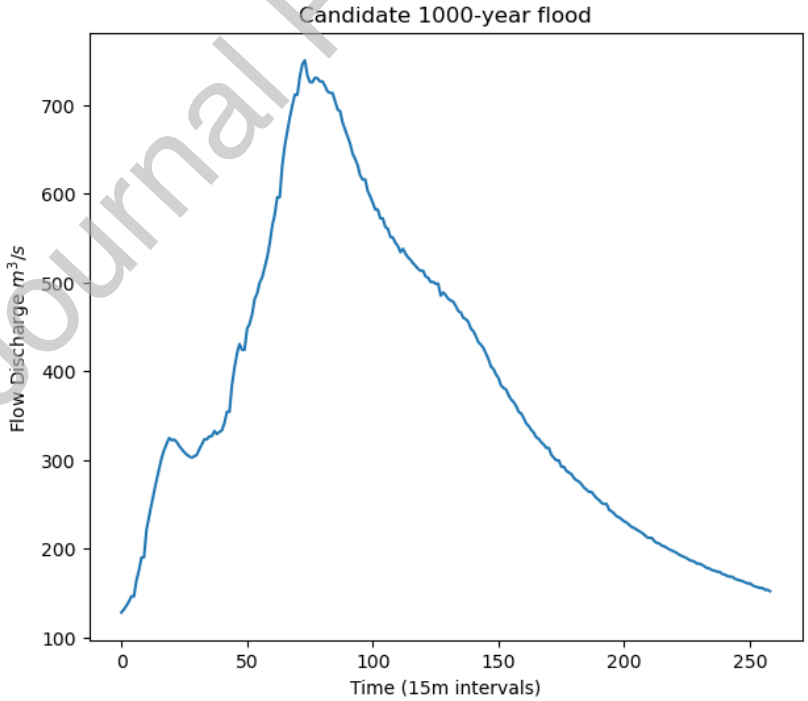

Fig. (21): Hypothetical 1000-year return period flood hydrograph.

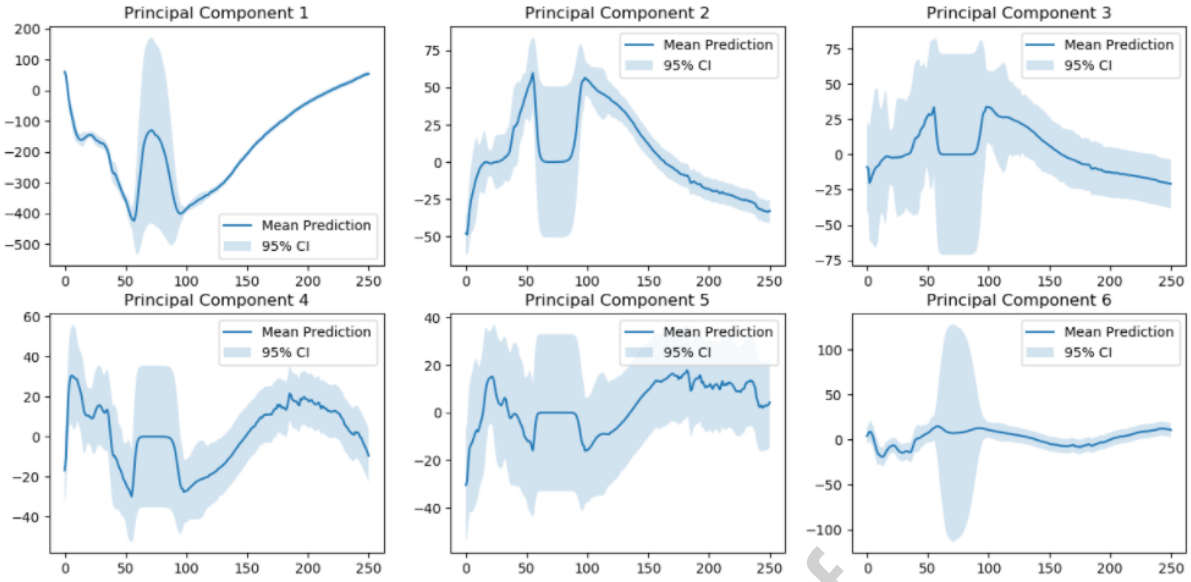

Fig. (22): Latent variable mean predictions with corresponding 95% confidence intervals.

Fig. (22) shows the GP's prediction for the 1000-year return period event in the latent feature space. Comparing the predictions made in this instance with those seen previously in Figs. (14) and (15), it can be observed that there is far greater uncertainty associated with these predictions as illustrated by the much wider confidence intervals, even among the first 3 principal components where accurate prediction was observed previously. The uncertainty observed in the latent feature space for this set of data can be associated with the fact that the emulator did not observe a training instance like the 1000-year return period hydrograph.

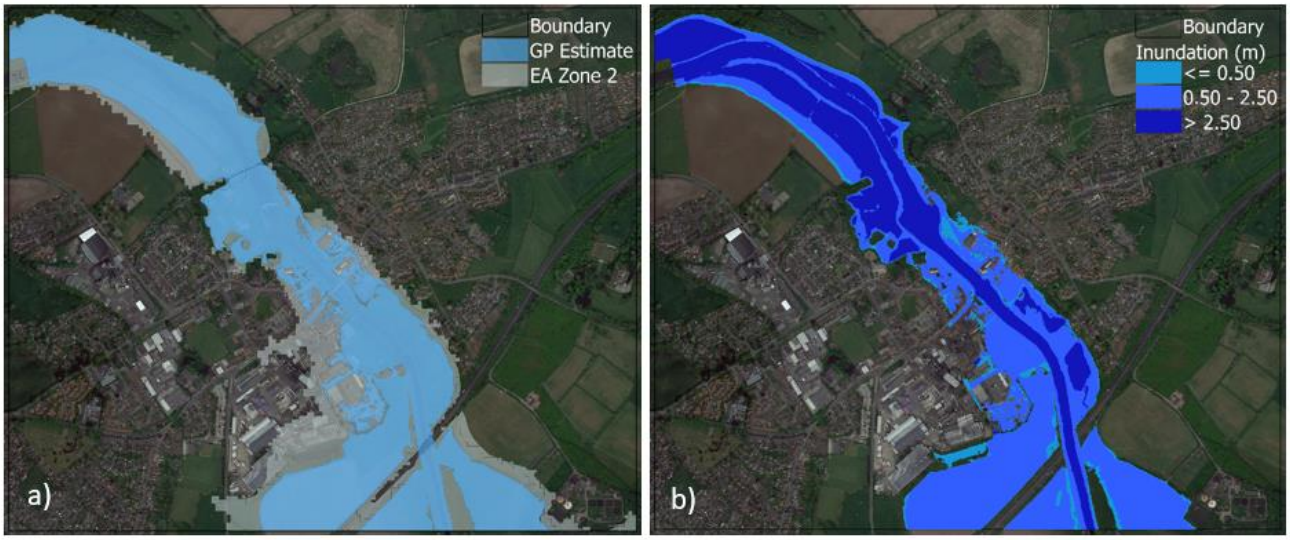

Fig. (23): (a) Blue illustrates inundation extent (>0.3m) estimated from GP Emulator applied to hypothetical 1000-year flood and grey illustrates EA Flood Zone 2, and (b) Inundation depths corresponding to estimated 1000-year flood event.

Fig. (23a) illustrates the inundation resulted from 1000-year flood event, with Fig. (23b) discretising this into sections of different inundation depths. The grey boundary illustrates the estimated area within the modelling domain that has a 0.1% or greater risk of flooding each year (as provided by the UK's Environment Agency). Overlayed in blue is the estimated inundated extent predicted from the GP model tested on the hydrograph described in Fig. (22). Different representations of the underlying processes or initial conditions of the model will propagate differences through the simulation and can result in varied outputs. For instance, the GP emulator models a flash-flood type scenario in which no previous inundation is present whereas the Environment Agency likely models on a longer timeframe and accounts for additional variables such as previous inundation and soil moisture content. However, there are also clear similarities and consistencies between these predictions which suggests that the proposed GP emulator model's predictions (and the estimation of a 1000-year flood) are in line with those of the Environment Agency.

The differences in maximum inundation between GP predictions at the upper and lower bounds of predicted 95% confidence intervals are analysed (Fig. 24). These intervals are calculated using the variance values predicted by the GP emulator in the latent feature space. Since GPs define a

distribution over function values at test inputs using a Gaussian framework, there is a mean prediction and corresponding variance estimates. For each test instance x_i^*, the upper and lower 95% confidence interval values are defined as:

$$CI = \mu(x_i^*) \pm 2\sigma(x_i^*) \#(24)$$

Fig. (24) show that most of the difference between inundation estimates is limited to less than 0.5m. However, in some sections, such as the river channel, inundation differences of greater than 1m between predictive interval bounds can be observed. These values can be considered to illustrate predictive uncertainty by the GP-based emulator.

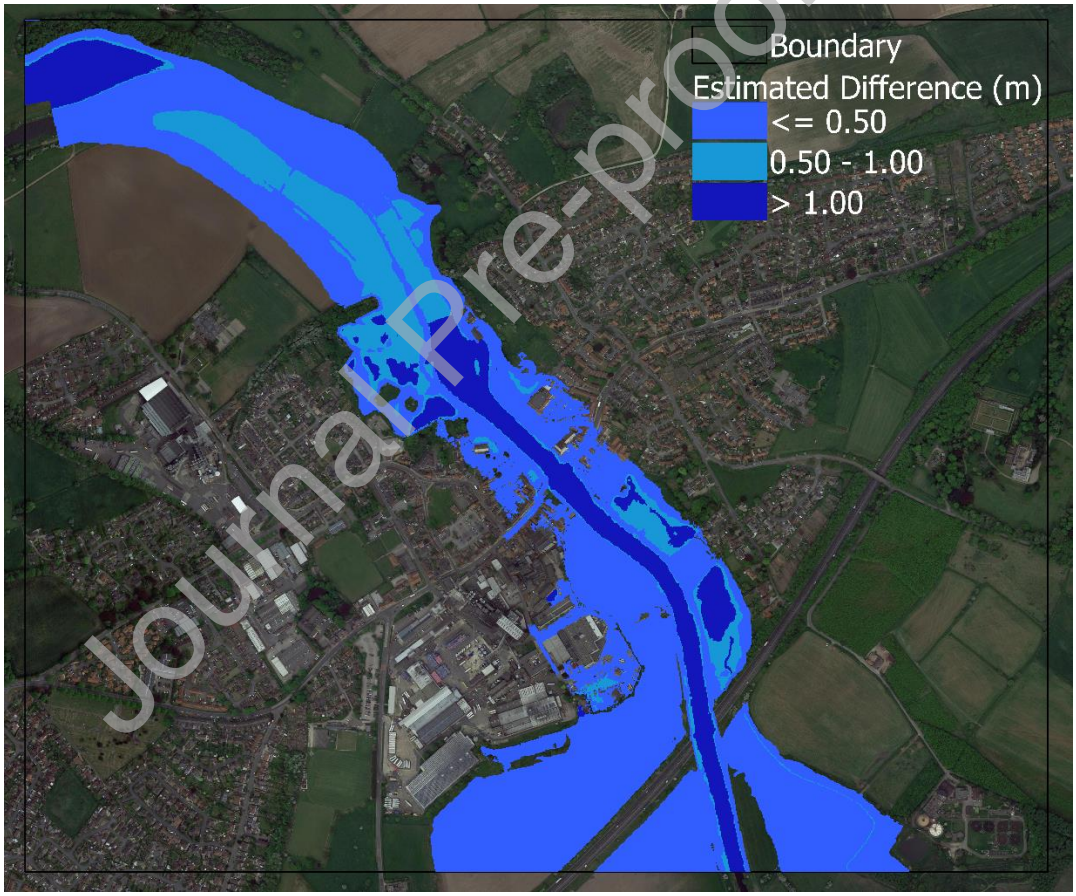

Fig. (24): Predicted inundation difference between upper and lower 95% confidence interval bounds.

6. Conclusions

A multi-output Gaussian Process-based statistical emulator was proposed for the emulation of high-dimensional outputs from a numerical flood simulator. The standard univariate GP regression methodology is extended to the multi-output case by producing a collection of univariate GPs to model each output feature independently. The GP model was developed to approximate the results of a coupled hydraulic flood model (i.e., LISFLOOD-FP), and significantly reduce the computational costs associated with numerical flood modelling at high spatiotemporal resolutions. To overcome the methodological issues surrounding the use of very high-dimensional data in GP modelling, a computationally efficient dimensionality reduction scheme was utilised.

The appropriateness and robustness of the proposed model is examined for a case study of a flood prone region in the UK. Topographical and hydrological data is collected and preprocessed to produce a realistic, high-resolution Digital Elevation Model and a collection of synthetic hydrographs to parameterise and run the LISFLOOD-FP simulations. Inundation data, in the form of water levels across the modelling domain over time, is generated from the numerical simulator which is then used to build and train the GP-based emulator.

The GP emulator was assessed against a CNN emulator by performing the cross-validation to ensure no over-fitting occurs in either model, and to assess each methodology's ability to generalise to out-of-sample tests. During these tests the performance of both the dimensionality reduction scheme and the GP-based emulator were found to be robust. The GP method outperformed the CNN emulator with respect to both regression and classification. Furthermore, the runtime for GP emulator was very significant reduced compared with the original numerical model and the CNN emulator.

Demonstrating the GP's high predictive accuracy, fast runtimes, and readily available quantification of predictive uncertainty, this study demonstrates the robustness and appropriateness of the proposed GP emulator as a more efficient method of probabilistically emulating complex physics-

based models over alternative approaches such as the CNN-based emulator. However, a statistical emulator is very sensitive to its experimental design. Choosing a representative sample of training data remains vital to building an emulator that can generalise well to out-of-sample tests. Therefore, the choice of training data should be carefully considered, and the training dataset should be as large (comprehensive) as possible, while considering the associated cost of collecting new training data.

It should be noted that each emulator is case-specific and so cannot be transferred to alternative scenarios in which different geographical areas are considered. Modelling a new scenario involves considerable further development time for the original simulator as well as the collection and processing of input data and simulation time. However, the GP emulator methodology proposed here can be easily translated to any scenario in which high-resolution spatiotemporal data is being generated from a highly complex model or processes (i.e., not just 2D inundation modelling). Furthermore, there are likely to be constraints to the spatiotemporal extent that emulators can effectively reproduce, especially with a GP methodology's computationally expensive training cost. However, this study demonstrated GP's high degree of accuracy for a very high spatial-resolution model with over 800,000 discrete cells.

The developmental cost of building emulators can be very high and therefore consideration should be given to whether an emulator is necessary in the first place. However, for cases where the underlying model's computational cost is prohibitive to real-time forecasting or for tasks requiring large numbers of simulations (i.e. scenario modelling), a GP-based emulator is an efficient modelling alternative finding a suitable balance between accuracy and complexity, and this study shows GPs to outperform competing methods of emulation.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Abolfathi, S., Yeganeh-Bakhtiari, A., Hamze-Ziabari, S. M., Borzooei, S., (2016). Wave runup prediction using M5′ model tree algorithm. Ocean Engineering, 112. Pp. 76-81. DOI:10.1016/j.oceaneng.2015.12.016

Abolfathi, S., Shudi, D., Borzooei, S., Yeganeh-Bakhtiari, A., & Pearson, J. (2018). Application of smoothed particle hydrodynamics in evaluating the performance of coastal retrofit structures. *Coastal Engineering Proceedings*, (36), 109-109.

Alvarez, M. A., Rosasco, L., & Lawrence, N. D. (2012). Kernels for vector-valued functions: A review. *Foundations and Trends® in Machine Learning*, *4*(3), 195-266.

BBC. Tadcaster bridge work starts after flood collapse. (2016). Available at https://www.bbc.co.uk/news/uk-england-york-north-yorkshire-35401832. Accessed on 09/11/2021.

Bates, P. D., & De Roo, A. P. J. (2000). A simple raster-based model for flood inundation simulation. *Journal of hydrology*, *236*(1-2), 54-77.

Bulti, D. T., & Abebe, B. G. (2020). A review of flood modeling methods for urban pluvial flood application. *Modeling earth systems and environment*, *6*(3), 1293-1302.

Chang, E. T., Strong, M., & Clayton, R. H. (2015). Bayesian sensitivity analysis of a cardiac cell model using a Gaussian process emulator. *PloS one*, *10*(6), e0130252.

Cheng, Z., Li, M., Jia, G., & Shi, Z. (2022). Adaptive Gaussian Process with PCA for prediction of complex dispersion relations for periodic structures. *European Journal of Mechanics-A/Solids*, *93*, 104547.

Conti, S., Gosling, J. P., Oakley, J. E., & O'Hagan, A. (2009). Gaussian process emulation of dynamic computer codes. *Biometrika*, *96*(3), 663-676.

Daneshkhah, A., & Bedford, T. (2013). Probabilistic sensitivity analysis of system availability using Gaussian processes. *Reliability Engineering & System Safety*, *112*, 82-93.

Dong, S., Salauddin, M., Abolfathi, S., Tan, Z. H., & Pearson, J. M. (2018). The influence of geometrical shape changes on wave overtopping: a laboratory and SPH numerical study. In *Coasts, Marine Structures and Breakwaters 2017: Realising the Potential* (pp. 1217-1226). ICE Publishing.

EA. Tadcaster Flood Alleviation Scheme (FAS) Information Page. Available at: https://consult.environment-agency.gov.uk/yorkshire/tadcaster-flood-alleviation-scheme/. Accessed on 09/11/2021.

Feng, X., Xie, Y., Song, M., Yu, W., & Tang, J. (2018, November). Fast randomized PCA for sparse data. In *Asian conference on machine learning* (pp. 710-725). PMLR.

Fewtrell, T. J., Duncan, A., Sampson, C. C., Neal, J. C., & Bates, P. D. (2011). Benchmarking urban flood models of varying complexity and scale using high resolution terrestrial LiDAR data. *Physics and Chemistry of the Earth, Parts A/B/C*, *36*(7-8), 281-291.

García-Alba, J., Bárcena, J. F., Ugarteburu, C., & García, A. (2019). Artificial neural networks as emulators of process-based models to analyse bathing water quality in estuaries. *Water research*, *150*, 283-295.

Ghanem, R., Higdon, D., & Owhadi, H. (Eds.). (2017). *Handbook of uncertainty quantification* (Vol. 6). New York: Springer.

Halko, N., Martinsson, P. G., & Tropp, J. A. (2009). Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions.

Kabir, S., Patidar, S., Xia, X., Liang, Q., Neal, J., & Pender, G. (2020). A deep convolutional neural network model for rapid prediction of fluvial flood inundation. *Journal of Hydrology*, *590*, 125481.

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

Lee, G., Kim, W., Oh, H., Youn, B. D., & Kim, N. H. (2019). Review of statistical model calibration and validation—from the perspective of uncertainty structures. *Structural and Multidisciplinary Optimization*, *60*(4), 1619-1644.

Lin, Q., Leandro, J., Wu, W., Bhola, P., & Disse, M. (2020). Prediction of maximum flood inundation extents with resilient backpropagation neural network: case study of Kulmbach. *Frontiers in Earth Science*, 332.

Longobardi, S., Lewalle, A., Coveney, S., Sjaastad, I., Espe, E. K., Louch, W. E., ... & Niederer, S. A. (2020). Predicting left ventricular contractile function via Gaussian process emulation in aortic-banded rats. *Philosophical Transactions of the Royal Society A*, *378*(2173), 20190334.

Massoud, E. C. (2019). Emulation of environmental models using polynomial chaos expansion. *Environmental Modelling & Software*, *111*, 421-431.

Morales-Hernández, M., Sharif, M. B., Kalyanapu, A., Ghafoor, S. K., Dullo, T. T., Gangrade, S., ... & Evans, K. J. (2021). TRITON: A Multi-GPU open source 2D hydrodynamic flood model. *Environmental Modelling & Software*, *141*, 105034.

Moreno-Rodenas, A. M., Bellos, V., Langeveld, J. G., & Clemens, F. H. (2018). A dynamic emulator for physically based flow simulators under varying rainfall and parametric conditions. *Water research*, *142*, 512-527.

Mosavi, A., Ozturk, P., & Chau, K. W. (2018). Flood prediction using machine learning models: Literature review. *Water*, *10*(11), 1536.

Neal, J., Dunne, T., Sampson, C., Smith, A., & Bates, P. (2018). Optimisation of the two-dimensional hydraulic model LISFOOD-FP for CPU architecture. *Environmental modelling & software*, *107*, 148-157.

O'Loughlin, F. E., Neal, J., Schumann, G. J. P., Beighley, E., & Bates, P. D. (2020). A LISFLOOD-FP hydraulic model of the middle reach of the Congo. *Journal of hydrology*, *580*, 124203.

O'Hagan, A. (2006). Bayesian analysis of computer code outputs: A tutorial. Reliability Engineering & System Safety, 91(10-11), 1290-1300.

Oakley, J. E., & O'Hagan, A. (2004). Probabilistic sensitivity analysis of complex models: a Bayesian approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *66*(3), 751-769.

Papaioannou, G., Efstratiadis, A., Vasiliades, L., Loukas, A., Papalexiou, S. M., Koukouvinos, A., ... & Kossieris, P. (2018). An operational method for flood directive implementation in ungauged urban areas. Hydrology, 5(2), 24.

Pinos, J., & Timbe, L. (2019). Performance assessment of two-dimensional hydraulic models for generation of flood inundation maps in mountain river basins. Water Science and Engineering. 12. 11-18. 10.1016/j.wse.2019.03.001.

Williams, C. K., & Rasmussen, C. E. (2006). *Gaussian processes for machine learning* (Vol. 2, No. 3, p. 4). Cambridge, MA: MIT press.

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

Sarri, A., Guillas, S., & Dias, F. (2012). Statistical emulation of a tsunami model for sensitivity analysis and uncertainty quantification. *Natural Hazards and Earth System Sciences*, *12*(6), 2003-2018.

Sharif, M. B., Ghafoor, S. K., Hines, T. M., Morales-Hernández, M., Evans, K. J., Kao, S. C., ... & Gangrade, S. (2020, June). Performance Evaluation of a Two-Dimensional Flood Model on Heterogeneous High-Performance Computing Architectures. In *Proceedings of the Platform for Advanced Scientific Computing Conference* (pp. 1-9).

Shaw, J., Kesserwani, G., Neal, J., Bates, P., & Sharifian, M. K. (2021). LISFLOOD-FP 8.0: the new discontinuous Galerkin shallow-water solver for multi-core CPUs and GPUs. *Geoscientific Model Development*, *14*(6), 3577-3602.

Shustikova, I., Domeneghetti, A., Neal, J. C., Bates, P., & Castellarin, A. (2019). Comparing 2D capabilities of HEC-RAS and LISFLOOD-FP on complex topography. *Hydrological Sciences Journal*, *64*(14), 1769-1782.

Soize, C. (2017). *Uncertainty quantification*. Springer International Publishing AG.

Teng, J., Jakeman, A. J., Vaze, J., Croke, B. F., Dutta, D., & Kim, S. J. E. M. (2017). Flood inundation modelling: A review of methods, recent advances and uncertainty analysis. *Environmental modelling & software*, *90*, 201-216.

Tran, G. T., Oliver, K. I., Holden, P. B., Edwards, N. R., Sóbester, A., & Challenor, P. (2019). Multi-level emulation of complex climate model responses to boundary forcing data. *Climate Dynamics*, *52*(3), 1505-1531.

Van Steenbergen, N., Ronsyn, J., & Willems, P. (2012). A non-parametric data-based approach for probabilistic flood forecasting in support of uncertainty communication. *Environmental Modelling & Software*, *33*, 92-105.

Wang, J., Balaprakash, P., & Kotamarthi, R. (2019). Fast domain-aware neural network emulation of a planetary boundary layer parameterization in a numerical weather forecast model. *Geoscientific Model Development*, *12*(10), 4261-4274.

Yan, X., Mohammadian, A., & Khelifa, A. (2021). Modeling spatial distribution of flow depth in fluvial systems using a hybrid two-dimensional hydraulic-multigene genetic programming approach. *Journal of Hydrology*, *600*, 126517.

Yang, J., Jakeman, A., Fang, G., & Chen, X. (2018). Uncertainty analysis of a semi-distributed hydrologic model based on a Gaussian Process emulator. *Environmental Modelling & Software*, *101*, 289-300.

Yang, S. N., & Chang, L. C. (2020). Regional inundation forecasting using machine learning techniques with the internet of things. *Water*, *12*(6), 1578.

Yeganeh-Bakhtiary, A., Houshangi, H., and Abolfathi, S. (2020). Lagrangian two-phase flow modeling of scour in front of vertical breakwater. Coastal Engineering Journal, 62:2, 252-266, DOI:10.1080/21664250.2020.1747140.

Zhou, Y., Wu, W., Nathan, R., & Wang, Q. J. (2021). A rapid flood inundation modelling framework using deep learning with spatial reduction and reconstruction. *Environmental Modelling & Software*, *143*, 105112.

Zischg, A. P., Mosimann, M., Bernet, D. B., & Röthlisberger, V. (2018). Validation of 2D flood models with insurance claims. *Journal of hydrology*, *557*, 350-361.

Graphical abstract