Analysis of the performance of a hybrid CPU/GPU 1D2D coupled model for real flood cases

Isabel Echeverribar, Mario Morales-Hernández, Pilar Brufau and Pilar García-Navarro

ABSTRACT

Coupled 1D2D models emerged as an efficient solution for a two-dimensional (2D) representation of the floodplain combined with a fast one-dimensional (1D) schematization of the main channel. At the same time, high-performance computing (HPC) has appeared as an efficient tool for model acceleration. In this work, a previously validated 1D2D CPU model is combined with an HPC technique for fast and accurate flood simulation. Due to the speed of 1D schemes, a hybrid CPU/GPU model that runs the 1D main channel on CPU and accelerates the 2D floodplain with a Graphics Processing Unit (GPU) is presented. Since the data transfer between sub-domains and devices (CPU/GPU) may be the main potential drawback of this architecture, the test cases are selected to carry out a careful time analysis. The results reveal the speed-up dependency on the 2D mesh, the event to be solved and the 1D discretization of the main channel. Additionally, special attention must be paid to the time step size computation shared between sub-models. In spite of the use of a hybrid CPU/GPU implementation, high speed-ups are accomplished in some cases.

Key words | coupled model, flood, hybrid GPU, shallow water, simulation

Isabel Echeverribar (corresponding author)
Pilar Brufau
Pilar García-Navarro
Fluid Mechanics, Universidad de Zaragoza,
LIFTEC-CSIC,
Zaragoza,
Spain
E-mail: *echeverribar@unizar.es*

Isabel Echeverribar
Hydronia Europe,
Madrid,
Spain

Mario Morales-Hernández
Computational Science and Engineering Division,
Oak Ridge National Laboratory,
Oak Ridge, TN 37830,
USA

HIGHLIGHTS

- A hybrid CPU/GPU coupled 1D2D model is presented.
- Details of the acceleration technique and code implementation are provided.
- A sensitivity analysis is performed in order to evaluate the model efficiency.
- Runtimes for each model and the transference are analysed for different hybrid configurations.
- The model is applied to a realistic case: a large stretch of the Ebro River.

INTRODUCTION

According to a European Environment Agency (EEA) survey (EEA), flood events are prone to occur more frequently in many river basins giving rise to catastrophic consequences. To control their impact on the environment and the society, management plans are being developed and flood modelling is starting to represent an important tool for flood risk maps design, as considered in the European Floods Directive (C. o. t. E. U. European Parliament 2007). However, flood forecasting is still not as widely used by decision-makers as it could be, not because of the quality of the results, but due to the speed of the simulations (Leskens *et al.* 2014).

The reduction of computational time when simulating real or practical cases has been one of the most important challenges of computational fluid dynamics. In particular, when trying to reproduce flood events on a large spatial and temporal domain, the efficiency of the method becomes crucial to make a computational tool affordable and with a

doi: 10.2166/hydro.2020.032

practical use. For this purpose, the most common strategies can be divided into two groups: model reduction and computation acceleration. The complexity of a real flow may require the consideration of the three-dimensional (3D) Navier–Stokes equations for the total representation of its nature. However, the prediction of hazards, such as flood events or evolution of pollutant discharges on rivers, does not need such a detail in many cases. Thus, the simplification of the model under shallow water conditions leads to the non-linear Shallow Water Equations (SWE), whose resolution is simpler and faster than the complete equations. Additionally, the assumption of dimensional hypothesis, which allows the modeller to work with one-dimensional (1D) models, also can accelerate the computations. However, the limits of these models must be always analysed in order not to lose accuracy in the solution (Horritt & Bates 2002; Costabile *et al.* 2015).

For flood event simulations, the two-dimensional (2D) shallow water system of equations has been demonstrated to be suitable enough to reproduce the flow behaviour (Knijff *et al.* 2010; Sanders *et al.* 2010; Masoero *et al.* 2013; Lacasta *et al.* 2014; Vacondio *et al.* 2016). Thus, although 3D models are widely extended on other CFD applications, their use for flood events still presents unaffordable times for large-scale events even when using parallelization techniques for acceleration. Additionally, their computational cost is not compensated by the accuracy increment in comparison with simplified models since huge flood events have a natural 2D behaviour (Horna-Muñoz & Constantinescu 2018). Thus, model reduction is a common strategy when the application allows for it. However, although the use of 2D models provides accurate results when dealing with floods due to the proper representation of the 2D velocity field, their computational cost may be still excessive, specially due to the necessity of small cells in the river bed or in hydraulic structures (Caviedes-Voullième *et al.* 2012; Echeverribar *et al.*). This drawback of 2D models has caused during years an extended use of 1D models for flood simulation (Petaccia *et al.* 2012; Murillo & Garcia-Navarro 2014). 1D models are advantageous when simulating complex channel nets and have been widely used by authorities, decision-makers and flood modellers due to their relatively short computational time. However, they have a clear limit in the simulation of 2D velocity fields and have been proved not to be accurate enough for flood events (Horritt & Bates 2002; Costabile *et al.* 2015; Morales-Hernández *et al.* 2018). Alternatively, some researchers have proposed 1D2D coupled models to combine a 2D representation on the floodplain and a 1D numerical schematization for the main channel (Bladé *et al.* 2012; Domeneghetti *et al.* 2013; Morales-Hernández *et al.* 2013, 2015). These models lead not only to a reduction of the simulation time but also to a more accurate representation of the river channel flow, whose main velocity direction and geometry are better captured by 1D models. Nevertheless, even though these combined models present a speed-up in comparison with full 2D models, when dealing with large-scale events they still present a high computational cost.

On the other hand, there exist other techniques which are focused not on the number of operations to compute depending on the model and its dimensions but on the speed to manage all those operations. Based on parallelization techniques that divide the computing workload into different cores that solve the scheme simultaneously, different technologies can be found. One of the most common strategies is Open Multi-Processing (OpenMP) (Board 2015), that accelerates the calculations depending on the number of cores, which depends on the number of CPUs available. Beyond this, high-performance computing (HPC) generally refers to the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation in order to solve large problems in science or engineering. In particular, schemes can be implemented to run on graphics cards (GPUs) (NVIDIA 2010), which may contain up to thousands of cores in the same device and work together with only one CPU. The implementation of 2D models on GPUs became extended few years ago and several 2D models running on Graphics Processing Unit (GPU) can be found. They offer computational speed-ups turning 2D models into affordable tools for flood forecasting (Castro *et al.* 2011; Brodtkorb *et al.* 2012; Lacasta *et al.* 2014; Petaccia *et al.* 2016; Vacondio *et al.* 2016). Additionally, if several GPU devices are used, the acceleration may be even higher. However, in addition to the large computing facilities required, the communication time penalty is not always worth it for a large number of GPUs, requiring efficient implementations (Morales-Hernandez *et al.*; Sharif *et al.* 2020; Turchetto *et al.* 2020).

Therefore, the combination of these two strategies – HPC and model dimension reduction – is proposed to increase even more the performance during a flood simulation. With the aim of carrying out this combination, the coupled 1D2D model proposed by Morales-Hernández *et al.* (2015) is adequately programmed for GPU devices, in the HPC context. In this model, previously validated running on CPU (Morales-Hernández *et al.* 2015), the main river bed is modelled so that the whole framework works as a pure 1D model when channel overflow does not occur (Morales-Hernández *et al.* 2016). During a flood event, the terrain adjacent to the river bed becomes inundated and a 2D model is used for the representation of the velocity field evolution on the floodplain. By means of an appropriate geometric link, the 1D and the 2D models are coupled. They are both based on the SWE and solved using a finite volume explicit upwind first-order numerical scheme with approximate Roe solvers that are able to deal correctly with wet/dry fronts, advance over dry beds and transient flows over irregular topography (Murillo *et al.* 2009; Morales-Hernández *et al.* 2015).

The computation of the combined 1D2D model consists of a combination of CPU and GPU algorithms and data transfers. The GPU programming procedures present non-trivial problems regarding the implementation efficiency of the coupled model. While the 2D model is running on the *device* (GPU), the cells for the 1D representation of the main channel run on the *host* (CPU). At this point, the computation of the coupled cell edges must be analysed to find the more efficient way of implementation. In this case, GPU computing was chosen also for these 1D2D shared edges.

The interest of the present analysis resides on the study of the combination of efficiency in the model and on the architecture of the parallelization, as the speed-up of the coupled model is not that obvious due to its particular hybrid implementation. This implementation is based on combining CPU and GPU algorithms for each of the subdomains and the necessity of data transfers. Having a 1D2D fully GPU model could be a possibility depending on the application. However, in the cases of interest in the present study, the fast 1D frameworks and the high involvement of the floodplain in flood events place this hybrid model into an interesting position to be studied for cases where the main river does not represent a large proportion of the computation. Therefore, the aim is to analyse not only computational times depending on the existing models but also the influence of the implementation and the data transfer within the global computational time.

Depending on what the model is compared to, this particular code architecture could lead to different speed-ups that are schematized in Figure 1. Speed-up 1 (SU1) has been previously analysed in several works where a 2D model has been implemented on GPU. It is worth recalling the dependency not only on the algorithm implementation but also on the GPU device used. For instance, Lacasta *et al.* (2014) report approximately constant values of 60x with an optimized code. With the same purpose, Vacondio carried out a deep analysis of simulation times for different devices obtaining speed-ups that vary from 10x up to 200x (Vacondio *et al.* 2014). On the other hand, speed-up 4 (SU4), as mentioned before, has also been studied in previous works of the present model showing speed-ups up to 30x (Morales-Hernández *et al.* 2015). It is relevant to point out that speed-ups 1 and 2 are strictly computational speed-ups. However, speed-ups 3 and 4 require model reduction, and therefore, the speed-up comes with the penalty of accuracy and complexity (dimensional in this case) of the solution. Thus, this work aims to study speed-up 2 (SU2), the comparison between coupled models running on CPU and hybrid CPU/GPU, and speed-up 3 (SU3), the comparison of the hybrid CPU/GPU coupled model with a pure 2D model running on GPU.

For this purpose, different test cases are carried out. First, a stretch of the Severn River (UK), proposed by the

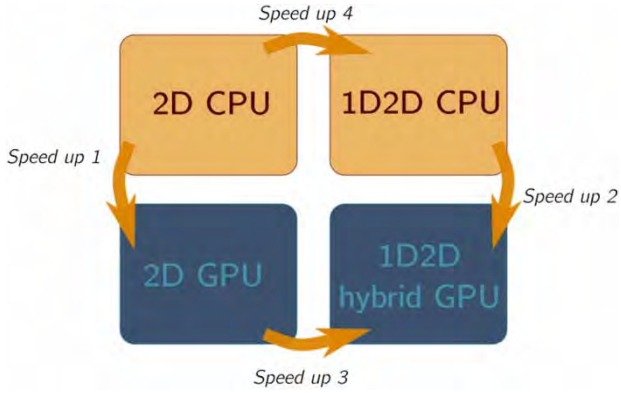

Figure 1 | Diagram sketching model comparisons.

UK Environmental Agency (U. E. Agency), is used as validation test to ensure the proper implementation of the 1D2D coupled model on the hybrid CPU–GPU paradigm, comparing the obtained results with other models on a fixed mesh size. Once the model has been tested, a benchmark case is designed to analyse the sensitivity of the model to the data transfers depending on parameters that are susceptible to change depending on the case, such as the number of 1D cells, the contribution of floodplain due to discharge peak of the event and duration of the hydrograph. Finally, the model is applied to a real test case: the Ebro River (Spain). The simulation of two historical events in the middle reach of the river, encompassing 125 km of river within a 400 km^2 area, is used to study the performance in real cases where the floodplain plays an important role and the presence of hydraulic structures involves higher resolution.

GOVERNING EQUATIONS AND NUMERICAL SCHEME

1D SWE

For the main channel, the 1D SWE are solved only involving the longitudinal velocity of the river (Cunge *et al.* 1980). The mass and momentum conservation laws are written for the channel longitudinal direction s as follows:

$$\frac{\partial U_{1D}(s,t)}{\partial t} + \frac{\partial F_{1D}(s,U)}{\partial s} = S_{1D}(s,U) \tag{1}$$

The vectors, which are derived in time, t, and in space, s, are

$$U_{1D} = \begin{pmatrix} A \\ Q \end{pmatrix}; \quad F_{1D} = \begin{pmatrix} Q \\ \frac{Q^2}{A} + gI_1 \end{pmatrix}; \quad S_{1D} = \begin{pmatrix} 0 \\ g[I_2 + A(S_0 - S_f)] \end{pmatrix}; \tag{2}$$

where Q (m^3/s) is the discharge, A (m^2) is the cross-section wetted area, and g (m/s^2) is the acceleration due to the gravity. The source term vector contains the bed variation term, S_0 (m/m), and the friction slope, S_f (m/m), where friction losses are represented by means of the Gauckler–Manning's roughness coefficient, n ($s/m^{1/3}$) (Henderson 1966; Chanson 2004), present on the empirical Manning formula that defines S_f (Morales-Hernández *et al.* 2015). Finally, I_1 (m^3) and I_2 (m^2) account for hydrostatic and longitudinal width variation pressure forces, respectively (Burguete & García-Navarro 2001).

2D SWE

For the floodplain, the 2D hyperbolic Shallow Water system of equations – also based on mass and momentum conservation (Cunge *et al.* 1980), taking into account the two directions of the flow for the momentum equations – is given by

$$\frac{\partial U_{2D}}{\partial t} + \frac{\partial F_{x2D}(x,y,U)}{\partial x} + \frac{\partial F_{y2D}(x,y,U)}{\partial y} = S_{2D}(U) \tag{3}$$

where

$$U_{2D} = \begin{pmatrix} h \\ hu \\ hv \end{pmatrix}, \quad F_{x2D} = \begin{pmatrix} hu \\ hu^2 + g\frac{h^2}{2} \\ huv \end{pmatrix}, \quad F_{y2D} = \begin{pmatrix} hv \\ huv \\ hv^2 + g\frac{h^2}{2} \end{pmatrix}, \quad S_{2D} = \begin{pmatrix} 0 \\ gh(S_{0x} - S_{fx}) \\ gh(S_{0y} - S_{fy}) \end{pmatrix} \tag{4}$$

where U_{2D} contains the 2D conserved variables of the problem: water depth, h (m), and the unit discharges in the x- and y-direction, hu (m^2/s) and hv (m^2/s), respectively. F_{x2D} and F_{y2D} represent the fluxes of those variables in the x- and y-direction. Finally, source terms are compacted in S_{2D} containing the friction and bed slopes, $S_f = (S_{fx}, S_{fy})$ and $S_0 = (S_{0x}, S_{0y})$, respectively, projected on the x- and y-direction (Morales-Hernández *et al.* 2015).

Numerical scheme

In order to solve both systems of equations, a first-order finite volume numerical scheme is used. Both models can

be expressed in a compact way as follows:

$$\frac{\partial U}{\partial t} + \vec{\nabla} E = S \tag{5}$$

where E stands for variable fluxes: F_{1D} in 1D and (F_{x2D}, F_{y2D}) in 2D. S represents a generic source terms vector for both models. If the Gauss' divergence theorem is applied in a computational cell i, where Equation (5) is integrated, the finite volume (FV) numerical scheme can be derived. The hyperbolic character of Equation (5) offers the possibility to calculate the real eigenvalues and eigenvectors of the Jacobian matrix. They are used to build the updating schemes.

For the 1D schematization (see Figure 2(a)), and according to Morales-Hernández *et al.* (2015), the expression for the 1D updating at each cell i is

$$U_i^{n+1} = U_i^n - \frac{\Delta t_{1D}}{\Delta s}\left[\left(\sum_{m=1}^{2} \tilde{\lambda}^{+}\tilde{\gamma}\tilde{e}\right)_{i-1/2}^{m} + \left(\sum_{m=1}^{2} \tilde{\lambda}^{-}\tilde{\gamma}\tilde{e}\right)_{i+1/2}^{m}\right]^{n} \tag{6}$$

where Δs is the 1D grid size, $\lambda_{i+1/2}^{\pm m}$ are the Jacobian eigenvalues (with $\pm$ superscripts denoting the upwind discretization), $e_{i+1/2}^{m}$ are the Jacobian eigenvectors and $\tilde{\gamma}_{i+1/2}^{m}$ are wave coefficients. Expression (6) represents the time updating each time step at each cell with the ingoing contributions, as depicted in Figure 2(a). Finally, since the numerical scheme has an explicit nature, the time step size must be limited by the CFL stability condition (LeVeque 1992):

$$\Delta t_{1D} = \mathrm{CFL} \min_{i,m}\left(\frac{\Delta s}{|\tilde{\lambda}_{i+1/2}^{m}|^{n}}\right), \quad 0 \le \mathrm{CFL} \le 1 \tag{7}$$

Analogously, the 2D upwind explicit scheme also updates the cells according to the in-going contributions of the fluxes and source terms of the neighbouring cells (see Figure 2(b)). However, since the flow is computed on a 2D framework, the expression uses as many contributions as edges per cell and the area of cell i, Ω_i, instead of Δs. If a generic edge between cells i and j is called k, the problem is projected onto k and the matrix eigenvectors basis. Finally, the updating equation in a triangular mesh is

$$U_i^{n+1} = U_i^n - \frac{\Delta t_{2D}}{\Omega_i}\sum_{k=1}^{NE}\sum_{m=1}^{3}\left[(\tilde{\lambda}^{-}\tilde{\gamma}\tilde{e})_k^m\, l_k\right]^n \tag{8}$$

where $m = 1, \ldots, 3$ stands for the number of eigenvectors (or variables) and k runs the number of involved neighbouring walls (with NE = 3 in the triangular case). Again, the time step has to be limited through the CFL condition:

$$\Delta t_{2D} = \mathrm{CFL} \min_{m,k}\left(\frac{\min(\chi_i, \chi_j)}{|\tilde{\lambda}_k^m|^n}\right), \quad 0 \le \mathrm{CFL} \le 1 \tag{9}$$

where k loops over all computational walls and χ_i is defined at each cell depending on the area, Ω_i, and the length of its p

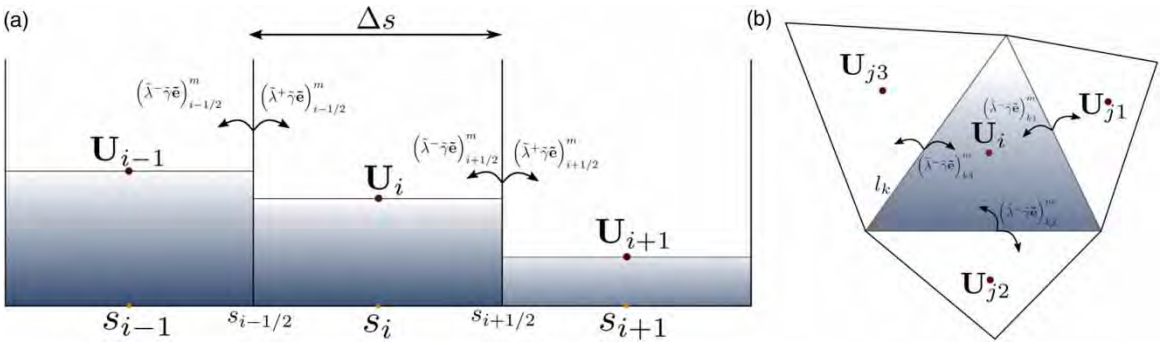

Figure 2 | Graphical representation of the 1D mesh (a) and 2D mesh (b) and the contributions at each cell edge.

neighbouring walls:

$$\chi_i = \frac{\Omega_i}{\max\limits_{p=1,\mathrm{NE}} l_p} \tag{10}$$

RCE COUPLED MODEL AND EXTENDED NUMERICAL SCHEME

Formulation of the RCE coupled model

The Riemann Coupled Edge (RCE) from Morales-Hernández *et al.* (2015) model is based on the definition of local Riemann problems between the 1D and the 2D models that will additionally contribute to the normal updating procedure of the cells in both 1D and 2D models, as represented in Figure 3. The coupled model is based on a new element of discretization: the coupling zone (CZ). Each CZ contains one 1D cell and an entire number of 2D cells that are laterally coupled at both left and right margins. It is important to note that at least one 2D cell must connect to a 1D cell. Some more details about the geometric linkage of the models can be found in Morales-Hernández *et al.* (2015).

The main idea of the RCE procedure is the conversion of the 1D values into 2D equivalent quantities, so the local Riemann problem at both sides of the CZ can be computed as an ordinary 2D problem. Therefore, additional contributions are computed in the lateral coupled edges (Morales-Hernández *et al.* 2015). According to the CZ angle, θ, the 1D cell values can be transformed into a 2D vector. In particular, the 1D model discharge at each cell, Q_{1D}, contributes to the 2D cells properly projected, allowing an unstructured coupling. The detailed procedure for the angle, θ, calculation is explained in Morales-Hernández *et al.* (2015), as well as the final expression for eigenvectors and values for the coupled edges.

At the end, each Riemann problem can be projected over the normal direction of the vector n_κ defined as the normal vector of the coupled edge, κ, between both models. Consequently, each coupled cell in both 1D and 2D models has additional contributions coming for the coupled part and Equations (6) and (8) are modified subsequently.

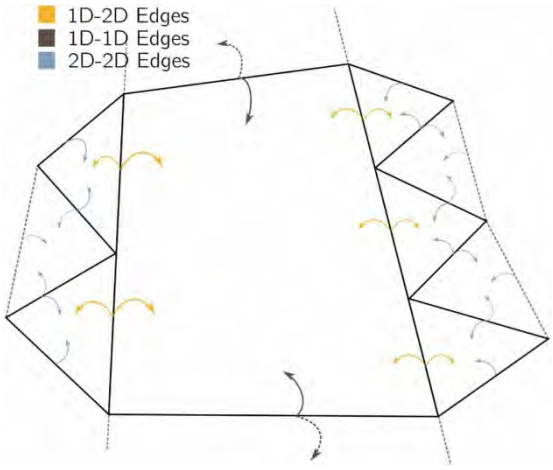

Figure 3 | Lateral coupling zone with different wave contributions depending on sub-domains interactions. Please refer to the online version of this paper to see this figure in colour: https://doi.org/10.2166/hydro.2020.032.

Extended numerical scheme

Expressions (6) and (8) update the cell values in the 1D and the 2D models. Since there is a CZ where both models coexist, all the cells that lie in the coupled framework must be updated not only with the ordinary in-going contributions coming from their own model but also with the contributions computed at the coupled edges. This is seen in Figure 3, where the 1D cell is not only updated with 1D contributions (dark brown) but also with the 1D2D edges (orange). However, before the final updating procedure, it is of crucial importance to set the same time step size for the three different steps: the coupled edges and the ordinary 1D and 2D cells. The chosen Δt is the minimum of the time step sizes of the three kinds of edges:

$$\Delta t = \min\left(\Delta t_{1D}, \Delta t_{2D}, \Delta t_{1D2D}\right) \tag{11}$$

where Δt_{1D2D} is the time step size given by the coupled edges contributions. Analogously to Equations (7) and (9), the time step restricted by the coupled edges is limited by the CFL condition and depends on the eigenvalues and

geometry:

$$\Delta t_{1D2D} = \mathrm{CFL} \min_{m,\kappa} \frac{\chi_\kappa}{|\tilde{\lambda}_\kappa^m|}, \quad \chi_\kappa = \min(\Delta s_{1D}, A_{2D}/l_\kappa) \tag{12}$$

where κ runs over the coupling edges and, thus, l_κ stands for the length of each coupled wall between the 1D cell, with size Δs_{1D}, and the 2D cell, with an area A_{2D}.

With the new Δt, the final numerical scheme updates all the cells in the domain with its new extended expressions, taking into account not only the normal cells but also the coupled contributions. This situation implies an additional term in the updating expressions (6) and (8), leading to

$$U_i^{n+1} = \underbrace{U_i^n - \frac{\Delta t}{\Omega_i} \sum_{k=1}^{\mathrm{NE}} \sum_{m=1}^{3} [(\tilde{\lambda}^- \tilde{\gamma} \tilde{e})_k^m l_k]^n}_{\text{2D–2D edges}} \underbrace{- \frac{\Delta t}{\Omega_i} \sum_{m=1}^{3} [(\tilde{\lambda}^- \tilde{\gamma} \tilde{e})_\kappa^m l_\kappa]^n}_{\text{1D–2D edges}} \tag{13}$$

for the 2D model. For the 1D model, a splitting of the two conserved variables is necessary due to their projection and treatment. For the wetted area, the final coupled expression is

$$A_i^{n+1} = A_i^n \underbrace{- \frac{\Delta t}{\Delta s} \left[\left(\sum_{m_1=1}^{2} \tilde{\lambda}^+ \tilde{\gamma} \tilde{e}_1 \right)_{i-1/2}^{m_1} + \left(\sum_{m_1=1}^{2} \tilde{\lambda}^- \tilde{\gamma} \tilde{e}_1 \right)_{i+1/2}^{m_1} \right]^n}_{\text{1D–1D edges}} \underbrace{- \frac{\Delta t}{\Delta s} \sum_{i=0}^{Nc} \sum_{m_2=1}^{3} [(\tilde{\lambda}^- \tilde{\gamma} \tilde{e}_1)_\kappa^{m_2} l_\kappa]^n}_{\text{1D–2D edges}} \tag{14}$$

where for the discharge:

$$Q_i^{n+1} = Q_i^n \underbrace{- \frac{\Delta t}{\Delta s} \left[\left(\sum_{m_1=1}^{2} \tilde{\lambda}^+ \tilde{\gamma} \tilde{e}_2 \right)_{i-1/2}^{m_1} + \left(\sum_{m_1=1}^{2} \tilde{\lambda}^- \tilde{\gamma} \tilde{e}_2 \right)_{i+1/2}^{m_1} \right]^n}_{\text{1D–1D edges}} \underbrace{- \frac{\Delta t}{\Delta s} \sum_{i=0}^{Nc} \sum_{m_2=1}^{3} [(\tilde{\lambda}^- \tilde{\gamma} \tilde{e}_2)_\kappa^{m_2} l_\kappa]^n \cos\theta - \frac{\Delta t}{\Delta s} \sum_{i=0}^{Nc} \sum_{m_2=1}^{3} [(\tilde{\lambda}^- \tilde{\gamma} \tilde{e}_2)_\kappa^{m_2} l_\kappa]^n \sin\theta}_{\text{1D–2D edges}}, \tag{15}$$

All the specific eigenvalues and eigenvectors, as well as other details of the coupled numerical scheme, can be found in Morales-Hernández *et al.* (2015).

HPC IMPLEMENTATION

Present-day engineering problems involve a high amount of data management and the necessity of fast results requiring HPC methods. In this work, the NVIDIA CUDA (Compute Unified Device Architecture) toolkit is used to carry out the simulations on a GPU-based solution.

GPU architecture and CUDA toolkit

The GPU was initially designed to deal with computer graphics operations. However, their development has turned these devices into a more general tool that can offer its capabilities for other purposes, such as the engineering problem acceleration. This approach is also known as GPGPU (General-purpose computing on Graphics Processing Units), and it allows the coder to implement algorithms that can run on a GPU hardware using high-level language. Particularly, NVIDIA developed the CUDA toolkit to run parallel solutions on GPU. It is a computing platform and programming model where the developer still programs in its familiar language (C, in this case) and incorporates extensions that express massive amounts of parallel operations.

It is important to mention that the GPU device has its own memory and this fact affects the way of algorithm implementation, since the location of the variables (CPU or GPU) must be always specified on the code and, usually, implies a double copy of the variables and a large amount of data transfer operations. Although some developments aim to make this transparent to the developer by means of a *unified memory*, the most common performance is based on memory copy operations. This point requires special attention due to two main implications. First, the performance of the GPU solution may be highly reduced if there is a large number of transfer in the code. Secondly, this GPU performance may not be appropriate for small applications where the transfer operations have a weight higher than calculations. Note that GPUs were initially oriented to perform

arithmetical operations on vector-based information. Because of this, original CPU codes used commonly have to be reorganized on these mentioned vector structures to implement them on CUDA.

Details of code implementation

The implementation on GPU is not a trivial task when dealing with combined models with different parts that lead to different performance efficiencies. It is clear that, in case of having a huge model including an entire river network where the number of 1D cells compares with the number of involved 2D cells, the 1D2D fully GPU could be justified since the global computational time would not be limited by the 2D floodplain, but by the river network. However, the focus here relies on the assumption that the number of cells in the 1D model is much lower than in the 2D model, and the number of computations in the latter is relatively large so as to require GPU acceleration. Then, the 1D framework is kept in its original CPU environment, avoiding the GPU porting. In particular, the original 1D code, as many CPU codes written in C, is organized using *Arrays of Structures*. Thus, the reorganization of the data into *Structures of Arrays* for the GPU implementation would be mandatory to get an acceptable performance on the GPU (Lacasta *et al.* 2014).

For all those reasons, in the present work, the 2D main loop and the coupled part are computed by the GPU, while the main channel, solved with a 1D model, is run on the CPU. The 2D structures (mesh and cells containing variables), as well as coupled structures (coupling zones, CZ), are mapped using *Structure of Arrays* and copied to the GPU at the beginning of the simulation. All the 1D structures (channels and cells with variables) remain on the CPU memory and are computed without this acceleration. Each time step, the results of the 1D models are transferred to the GPU to feed the CZ. This is sketched in Figure 4, where the operation sequence is outlined, distinguishing between the CPU and GPU codes.

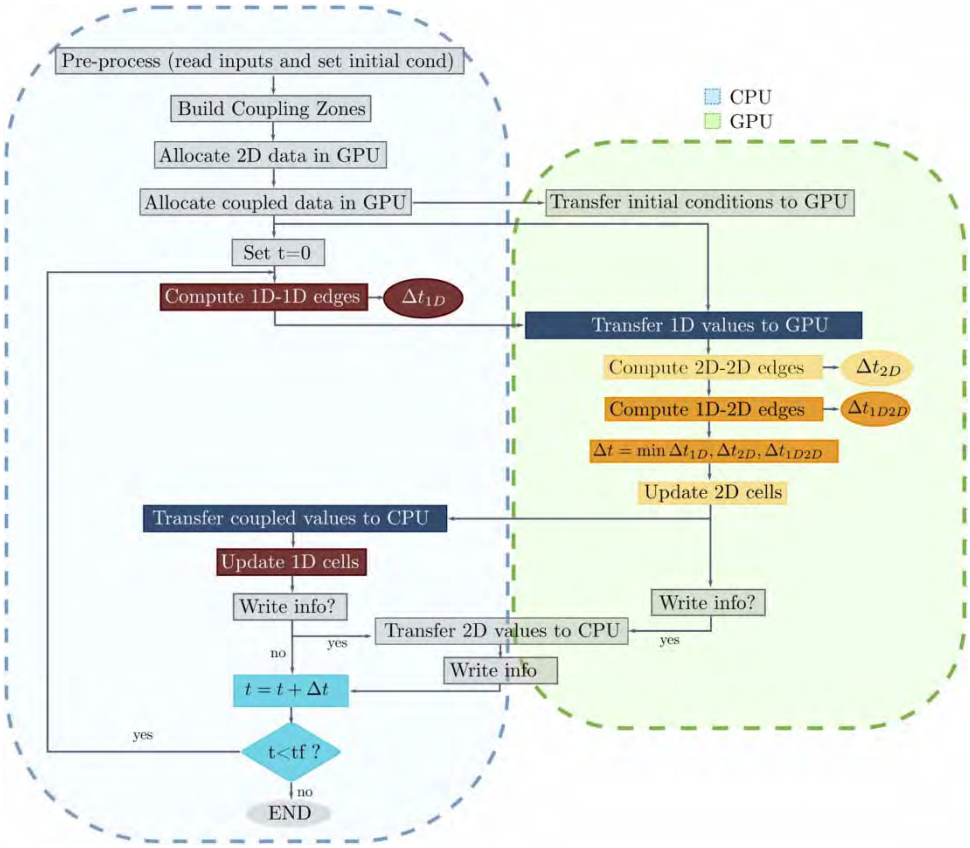

Figure 4 | Sequence diagram of the simulation process.

The functions of the algorithm have been divided into four groups that will be used to quantify the computational cost of each group and to carry out a detailed performance analysis. These four groups that can be seen in Figure 4 are:

- 1D–1D cell edges computation and 1D model update;
- 1D–2D cell edges computation;
- 2D–2D cell edges computation and 2D model update; and
- Data transfer between sub-domains.

The pre-process functions will not be taken into account later for time analysis. Different colours have been assigned to those transfer processes, sketched in Figure 4, to enable an easy graphical visualization in the plots presented in next sections.

SIMULATION CASES

In general, there is a noticeable speed-up when moving from a 2D model to a 1D2D model due to the substitution of the 2D refined river bed by 1D cross-sections. However, when a model moves on to the GPU, an additional speed-up analysis must be done due to the time-consuming data transfers. Specially, when there is a part of the code still running on the CPU, as the 1D model in this case, and data transfer between sub-domains is performed each time step.

The test cases of this section aim to demonstrate, first, the accuracy of the model. The method is applied for that purpose to a reach of the Severn River (UK), whose data are provided by the UK Environmental Agency with the aim of testing 1D2D models on a fixed size mesh. Secondly, the test cases intend to analyse the hybrid CPU/GPU coupled model performance regarding total simulation times and transfers. This is done with an idealized long river. Finally, it is significant to study the limits of the model concerning speed-up when it is applied to a real case where the 2D floodplain governs completely the simulation time. To do so, the model is applied to a long reach of the Ebro River (Spain).

Validation test case: the Severn River (UK)

The UK Environmental Agency (UK EA) (U. E. Agency) proposed different test cases for model validations. In this work, the 7th test case, a stretch of the Severn River (UK), is chosen to evaluate 1D2D model linking and exchanges of flood volume between the main channel and the floodplain running on GPU. As the intention was not to replicate an observed flood, upstream boundary conditions were designed by the UK EA, and the results of several models were provided to compare with.

This river consists of a 7 km long by 0.75–1.75 km wide floodplain, modelling a 20 km stretch of the Severn River. Boundary conditions are a hypothetical inflow hydrograph (see Figure 5(a)) and a downstream rating curve. An initial steady condition of 200 m^3/s is imposed. Several gauging points are spread over the domain to compare water level temporal evolution with other models. The location of these probes is also indicated in Figure 5(b) over the topography.

According to the benchmark test case rules, a 2D mesh for the floodplain has been built with 65,708 cells and linked with a 1D channel that contains 300 cells. In total, there are 858 coupling 2D cells that will need the information of the

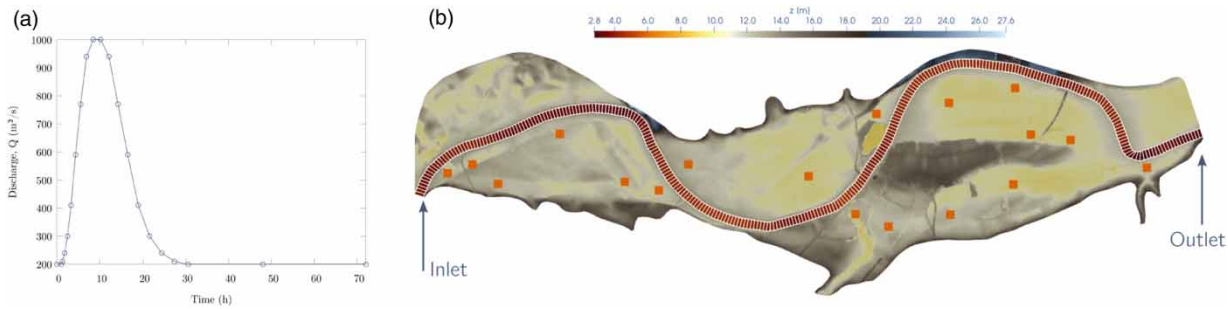

Figure 5 | Inlet hydrograph (a) and topography and probes location located with orange points (b). Please refer to the online version of this paper to see this figure in colour: https://doi.org/10.2166/hydro.2020.032.

1D cells to update their own variables every time step. Under these conditions, the GPU implementation is not so relevant due to the small efficiencies that these devices present for small cases. However, it might be worth mentioning that this 72 h flood event took 1,003.28 s for the model running on the GPU, whereas the CPU version lasted 16,903 s, getting a speed-up of 16.84x, due to the high influence of the 2D floodplain on the simulation.

There are three different areas with several measurement points on each. In Figure 6, the proposed model results are compared with the lower and upper envelope of other models (provided in U. E. Agency). Only one probe per floodplain/area has been chosen.

The accurate implementation of levees crest elevation is crucial for a proper representation of the volume transferred from the main channel to the floodplain. The original information contained in the 1D cross-sections regarding river embankments overlaps with the floodplain provided DEM and the modeller must make a choice. This decision involves different overtopping times unavoidably and, although this generates discrepancies between different models, the results are all coherent and similar. The model of this work also provides results in the proper range.

Sensitivity analysis case: long channel with a lateral floodplain area

The data transferred to the GPU due to coupled cells updating could involve a deceleration that may counteract the coupled model advantages. In order to test the sensitivity of this model to transfers, a test case has been designed with real river dimensions. The case involves a prismatic channel 86 km long, 8 m deep and 100 m wide; it has a surrounding plain flood-prone area with irregular microtopography (created as a combination of random *sin* and *cos* functions); and it has a soft longitudinal slope ($S_0 = -0.0005716$ m/m). The whole spatial domain encompasses 960 km^2. The performance analysis has been done focusing on different parameters: (a) number of 1D cells on the river bed (thus, larger amount of data transfer) and (b) simulated hydrograph (thus, different floodplain areas to be computed on the 2D floodplain and different durations). With reference to Figure 1 in the introduction, SU2 and SU3 will be compared.

For the purpose of this comparison, two different effects are to be analysed. First, the speed-up decrease due to the data transfer between devices (CPU/GPU) and the influence of the CPU part of the code. Second, the differences on speed-up depending on the percentage of floodplain involved (wet). Figure 7 shows a zoom view of several meshes that have been designed for the study. There is one mesh for the 2D model with a refined riverbed that acts as reference (a), and several coupled meshes that vary the discretization refinement on the main channel from 100 (b) to 2,000 1D cells (f), maintaining the same resolution on the flood-prone area, so that this parameter does not affect the results. Considering the necessity of the model to have at least one 2D cell linked to each 1D cell, a finer 1D discretization is not considered since it would lead to an extra increase of the 2D resolution unavoidably. The full 2D mesh has been refined at the river bed for a proper bathymetry representation. Therefore, while the coupled mesh contains 207,317 triangular elements within the floodplain, the full 2D mesh has 454,159 triangular cells in the whole

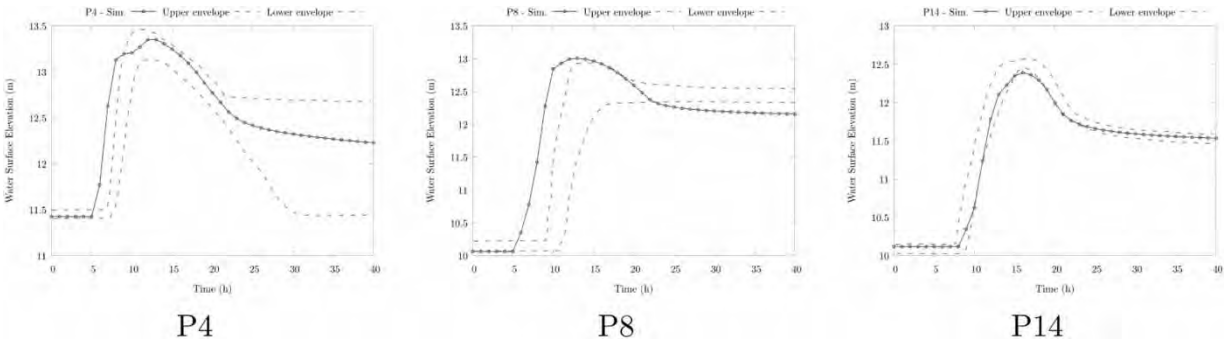

Figure 6 | Temporal evolution of simulated water surface elevation (dark blue) in comparison with the upper and lower envelope of results of other models (dashed grey line) at P4, P8 and P14. Please refer to the online version of this paper to see this figure in colour: https://doi.org/10.2166/hydro.2020.032.

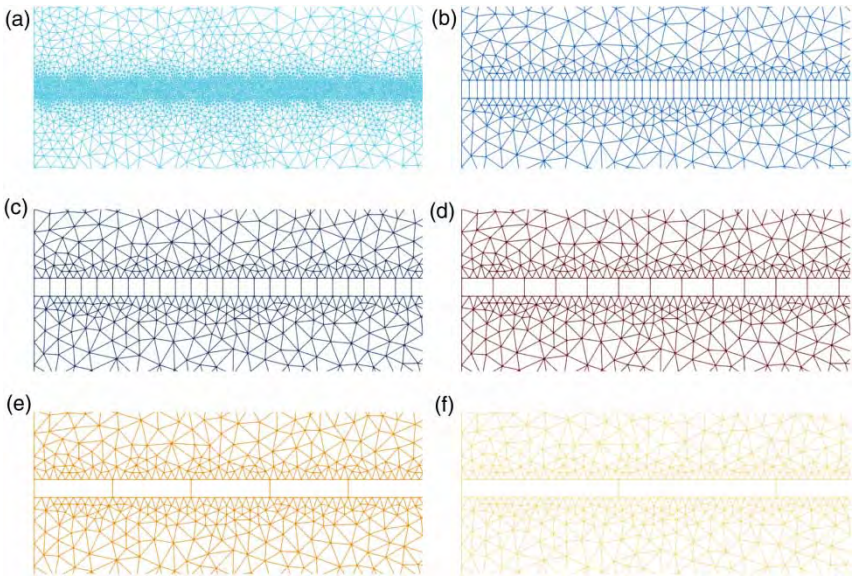

Figure 7 | Different meshes designed for the case: (a) full 2D, and coupled meshes with (b) 100, (c) 200, (d) 500, (e) 1,000 and (f) 2,000 1D cells.

domain. Over these meshes, four hydrographs have been propagated. All of them can be seen in Figure 8(a), where the maximum flooded area generated by the highest peak (H1 or H2) and the lowest (H3 or H4) of them is also shown in Figure 8(b).

For the computational time analysis of the hybrid model, four chronometers have been set on the code in order to do a profiling study distinguishing between four different costs: 1D cells computation, 2D cells computation, coupled areas computation and data transfer between sub-domains (1D/2D) and, thus, devices (CPU/GPU). For this analysis, an NVIDIA GeForce GTX Titan Black has been used as a *device* in combination with an Intel Core i7-8700 as a *host*. Figure 9 shows, for each hydrograph, the profiling for each number of 1D cells. All the colours correspond to the colour code stated in the flowchart depicted in Figure 4.

In Figure 9, where the computational time is divided into the aforementioned groups, the remaining time consumption is gathered under the 'other' flag, so the whole column represents the total simulation time. The five different columns correspond to different number of 1D cells on the river bed. All the graphs show how the 2D computation consumes the vast majority of the simulation time, followed by the coupled computation and the 1D edges. The transfer time between sub-domains, although not negligible, does not govern the total computational time even in the most unfavourable case (2,000 1D cells). However, it can be seen how the 1D time entails a potential bottleneck as the resolution (the number of cells) increases.

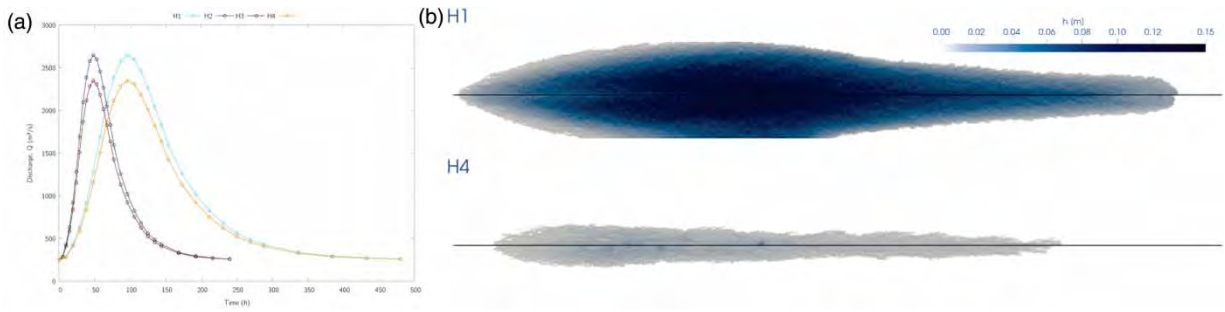

Figure 8 | Different hydrographs (a) and the maximum flooded area generated by H1 (b) upper and H4 (b) lower.

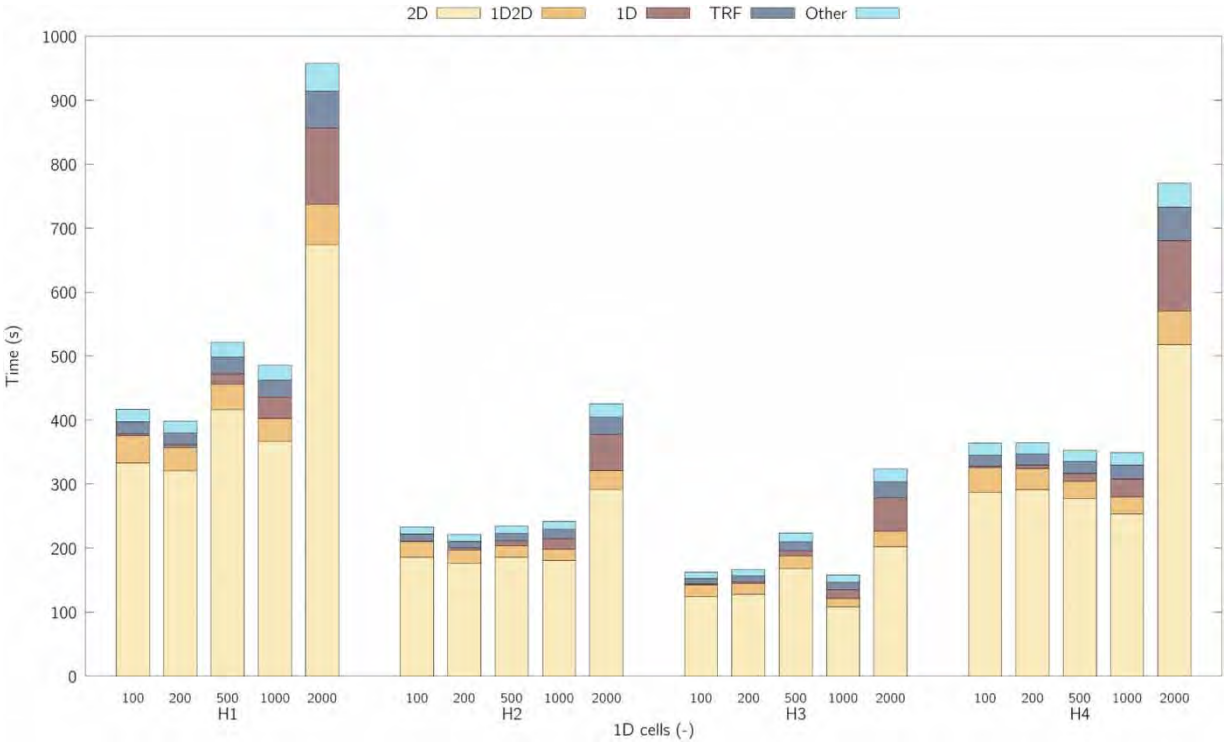

Figure 9 | Simulation times for different hydrographs (H1, H2, H3 and H4) and for different number of 1D cells on river bed running in CPU/GPU.

As depicted in Figure 4, the 1D domain is computed on the CPU and transferred from *host to device* in order that the coupled edges can be computed on the GPU with the main channel information. Thus, when increasing the number of 1D cells, the theoretical tendency should be an increase of 1D time and an increase of transfer time, while 2D and coupled edges times should remain approximately constant. This trend can be seen in Figure 10, where the normalized computational times show the relative importance of each part of the algorithm and the percentage of the 1D model increases with the number of cells, whereas the transfer time, directly proportional to the number of 1D cells, remains with the same importance. However, some exceptions are worth mentioning.

The coupled model presented needs a homogenization of the time step sizes between all the sub-domains (see Figure 4). Thus, the most restrictive mesh and flow condition governs the simulation time. As Δs_{1D} is usually higher than 2D cell edge sizes, the floodplain usually governs the global Δt. However, some extreme cases can occur when too many 1D cells are used and their size decreases, leading to a restriction imposed by the 1D model that slows down the simulation time. This is specially seen in the case with 2,000 1D cells on the main channel. It is worth mentioning that this fact affects only the 2D model simulation time and has nothing to do with the model being implemented on GPU in this hybrid form, but rather with the generic algorithm for the 1D2D model.

When this model runs on CPU, the transfer time does not exist and the percentage of the total simulation time consumed by the 2D floodplain is higher. This is shown in Figure 11, where the same itemization of the different computational times already seen for GPU is depicted for the model running purely on CPU. Note that the 1D channel consumes the same time in both the hybrid and the CPU models. The difference resides on the 2D model and the importance of the data transfer. The relevance of each part can be seen in Figure 12, where normalized times are shown. The 2D model takes the highest percentage of the total simulation times at every case, although the finer 1D resolution (2,000 cells) gets a significant increase in the total runtime.

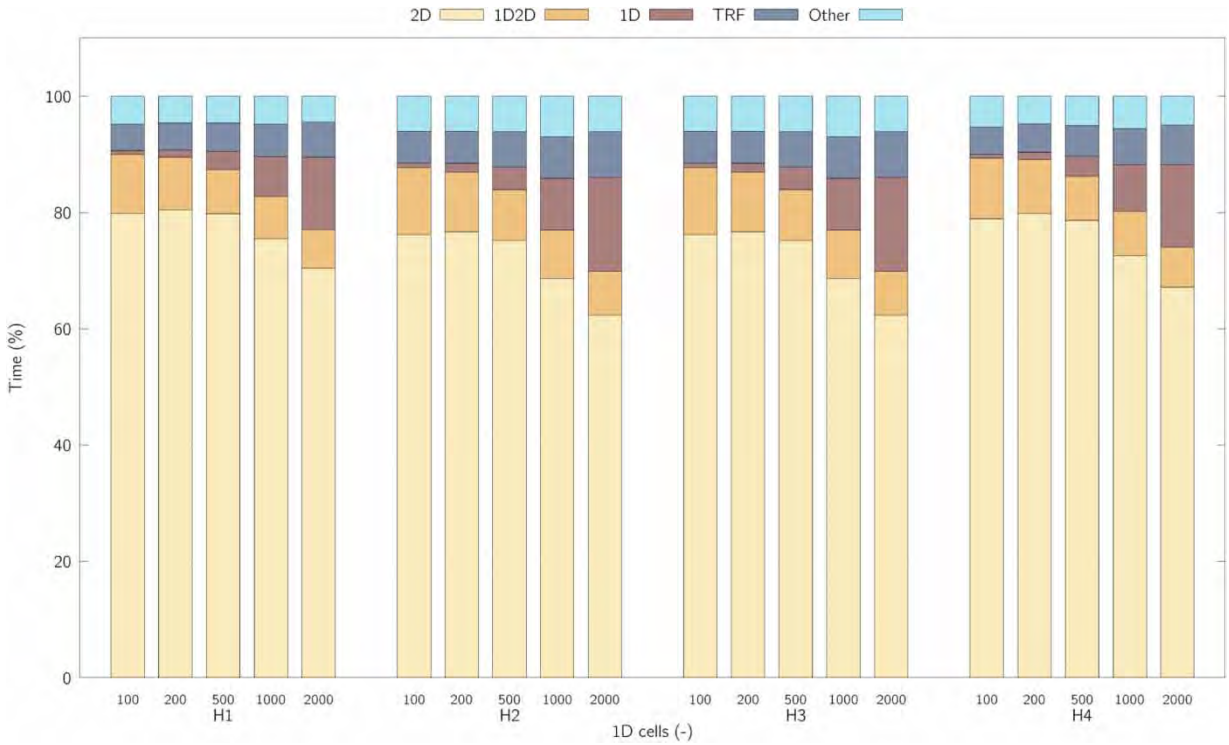

Figure 10 | Normalized simulation times for different hydrographs (H1, H2, H3 and H4) and for different number of 1D cells on river bed running in CPU/GPU.

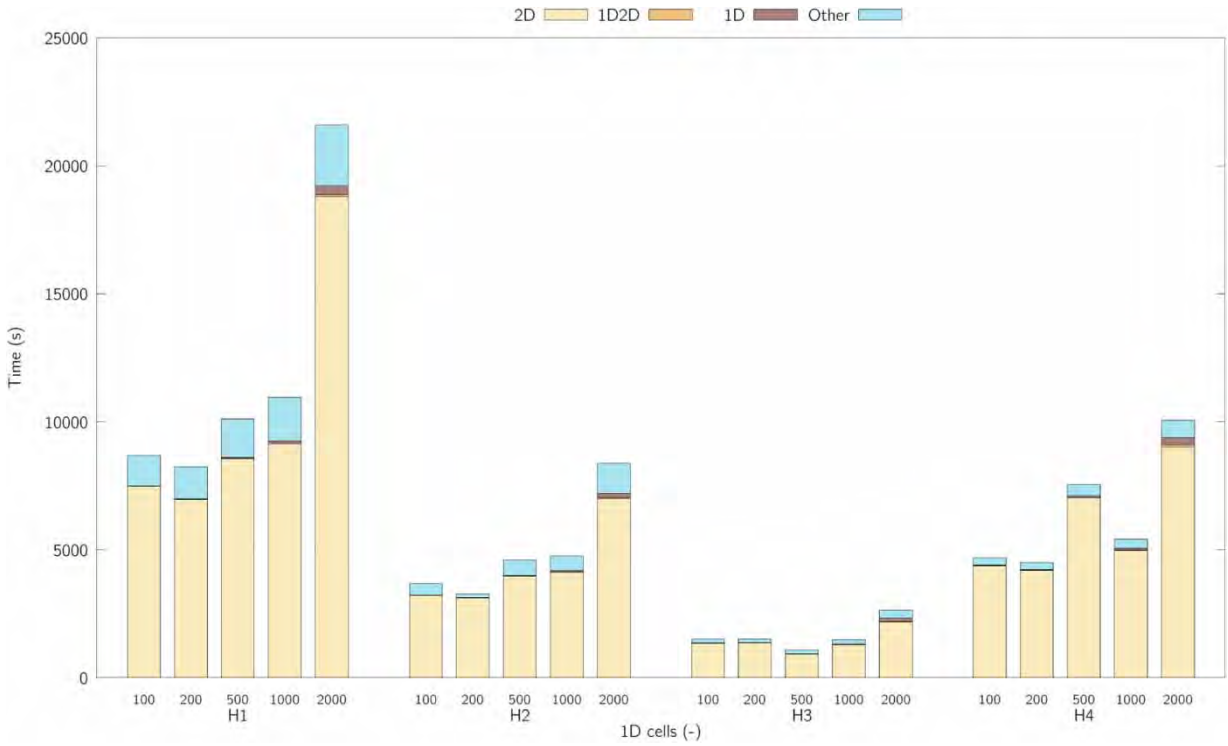

Figure 11 | Simulation times for different hydrographs (H1, H2, H3 and H4) and for different number of 1D cells on river bed running on CPU.

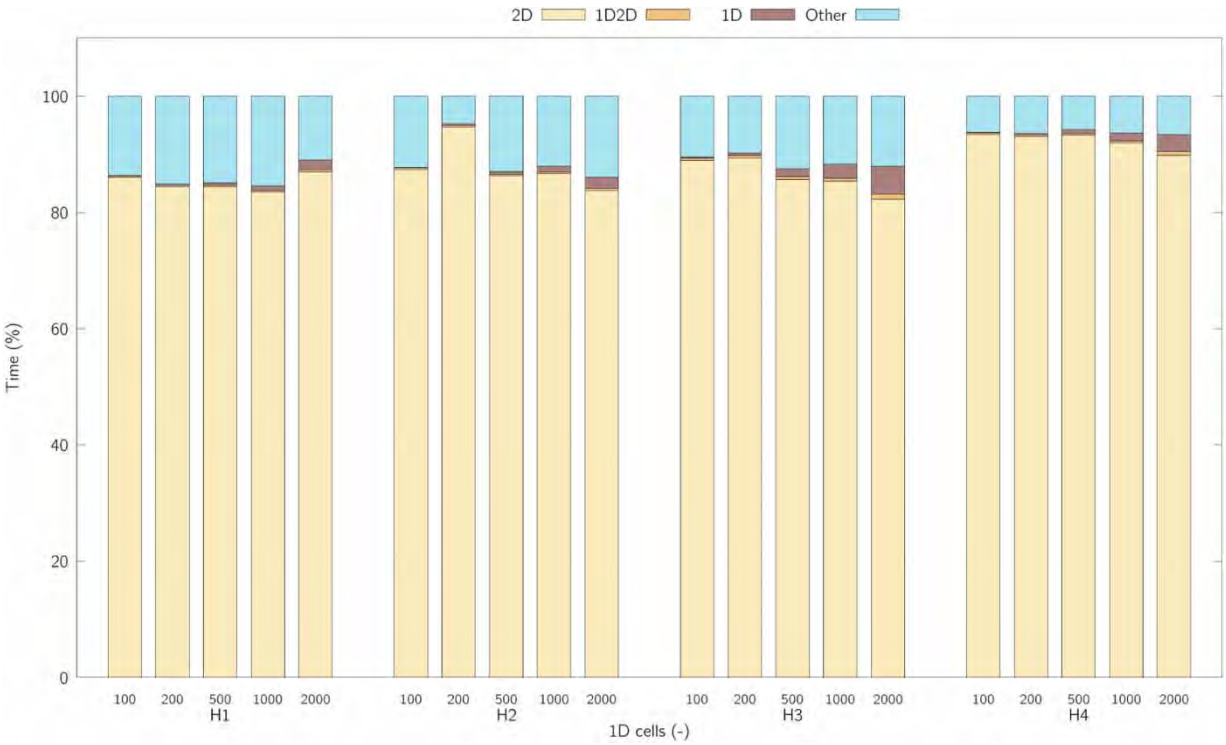

Figure 12 | Normalized simulation times for different hydrographs (H1, H2, H3 and H4) and for different number of 1D cells on river bed running on CPU.

On the other hand, when a full 2D framework is used transfer time is also missing since the whole model runs on the GPU. In this case, the small cells at the main channel require a computational time much higher than the transfer time and the 1D time together so the speed-up of the coupled model is still worthwhile. Besides, in this case, the main channel is specially time consuming in comparison with the floodplain since it contains the smallest cells (see Figure 7). The global speed-up accomplished by the hybrid coupled model in comparison with the full 2D is the SU3, sketched at the introduction in Figure 1. According to this figure, SU2 is the comparison of the hybrid model with the full CPU coupled model. Both speed-ups are represented for each hydrograph in Figure 13 (H1 to H4, from upper left to lower right), and for the number of 1D cells in the main channel.

In this case, it is clearly seen that the main channel has a strong influence in the results, provoking a higher speed-up when the river bed is substituted by a 1D model (speed-up 3). For this reason, when the model is compared with its CPU equivalent (speed-up 2), the acceleration is not that high.

Real test case: the Ebro River (Spain)

In this section, the coupled model is applied to a 125 km long stretch of the Ebro River (Spain) that encompasses a surface of 392 km^2. Two different historical events are propagated over the mesh and the same speed-up analysis is carried out to study the performance of the model under realistic conditions with complex and non-homogeneous floodplains. Figure 14(a) shows the upstream hydrographs for both events of different duration and peak discharge. The shortest, E1, corresponds to a 2010 flood event that lasted 5 days reaching 2,000 m^3/s. The second one, E2, stands for the 2015 event that lasted 21 days and, containing two peaks, reached more than 2,500 m^3/s.

The previous case has revealed how the 2D model may govern the flow, specially when the floodplain gets involved to a great extent. This effect can be seen even more dramatically if the floodplain contains hydraulic structures, as levees, that require a mesh refinement in order to capture their crest properly. This is the particularity of the present case, as can be seen in Figure 14(b), where a zoom view

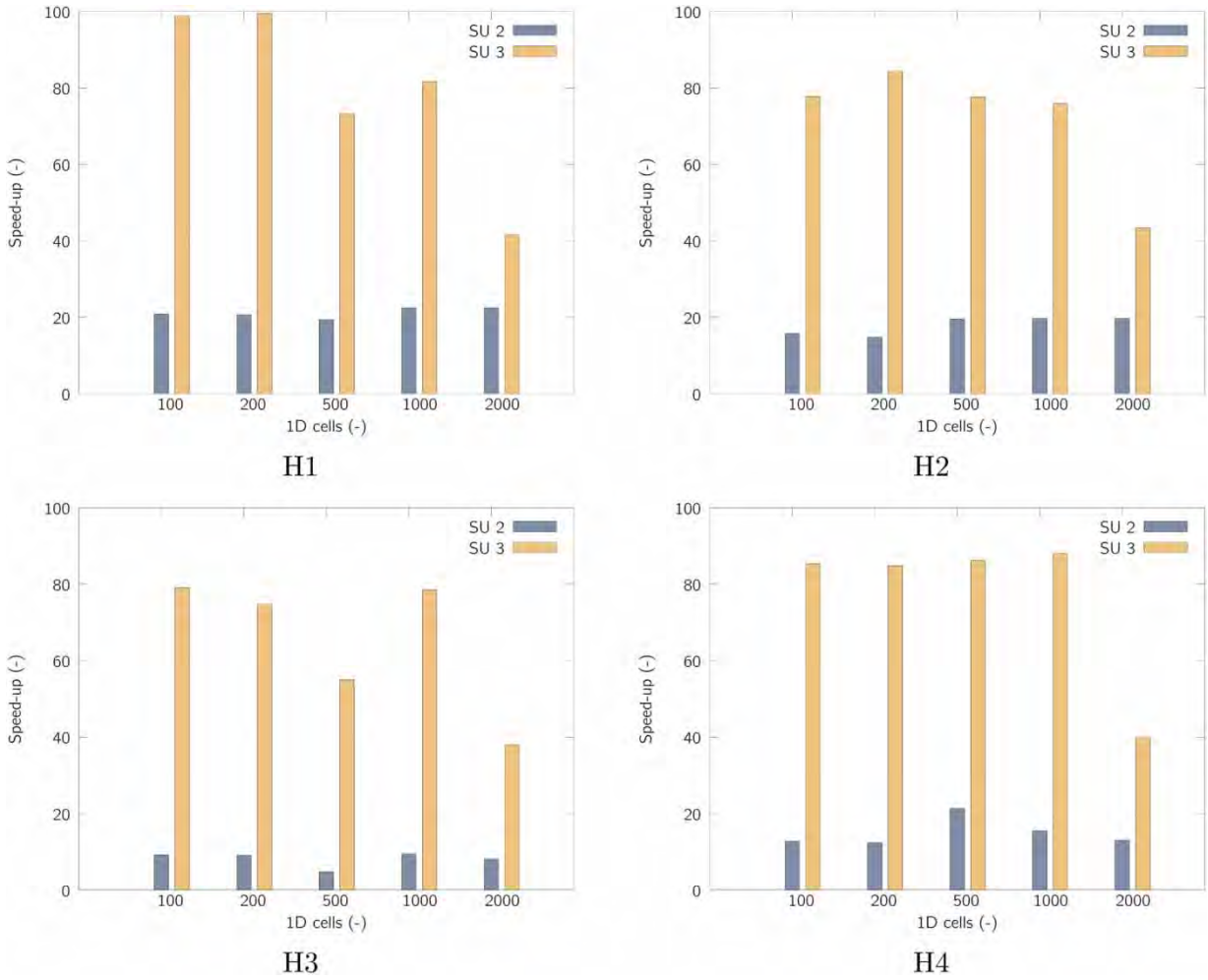

Figure 13 | Speed-up 2 (dark blue) and 3 (yellow) for different hydrographs at the synthetic long straight river case. Please refer to the online version of this paper to see this figure in colour: https://doi.org/10.2166/hydro.2020.032.

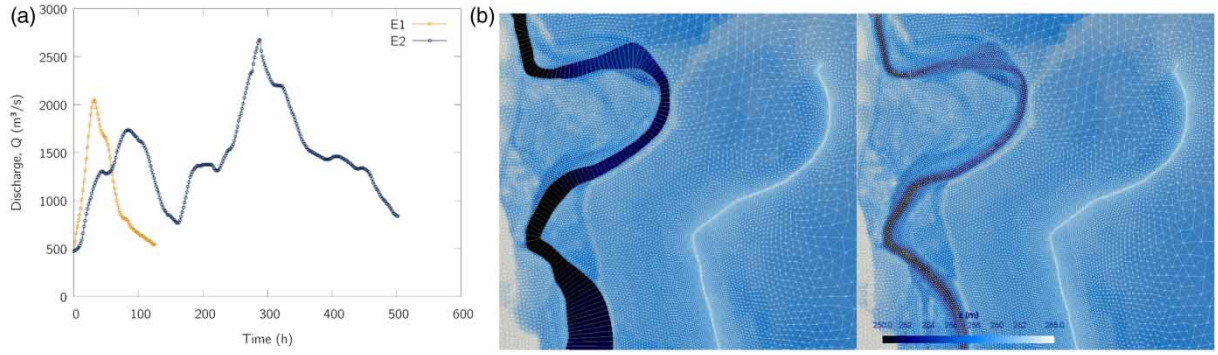

Figure 14 | Inflow hydrographs used as an inlet boundary condition in the Ebro River (a) and a small part of the domain discretized by 1D2D ((b) left) and 2D ((b) right) cells.

allows to see the comparison of the coupled 1D2D mesh and the full 2D mesh is shown. The figure reveals how the presence of narrow levees between field crops forces the 2D mesh to contain cells much smaller even than those in the main channel and this increases the computational time (Echeverribar *et al.*).

The geometry of the case also determines the proper number of 1D cells. In this case, although the flood wave is of low frequency and not too many 1D cells are necessary to reproduce proper transport velocities, the curvature of the river forces the necessity of small values of Δs within the 1D framework (also seen in Figure 14(b)), leading to 2,201 cells on the river bed.

Figure 15 shows the computational time divided by operations for the two different hydrographs in the Ebro River using the coupled model. For the sake of clarity, both normalized (b) and absolute (a) times are shown. Obviously, the E2 event consumes more time due to its own hydrograph duration. However, normalized times show that the trends are the same in both cases. The transfer time is not so predominant and the floodplain and coupled edges (all computed on GPU) have a strong influence on global time. Finally, due to the extent of the flooded area where small cells are required to represent the levees and the high consumption of the 1D model, simulation times are higher than it could be expected according with the idealized case previously simulated.

Table 1 | Simulation times for flood events in the Ebro River depending on the model and hydrograph

Model	E1	E2
1D2D	12 h	2.4 h
Full 2D	15 h	3.41 h
Speed-up	1.25x	1.42x

Table 1 shows the computational times of the coupled model in comparison with the full 2D model. Unlike the previous test case, the presence of small cells in the floodplain and their high influence, combined with a high number of 1D cells in the coupled model, causes a lower speed-up.

CONCLUSIONS

Even though 2D river flow models have become widely extended due to the detailed view of the flooding pattern that they offer, their calculation on large areas still remains excessively time consuming. Coupled 1D2D models emerged in the past as an alternative to accelerate the computation in river flood simulation cases, avoiding the use of a large number of 2D cells in the main channel and leading to a reduction of the total computational time. On the other hand, nowadays, 2D models have been improved in terms of computational efficiency thanks to HPC. The proposed work has merged these two strategies seeking an even greater speed-up.

For the analysis presented in this work, the 2D model runs on GPU and the 1D model is solved on the CPU. The computational performance of the coupled model has been analysed. From the examples presented, it can be concluded that, even with the additional data transfer time, there exists a positive speed-up with respect to the full CPU 1D2D coupled model and the full 2D GPU model in all cases. However, a 1D2D fully GPU could be justified in

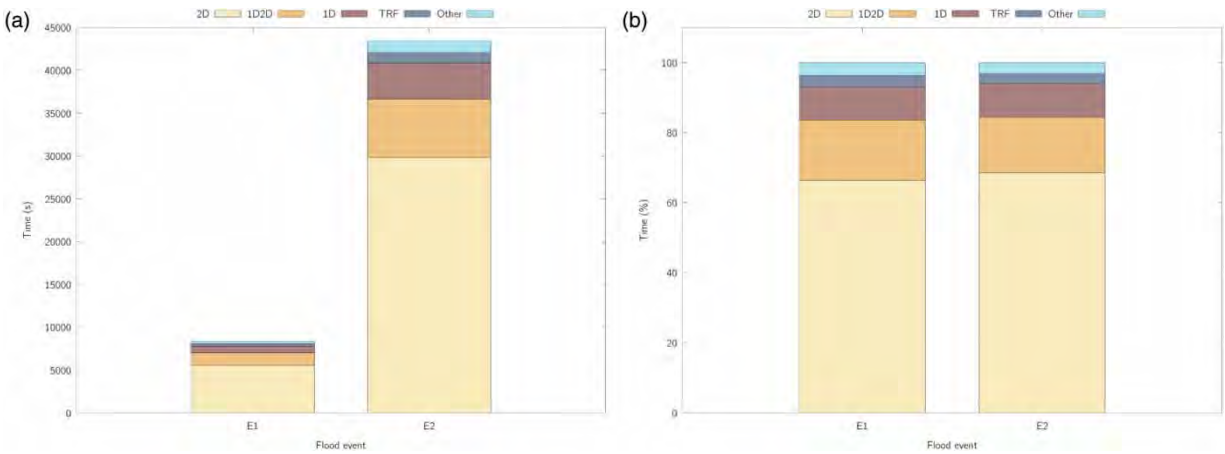

Figure 15 | Simulation times in absolute terms (a) and normalized (b) for different events (E1 and E2) in the Ebro River with the hybrid CPU/GPU model divided by operation.

case of having a number of 1D cells that compares with the number of involved 2D cells.

The sensitivity analysis has shown the influence of different parameters of the model on the results. First, the number of elements within the 1D framework implies a big difference on the computational time of the main channel, reaching values with the highest number of cells (2,000) that are around ten times higher than the most advantageous case (100 cells). However, this time does not govern the total computational time. At this point, it is worth recalling the linking strategy of the model in which each 1D cell requires at least one 2D cell linked and, thus, a higher resolution on the main channel would lead to a finer 2D mesh. Therefore, it would always be more time consuming than the 1D framework itself. Additionally, even with a large amount of 1D cells on the main river, the number of operations required by the 1D2D model is always lower than the number of operations on a refined 2D main channel that requires a high amount of small 2D cells in order to properly represent an irregular bathymetry of a river. There can be an effect of speed-up reduction if Δt_{1D} governs the flow, slowing down the floodplain calculation and, as a consequence, the global time. Although, even in this case, the speed-up in comparison with the full 2D GPU model is worthwhile, it could be a matter of future research to explore the possibility of not exchanging information between models at every time step. However, it must be taken into consideration that both models share the same characteristic time scale and magnitude of wave propagation celerity. Therefore, a local time step for each model could complicate the algorithm without a clear advantage. In any case, it could be concluded that the fact of having the 1D model implemented on CPU does not affect the global computational time significantly. The influence of the floodplain in the whole simulation has also been evaluated. Depending on the number of wet cells on the floodplain, their area and their velocity (thus, the time step size), the time consumed by the 2D scheme will be higher. The sensitivity test case, with a finely discretized bathymetry, has been subjected to different hydrographs studying the influence of the maximum flooded area.

After the sensitivity analysis in an idealized case, the coupled model is applied to a specific real case. The challenge lies in all the hydraulic structures that the floodprone area contains, and the necessity of very small cells for their representation. This situation leads to lower speed-ups due to the extreme influence of the floodplain on the simulation time. Although a speed-up greater than one is accomplished, the profitability of the coupled model could be discussed for short cases. It is worth mentioning that the speed-up in these cases is fully dependent on the duration of the event. For the 5 days event, with a low peak, it is not that as high as for the long case with a higher peak.

The use of a coupled model can be extremely worthwhile for substituting a full 2D model for flood simulation when HPC is used, as in the proposed model. The use of the CPU only for the computation of the 1D scheme does not present a bottleneck and the efficiency of the model has been proved. Nevertheless, it is important to evaluate the type of simulation to be run so that the efficiency of the coupled model is ensured to be worth it. Another final fact that should be mentioned is the additional effort that the geometric link between sub-domains requires for mesh creation. Thus, although the presented results are positive and the model has proved its efficiency, each simulation case should be evaluated before choosing a coupled or a full 2D model.

ACKNOWLEDGEMENTS

This work is part of the PGC2018-094341-B-I00 research project funded by the Ministry of Science and Innovation/FEDER. Additionally, Mario Morales-Hernández was partially supported by the U.S. Air Force Numerical Weather Modeling Program.

REFERENCES

Bladé, E., Gómez-Valentín, J., Dolz, J., Aragón-Hernández, J. L., Corestein, G. & Sánchez-Juny, M. 2012 Integration of 1D and 2D finite volume schemes for computations of water flow in natural channels. *Advances in Water Resources* **42**, 17–29.

Board, O. A. R. 2015 *OpenMP Application Programming Interface*. High-Performance Computing Center Stuttgart, Germany.

Brodtkorb, A., Saetra, M. & Altikanar, M. 2012 Efficient shallow water simulations on GPU's: implementation, visualization, verification and validation. *Computers and Fluids* **55**, 1–12.

Burguete, J. & García-Navarro, P. 2001 Efficient construction of high-resolution TVD conservative schemes for equations with source terms: application to shallow water flows. *International Journal for Numerical Methods in Fluids* **37** (2), 209–248.

Castro, M. J., Ortega, S., de la Asunción, M., Mantas, J. M. & Gallardo, J. M. 2011 GPU computing for shallow water flow simulation based on finite volume schemes. *Comptes Rendus Mécanique* **339** (2–3), 165–184.

Caviedes-Voullième, D., García-Navarro, P. & Murillo, J. 2012 Influence of mesh structure on 2D full shallow water equations and SCS curve number simulation of rainfall/runoff events. *Journal of Hydrology* **448**, 39–59.

Chanson, H. 2004 *Hydraulics of Open Channel Flow*. Butterworth-Heinemann, Oxford.

Costabile, P., Macchione, F., Matale, L. & Petaccia, G. 2015 Flood mapping using LIDAR DEM. limitations of the 1-D modeling highlighted by the 2-D approach. *Natural Hazards* **77** (2), 181–204.

C. o. t. E. U. European Parliament 2007 Directive 2007/60/EC of the European Parliament and of the Council of 23 October 2007 on the assessment and management of flood risks, Directive.

Cunge, J., Holly, F. & Verwey, A. 1980 *Practical Aspects of Computational River Hydraulics*. Pitman.

Domeneghetti, A., Vorogushyn, S., Castellarin, A., Merz, B. & Brath, A. 2013 Probabilistic flood hazard mapping: effects of uncertain boundary conditions. *Hydrology and Earth System Science* **17**, 3127–3140.

Echeverribar, I., Morales-Hernández, M., Brufau, P. & García-Navarro, P. Use of internal boundary conditions for levees representation: application to river flood management. *Environmental Fluid Mechanics*.

EEA. *European Environment – State and Outlook: Synthesis*. European Environment Agency, Copenhagen.

Henderson, F. 1966 *Open Channel Flow*. Macmillan Series in Civil Engineering, Macmillan.

Horna-Muñoz, D. & Constantinescu, G. 2018 A fully 3-D numerical model to predict flood wave propagation and assess efficiency of flood protection measures. *Advances in Water Resources* **122**, 148–165.

Horritt, M. S. & Bates, P. D. 2002 Evaluation of 1D and 2D numerical models for predicting river flood inundation. *Journal of Hydrology* **268**, 89–99.

Knijff, J. M. V. D., Younis, J. & Roo, A. P. J. D. 2010 LISFLOOD: A GIS-based distributed model for river basin scale water balance and flood simulation. *International Journal of Geographical Information Science* **24** (2), 189–212.

Lacasta, A., Morales-Hernández, M., Murillo, J. & García-Navarro, P. 2014 An optimized GPU implementation of a 2D free surface simulation model on unstructured meshes. *Advances in Engineering Software* **78**, 1–15.

Leskens, J. G., Brugnach, M., Hoekstra, A. Y. & Schuurmansb, W. 2014 Why are decisions in flood disaster management so poorly supported by information from flood models? *Environmental Modelling & Software* **53**, 53–61.

LeVeque, R. 1992 *Numerical Methods for Conservation Laws, Lectures in Mathematics*. ETH Zürich, Birkhäuser Basel.

Masoero, A., Claps, P., Asselman, N. E. M., Mosselman, E. & Di Baldassarre, G. 2013 Reconstruction and analysis of the Po river inundation of 1951. *Hydrological Processes* **27**, 1341–1348.

Morales-Hernández, M., García-Navarro, P., Burguete, J. & Brufau, P. 2013 A conservative strategy to couple 1D and 2D models for shallow water flow simulation. *Computers & Fluids* **81**, 26–44.

Morales-Hernández, M., Lacasta, A., Murillo, J., Brufau, P. & García-Navarro, P. 2015 A Riemann coupled edge (RCE) 1D–2D finite volume inundation and solute transport model. *Environmental Earth Sciences* **74** (11), 7319–7335. doi:10.1007/s12665-015-4754-3.

Morales-Hernández, M., Petaccia, G., Brufau, P. & García-Navarro, P. 2016 Conservative 1D–2D coupled numerical strategies applied to river flooding: The Tiber (Rome). *Applied Mathematical Modelling* **40** (3), 2087–2105.

Morales-Hernández, M., Echeverribar, I., García-Navarro, P. & Brufau, P. 2018 1D model vs. 2D model for flooding events. In *13th Hydroinformatics Conference*.

Morales-Hernandez, M., Sharif, M. B., Gangrade, S., Dullo, T., Kao, S.-C., Kalyanapu, A., Ghafoor, S., Evans, K., Madadi-Kandjani, E. & Hodges, B. R. High performance computing in water resources hydrodynamics. *Journal of Hydroinformatics*.

Murillo, J. & Garcia-Navarro, P. 2014 Accurate numerical modeling of 1D flow in channels with arbitrary shape: application of the energy balanced property. *Journal of Computational Physics* **260**, 222–248.

Murillo, J., García-Navarro, P. & Burguete, J. 2009 Numerical modelling of shock waves: simulation of a large number of laboratory experiments. *International Journal for Numerical Methods in Fluids* **60**, 1351–1377.

NVIDIA 2010 NVIDIA CUDA Programming Guide Version 3.0. Available from: http://developer.nvidia.com/object/cuda_3_0_downloads.html.

Petaccia, G., Natale, L., Savi, F., Velickovic, M., Zech, Y. & Soares-Frazão, S. 2012 Flood wave propagation in steep mountain rivers. *Journal of Hydroinformatics* **15** (1), 120–137.

Petaccia, G., Leporati, F. & Torti, E. 2016 OpenMP and CUDA simulations of Sella Zerbino dam break on unstructured grids. *Computational Geosciences* **20** (5), 1123–1132.

Sanders, B., Schubert, J. & Detwiler, R. 2010 Parbrezo: a parallel, unstructured grid, Godunov type, shallow water code for high resolution flood inundation modeling at the regional scale. *Advances in Water Resources* **33** (12), 1456–1467.

Sharif, M. B., Ghafoor, S., Hines, T. M., Morales-Hernandez, M., Evans, K., Kao, S.-C., Kalyanapu, A., Dullo, T. & Gangrade, S. 2020 Performance evaluation of a two dimensional flood model on heterogeneous high-performance computing architectures. In *Proceedings of the Platform for Advanced*

Scientific Computing Conference, June 29–July 01, Geneva, Switzerland.

Turchetto, M., Palù, A. D. & Vacondio, R. 2020 A general design for a scalable MPI-GPU multi-resolution 2D numerical solver. *IEEE Transactions on Parallel and Distributed Systems* **31** (5), 1036–1047.

U. E. Agency. *Benchmarking of 2D Hydraulic Modelling Packages*. Agency Report.

Vacondio, R., Dal Palú, A. & Mignosa, P. 2014 GPU-enhanced finite volume shallow water solver for fast flood simulations. *Environmental Modelling and Software* **57**, 60–75.

Vacondio, R., Aureli, F., Mignosa, P. & Dal Palú, A. 2016 Simulation of the January 2014 flood on the Secchia River using a fast and high-resolution 2D parallel shallow-water numerical scheme. *Natural Hazards* **80** (1), 103–125.

First received 10 February 2020; accepted in revised form 18 May 2020. Available online 2 July 2020