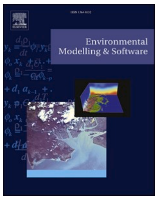
Environmental Modelling and Software 143 (2021) 105102

Contents lists available at ScienceDirect

Environmental Modelling and Software

journal homepage: www.elsevier.com/locate/envsoft



BASEMENT v3: A modular freeware for river process modelling over multiple computational backends

Davide Vanzo[a,b,*], Samuel Peter[e], Lukas Vonwiller[f], Matthias Bürgler[a], Manuel Weberndorfer[d], Annunziato Siviglia[c], Daniel Conde[a], David F. Vetsch[a]

[a] Laboratory of Hydraulics, Hydrology and Glaciology, ETH, Swiss Federal Institute of Technology, Zürich, Switzerland
[b] Dept. of Surface Waters - Research and Management. Eawag, Swiss Federal Institute of Aquatic Science and Technology, Kastanienbaum, Switzerland
[c] Dept. Civil, Environmental and Mechanical Engineering. University of Trento, Trento, Italy
[d] ID Scientific IT Services, ETH, Swiss Federal Institute of Technology, Zürich, Switzerland
[e] Axpo Group, Baden, Switzerland
[f] TK CONSULT AG, Zürich, Switzerland

ARTICLE INFO

Keywords:
GPU-CUDA
River modelling
Unstructured grid
Shallow water
Sediment transport
Pollutant transport

ABSTRACT

Modelling river physical processes is of critical importance for flood protection, river management and restoration of riverine environments. Developments in algorithms and computational power have led to a wider spread of river simulation tools. However, the use of two-dimensional models can still be hindered by complexity in the setup and the high computational costs. Here we present the freeware BASEMENT version 3, a flexible tool for two-dimensional river simulations that bundles solvers for hydrodynamic, morphodynamic and scalar advection-diffusion processes. BASEMENT leverages different computational platforms (multi-core CPUs and graphics processing units GPUs) to enable the simulation of large domains and long-term river processes. The adoption of a fully costless workflow and a light GUI facilitate its broad utilization. We test its robustness and efficiency in a selection of benchmarks. Results confirm that BASEMENT could be an efficient and versatile tool for research, engineering practice and education in river modelling.

Software availability

Name of software: BASEMENT, version 3 (v3),
Website: www.basement.ethz.ch.
E-mail: basement@ethz.ch.
Developer: Numerical Modelling division at the Laboratory of Hydraulics, Hydrology and Glaziology (VAW), ETH Zürich.
Language: C++, C, CUDA.
Interface: graphical user interface (GUI), command-line interface (CLI).
Hardware: CPUs, CUDA-enabled GPUs (optional).
OS: Windows, Linux (Ubuntu).
Availability: Freeware.
Test cases: repository available at ETH Zürich Research Collection, https://doi.org/10.3929/ethz-b-000482308.

1. Introduction

In the last decades the usage of numerical tools widely spread in several subjects of the environmental sciences. River science (*sensu* Gilvear et al., 2016) is no exception in this trend, with a number of tools been developed to address variegate research questions (e.g. Brewer et al., 2018; Shimizu et al., 2019). Modelled river physical processes span from flood simulation, hydraulic and sediment dynamics, pollutant and temperature transport, to vegetation and flow interactions, just to mention a few (e.g. Crosato and Saleh, 2011; Sharma and Kansal, 2012; Williams et al., 2016; Dugdale et al., 2017; Teng et al., 2017). Such river processes occur at different spatial and temporal scales, hence influencing the development and choice of suitable modelling tools.

Advances in computational power, numerical algorithms and optimization routines that occurred in the last decades allowed for the spread of more and more sophisticated numerical tools. In the context of river science, two-dimensional depth-averaged (hereinafter 2D) models are nowadays of common use in research and engineering practice. This

---

[*] Corresponding author. Laboratory of Hydraulics, Hydrology and Glaciology, ETH, Swiss Federal Institute of Technology, Zürich, Switzerland.
E-mail address: vanzo@vaw.baug.ethz.ch (D. Vanzo).

https://doi.org/10.1016/j.envsoft.2021.105102
Accepted 24 May 2021

is particularly true for some applications such as flood modelling and river morphodynamics (e.g. Shimizu et al., 2019; Zischg et al., 2018). The increasing usage of 2D river models is also closely bonded with the growing availability of high-resolution river datasets. In particular, advances in LiDAR, UAV-Photography and others remote-sensed survey technologies enable river topographic scans at an unprecedented level of detail (e.g. Marcus and Fonstad, 2010; Savage et al., 2016).

The increased computational capabilities and refined datasets open the gates for near-census (*sensu* Pasternack, 2011) numerical modelling of several river processes. Indeed, 2D river models have the capability to simulate fine spatial (centimeters to meters) and temporal scales (seconds to days). At such scales, relevant hydro-morphodynamic processes (e.g. bar formation) and also ecohydraulic processes (e.g. habitat dynamics) can hence be modelled (e.g. Maddock, 1999; Siviglia et al., 2013; Wyrick et al., 2014; Guan et al., 2016). Nevertheless, 2D river models can still be computationally demanding, with simulations lasting several days. This is particularly true when complex physical processes, such as morphodynamics, are accounted for (Siviglia and Crosato, 2016). Moreover, large-scale or near-census applications (i.e. with millions of computational cells) and/or long-term simulations (i.e. years) all concur to increase overall computational costs. Such drawbacks particularly apply for the investigation of highly unsteady river processes such as artificial or natural flood waves, where explicit numerical schemes are preferable. In such cases the overall computational time scales exponentially with the number of computational cells due to stability constraints (e.g. Toro, 2001).

Increasing the efficiency and the computational performance of river models represents yet a challenge. Pitfalls arise with the number of computational cells, but also with the inherent complexity of 2D models. For example, challenges are to be found in the setup of the computational domain (Nahorniak et al., 2018) but also in the definition of particular boundary conditions (Costabile and Macchione, 2015; Dazzi et al., 2020). Increasing the computational performance is also sought by developing alternative numerical solution strategies for the underlying physical governing equations. Examples are variegate, spanning from the adoption of a local timestep for the numerical integration (e.g. Sanders, 2008; Dazzi et al., 2018), the automatic adaptation of the computational mesh (e.g. Powell et al., 1993), the use of acceleration factors for the hydro-morphodynamic problem (e.g Carraro et al., 2018; Morgan et al., 2020), to the reformulation of the governing mathematical equations for water quality simulations (e.g. Vanzo et al., 2016), to mention a few.

Parallel computing solutions are the most popular strategies to reduce computational time. They historically benefit from the continuous improvements of parallel performance of both single CPUs and clusters, and the decrease of their unitary price. The general aim of parallelization techniques is to split the total computational load into tasks that can be executed simultaneously by different computational units (e.g. Afzal et al., 2016). The use of the Graphics Processing Unit (GPU) as a general-purpose computational resource developed rapidly in the last decade (Owens et al., 2008). For many parallelizable workloads, offloading work to GPUs is a relatively cheap and efficient high-performance computing strategy that is also easily upgradable in standard desktop workstations.

By means of GPU parallelization, numerical models can potentially be accelerated by a factor of tens and more (e.g. Lacasta et al., 2014). The efficiency of such a parallelization however depends on the data exchange between the main memory and the processors, with a complex memory hierarchy and bandwidth bottlenecks (Mudalige et al., 2012). These low-level constraints can limit computational speedup and depend on the model data/memory handling, the underlying model complexity and the nature of the governing equations to be solved. In applications such as 2D river models the type of computational mesh, i.e. structured or unstructured, has a significant influence on the final computational speedup.

In the last decade, river simulation models have benefited from GPU parallelization. Specific and *ad-hoc* implementations of GPU-based models for 2D hydrodynamic (e.g. Brodtkorb et al., 2012; Smith and Liang, 2013; Vacondio et al., 2014; Horváth et al., 2016; Vacondio et al., 2017), and occasionally morphodynamic simulations (e.g. Hou et al., 2020) have become available. The vast majority of these models are based on structured grids which allow for an easier implementation and for relatively higher computational speedups. This is due to the fact that, for structured meshes, the data structure is inherently simpler, which reduces the need for mappings and indirections. To the best of the Authors' knowledge, few hydrodynamic models implement GPU-acceleration on unstructured meshes (Lacasta et al., 2014, 2015; Castro et al., 2011; Petaccia et al., 2016), with very limited *ad-hoc* implementations for transient flows morphodynamics (e.g. Juez et al., 2016).

Bundled river modelling software that support GPU acceleration are available for commercial use (e.g. RiverFlow2D (hydronia.com/riverflow2d), TUFLOW (tuflow.com), but costless ones are still few (see García-Feal et al., 2018). An increase in availability of freeware GPU-based river models would be beneficial for environmental modelers in academic research and education, but also in consultancy and engineering offices.

In this paper we introduce the BASEMENT software (version 3), a freeware application developed at the Laboratory of Hydraulics, Hydrology and Glaciology of ETH Zürich. The software can simulate two-dimensional hydrodynamic, morphodynamic, and scalar advection-diffusion processes of scientific and practical interest. It can seamlessly run on GPU-enabled workstations, as well as on more standard multi-core CPUs. This flexibility in the choice of the backend, i.e. of the final computational hardware, is achieved by integrating the OP2 framework (Mudalige et al., 2012; Reguly et al., 2016; Giles et al., 2012). This framework provides an additional abstract layer for the acceleration of numerical models on unstructured computational meshes, and has been successfully implemented in similar modelling context (Reguly et al., 2018). The obtained parallelization performance alleviates the computational limitations when simulating high resolution (or large) computational domains and/or long term processes (e.g. Giles et al., 2020). This is particularly relevant when aiming at the calibration (Beckers et al., 2020) or at the uncertainty evaluation (Savage et al., 2016; Jung and Merwade, 2014) of deterministic models. As proof of concept, a flood wave uncertainty propagation analysis with BASEMENT has been proposed in Peter (2017).

In the current version BASEMENT is available for both Windows and Linux-based (Ubuntu) environments. It is provided with a command-line interface (CLI) to easily perform batch simulations, but also with a light graphical user interface (GUI). The BASEMENT software aims to enable a broad range of potential users to skilfully simulate river processes in the domain of river engineering and research on state-of-the-art computational hardware. Moreover, with accompanying scholar programs and extended documentation material and tutorials, the software is designed to be a valuable didactic tool for engineering and river science students.

The paper is structured as follows: §2 provides the software application context that justifies the adopted mathematical and numerical strategies. §3 to §5 report the mathematical basis, the numerical strategies and main features of the basic modules of BASEMENT. The software design, the modelling workflow and the parallelization solutions are presented in §6. A selection of benchmarks are reported in §7, whilst conclusions and outlooks are drawn in §8.

2. Application context

One of the main goals of the novel software design of version 3 is the capability to tackle river processes at different spatial and temporal scales. For example, BASEMENT can be used to simulate large scale (i.e. basin scale) flood propagation, but also reach scale morphodynamic processes such as formation and evolution of fluvial bars. Moreover, it can be applied together with high-resolution topographies (in the order of centimeters) to simulate ecohydraulic processes at different ecological scales (e.g. habitat modelling). This range of application possibilities is enabled by specific characteristics of the software. In particular:

unsteady and transitional flows: BASEMENT can deal with strongly unsteady flows and different flow regimes (sub- and super-critical). For this reason, BASEMENT is particularly suitable for simulations of river flows in Alpine contexts, the propagation of natural flood waves as well as hydropeaking events. This is ensured by the adoption of a robust and accurate shock-capturing explicit solver for the hydrodynamic problem (§4.2);
accurate front propagation: it is possible to simulate extreme events such as dam-break induced floods, but also ecologically-relevant processes such as the wetting-drying of riparian areas and in-channel morphologies due to artificial flow alterations. This is achieved by an implemented shock-wave capturing numerical scheme complemented with a robust treatment of wet-dry interfaces (§4.2);
complex river topographies: the use of unstructured grid for the computational domain discretization enables for an accurate description of complex river morphologies and riverine structures (§4.1). The adoption of an unstructured mesh also reduces the strong anisotropy of structured meshes, which can be crucial for particular applications;
large problems: the software adopts a parallelization strategy tailored to the acceleration of problems on unstructured meshes (§6.4). Moreover, BASEMENT simulations can efficiently be executed on different computational backends. Those backends include GPU cards, therefore allowing for the simulation of large domains (millions of computational cells) on standard workstations, having a limited cost;
multiple river processes: the software is designed in a modular way, so different river processes such as hydrodynamics, sediment transport or advection-diffusion of a scalar (e.g. a non-reactive pollutant) can be simulated by activating specific modules at setup time (§6). Different types of boundary conditions (§5) and closure relationships are available to simulate, for example, simple hydraulic structures (e.g. weirs) or flow inputs/outputs (e.g. water intakes). The modular design (§6) allows to retain good parallelization performances in the simulation of different river processes, as shown in (§7.7).

The basic modules available in BASEMENT are i) hydrodynamics, ii) morphodynamics and iii) advection-diffusion of scalar quantities. Each module is composed by different sets of hyperbolic equations describing the conservation and evolution of the water flow (hydrodynamics), the fluvial sediment (morphodynamics) and the concentration of passive solutes (scalar advection-diffusion). The governing equations represent a so-called Initial-Boundary Value Problem (Toro, 2001), where process-specific initial and boundary conditions are required to be set. The following Sections present the main governing equations and closure relationships (§3), the numerical strategies (§4) and finally the initial and boundary conditions (§5) for the three basic modules. The main module features are also listed in Table 1 of the Supplementary Material.

3. Mathematical formulation

3.1. Hydrodynamics

The hydrodynamic module solves the so-called *shallow water* equations (hereinafter SWE) (e.g. Toro, 2001). The two-dimensional SWE are of practical interest with regard to water flows with a free surface under the influence of gravity.

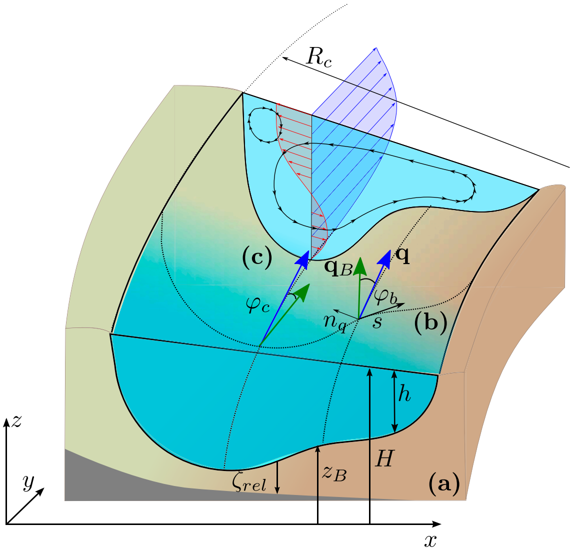


Fig. 1. Notation of scalar and vectorial quantities. (a) Reference system (x, y, z) with water surface elevation H, water depth h, bed elevation z_B and non-erodible fixed bed depth ζ_{rel}. **(b)** Bed load transport ($\mathbf{q}_B$) deviation angle φ_b from the flow direction $\mathbf{q}$ due to gravitational effects caused by the local lateral slope s. **(c)** Bed load transport ($\mathbf{q}_B$) deviation angle φ_c from the flow direction $\mathbf{q}$ due to the spiral flow motion caused by the curvature of radius R_c.

Considering a Cartesian reference system (x, y, z) where the z axis is vertical and the (x, y) plane is horizontal (Fig. 1a), the system of governing equations can be written as:

$$\begin{cases} \partial_t H + \partial_x q_x + \partial_y q_y = S_h \\ \partial_t q_x + \partial_x\left(\dfrac{q_x^2}{h} + \dfrac{1}{2}gH^2 - gHz_B\right) + \partial_y\left(\dfrac{q_x q_y}{h}\right) = -gH\partial_x z_B - ghS_{fx} \\ \partial_t q_y + \partial_x\left(\dfrac{q_x q_y}{h}\right) + \partial_y\left(\dfrac{q_y^2}{h} + \dfrac{1}{2}gH^2 - gHz_B\right) = -gH\partial_y z_B - ghS_{fy} \end{cases} \tag{1}$$

where the system unknowns are the water surface elevation H [m], and the two directional components of $\mathbf{q} = (q_x, q_y)$ [m^2/s], representing the flow discharge per unit width. With z_B [m] we indicate the bottom elevation, whilst $h = (H - z_B)$ [m] is the water depth, and g [m/s^2] the acceleration due to gravity. Note that the depth-averaged velocity vector can be consequently expressed as $\mathbf{u} = (u, v) = (q_x/h, q_y/h)$ [m/s]. Finally S_{fx} and S_{fy} [-] represent the dimensionless friction terms in x and y direction, whilst S_h [m/s] represents potential external contribution/subtraction of flow discharge to the mass conservation equation.

3.1.1. Hydrodynamic closure relationships

To solve the system (1), closure relationships for the friction terms S_{fx}, S_{fy} and the contribution of external inflow/outflow discharge S_h must be provided.

Friction terms. Under the hypothesis of turbulent flow, hence under the assumption that the energy line slope is proportional to the square of the flow velocity, the friction terms S_{fx}, S_{fy} can be written as:

$$S_{fx} = \frac{u \parallel \mathbf{u} \parallel}{ghc_f^2} \; ; \; S_{fy} = \frac{v \parallel \mathbf{u} \parallel}{ghc_f^2}, \tag{2}$$

where c_f is the dimensionless friction coefficient and $\|\mathbf{u}\|$ is the norm of the velocity vector. Several formulae are available for c_f. BASEMENT implements four well known formulations of power or logarithmic type,

Table 1

Friction closure relationships for the hydrodynamic problem. Formulations for the dimensionless friction coefficient c_f; for both Chézy and Bezzola entries, d_{90} is the 90th percentile of the sediment grain size distribution.

Closure	Expression	Parameters	Range	Ref
Strickler	$c_f = k_{str} h^{1/6}/\sqrt{g}$	k_{str} [m$^{1/3}$s^{-1}]	7–40	Armanini (2018)
Manning	$c_f = h^{1/6}/(n\sqrt{g})$	n [m$^{-1/3}$s]	0.025-0.143	Armanini (2018)
Chézy	$c_f = 6.25 + \log(h/K_s)$ for $h > K_s$ $c_f = 6.25$ for $h \leq K_s$	K_s [m]	$K_s = n_k d_{90}$, with $n_k = 2 \div 3$	Graf (1966)
Bezzola	$c_f = 2.5\sqrt{1 - y_R/h}\ \ln(10.9h/y_R)$ for $h/y_R > 2$ $c_f = 1.25\sqrt{h/y_R}\ \ln(10.9h/y_R)$ for $0.5 \leq h/y_R \leq 2$ $c_f = 1.5$ for $h/y_R < 0.5$	y_R [m]	$y_R = nd_{90}$, with $n \approx 1$	Bezzola (2002)

given in Table 1.

External inflow/outflow discharge. The term S_h [m/s] represents additional sources of water like rainfall and springs, or water abstraction (sink), and can be defined over subsets of the computational domain. The external water source can be provided by the user as total discharge [m^3/s] or as intensity [mm/h], per squared meter. Different behaviour can be imposed for each external source/sink area:

- Exact (source/sink): the exact given water volume is added (source) or extracted (sink) from the surface. This is the only option for water addition. In case of water abstraction, the simulation might end abruptly if the available water volume is smaller than the volume prescribed for subtraction. This option allows to have the full control on the water entering/leaving the computational domain, and is useful to simulate e.g. managed hydraulic structures, such as regulated water intakes.
- Available (sink): the given water volume to extract is limited by the available water volume in the single element (i.e. computational cell). With this abstraction option, the simulation proceeds with no interruptions because the water conservation is ensured. This option is useful to simulate particular unmanaged hydraulic structures, such as diversion spillways.
- Infinity (sink): all available water will be abstracted from the computational domain.

3.2. Morphodynamics

The basic morphodynamic module solves the so-called Exner equation (Exner, 1925). It describes the bed evolution due to erosion or deposition, which results in the elevation change of the actual bed level z_B. Assuming the same coordinate reference of Fig. 1a, it reads

$$(1-p)\partial_t z_B + \partial_x q_{B_x} + \partial_y q_{B_y} = S_b, \tag{3}$$

where p [-] is the bed sediment porosity, assumed constant in space and time, S_b [m/s] is an external source term specifying local inputs or outputs of sediment material (e.g. slope collapse or excavation) and $\mathbf{q}_B = (q_{B_x}, q_{B_y})$ [m^2/s] is the specific sediment transport flux.

The morphodynamic module in its basic form accounts only for sediment transport occurring in the form of bed-load or total-load (Armanini, 2018; Parker). The simulation of sediment transport as suspended load is delegated to a specific module, planned in future versions of the software.

3.2.1. Morphodynamic closure relationships

Two closure relationships are needed to numerically solve the governing equation (3): a sediment transport formula and the external source/sink of sediments.

3.2.1.1. Sediment transport formulae. BASEMENT implements four different types of sediment transport formulae as given in Table 2. The first two expressions, Meyer-Peter and Müller like (MPM-like) and Grass like (Grass-like), are adequate to simulate bed-load dominated sediment transport conditions. The Engelund and Hansen formula allows for the estimation of total sediment transport (i.e. suspended and bed-load), whilst Smart and Jäggi is used for bedload transport in steep channels.

The expressions and the typical parameter values to calculate the specific sediment transport magnitude $\|\mathbf{q}_B\|$ are given in Table 2. In the MPM-like formulation, θ is the dimensionless bed shear stress (i.e. Shields parameter Armanini, 2018), θ_{cr} is the critical dimensionless bed shear stress, d_m is the representative grain diameter, $s = \rho_s/\rho$ is the relative density of the sediment with respect to water. The coefficients α, m and the critical threshold θ_{cr} can be assigned by the user or adopted from literature (see Table 2). The Grass-like model proposes a simple bedload transport formula, where $\|\mathbf{q}_B\|$ is a function of the flow velocity magnitude, with u_{cr} as critical threshold velocity. The coefficients α, m and the critical threshold $\mathbf{u}_{cr}$ can be assigned by the user or adopted from literature (Table 2). It is worth remarking that the Engelund and Hansen formula (Engelund and Hansen, 1972), that quantifies the total sediment transport, does not prescribe a threshold condition for incipient motion.

3.2.1.2. Local corrections of the sediment transport. The morphodynamic module implements three corrections to the basic Exner equation (3) to account for the influence of local characteristics of the flow and the bottom on the sediment transport. Namely, i) the influence of local slope

Table 2

Sediment transport closure relationships for the morphodynamic problem. Expressions provide an estimation of the specific sediment transport magnitude $\|\mathbf{q}_B\|$ [m^2/s].

Type	Expression	Parameters	Ref
MPM-like	$\alpha(\theta-\theta_{cr})^m\sqrt{(s-1)gd_m^3}$	$\alpha = 8$, $m = 1.5$, $\theta_{cr} = 0.047$	Meyer-Peter and Müller (1948)
		$\alpha = 4.93$, $m = 1.6$, $\theta_{cr} = 0.047$	Wong and Parker (2006)
Grass-like	$\alpha(\|\mathbf{u}\| - u_{cr})^m$	$\alpha \approx \mathcal{O}(-2;-3), m = 3, u_{cr} = 0.0$	Grass (1981)
Engelund and Hansen	$0.05c_f^2\theta^{5/2}\sqrt{(s-1)gd_m^3}$	–	Engelund and Hansen (1972)
Smart and Jäggi	$\alpha\left(\frac{d_{90}}{d_{30}}\right)^{0.2} J^{0.6}\|\mathbf{u}\|(\theta-\theta_{cr})d_m$	$\alpha = 8$, $\theta_{cr} = 0.05$	Smart and Jaeggi (1983)

on incipient motion, ii) the effect of lateral bed slope and iii) of the flow curvature on the sediment transport direction.

The threshold condition for incipient motion of grains, by Shields (1936), is valid for an almost horizontal bed. In case of a sloped bed in flow direction or transverse to it, the stability of grains is either increased or reduced due to the gravity. The critical shear stress value can be adapted consequently to account for the influence of local longitudinal and transversal slopes. A common approach is to scale the critical shear stress for almost horizontal bed θ_{cr} with a correction factor k:

$$\theta^{*}_{cr} = k\theta_{cr}. \tag{4}$$

BASEMENT implements the correction factor k as proposed in (van Rijn, 1989) and (Chen et al., 2010). Implementation details are given in the official documentation.

The bedload direction can be corrected to account for two relevant morphodynamic processes linked to the slope of the bed and the curvature of the flow. The deviation of the bedload direction from the flow direction can thus be modelled as a deviation angle $\varphi = \varphi_b + \varphi_c$, sum of the correction angle for bed slope (φ_b) and curvature (φ_c), as depicted in Fig. 1 (b and c). The bedload vector is then rotated with the rotation matrix $\mathbf{T}(\varphi)$, being

$$\mathbf{T} = \begin{bmatrix} cos\varphi & -sin\varphi \\ \sin\varphi & cos\varphi \end{bmatrix}, \tag{5}$$

where the angle is positive counterclockwise.

The angle φ_b is estimated with the approach proposed in (Ikeda, 1982) and (Talmon et al., 1995) for the effect of the local transversal bed slope. In particular, the bedload direction deviates from the flow direction in presence of a local transversal bed slope, due to the gravity acting on the bedload sediment particles (Fig. 1b). The bed load deviation φ_b with respect to the flow is therefore evaluated as

$$\tan \varphi_b = -N_l\sqrt{\frac{\theta_{cr}}{\theta}}\mathbf{s}\cdot\mathbf{n}_q, \ \text{for } \mathbf{s}\cdot\mathbf{n}_q < 0, \tag{6}$$

where N_l is an experimental lateral transport factor ($0.75 \leq N_l \leq 2.63$), $s = (\partial_x z_B, \partial_y z_B)$ is the local bed slope and $\mathbf{n}_q$ is the unit vector perpendicular to $\mathbf{q}$ in downhill direction (Fig. 1b).

The angle φ_c accounts for the effect of a marked flow curvature. Due to three dimensional spiral flow motion that establishes in curved flows, the bed load direction tends to point towards the inner side of the curve, while the flow direction points towards the outer side (Fig. 1c). This curvature effect is taken into account according to an approach proposed by Engelund (1974), where the deviation angle φ_c is determined as

$$\tan \varphi_c = -N_* \frac{h}{R_c}, \tag{7}$$

where h is the water depth, N_* is a curvature factor, and R_c denotes the radius of the river bend, positive for curvature in counterclockwise direction. The curvature factor N_* mainly depends on bed roughness and assumes values $N_* \approx 7$ for natural streams (Engelund, 1974), and values up $N_* \approx 11$ for laboratory channels (Rozovskii, 1961).

3.2.1.4. External sediment input/output. The source term S_b represents additional sediment mass input or output (sink) that can be defined on subsets of the computational domain. The source can be specified as total volume flux including porosity [m^3/s]. Similarly to the hydrodynamic case (§3.1.1), different approaches are adopted for the sediment sink, namely *exact*, *available* and *infinity*.

3.2.2. *Fixed bed concept*

Morphodynamic simulations generate deposition and erosion patterns of the riverbed. Erosion processes, if not limited, can proceed indefinitely in the vertical direction. To account for the presence of non-erodible river bottom, as in case of bedrock or concrete cover, a non-erodible fixed bed depth ζ_{rel} (Fig. 1a) can be set. This threshold also determines the volume of sediment available for transport. The fixed bed elevation is defined relative to the initial bottom elevation z_B with $\zeta_{rel} \leq 0$.

3.2.3. *Gravitational transport*

Several algorithms and approaches have been proposed to simulate the sediment flux contribution generated by gravitational collapse and bank erosion (Stecca et al., 2017). In this version of the software, we employed a simple geometrical approximation for gravitational transport, assuming that it occurs when the local bed slope (expressed as an angle γ) between two neighbour computational cells exceeds a given critical angle γ_{cr}. This results in a sediment redistribution due to gravity towards the adjacent cells to restore a stable local slope, i.e. $\gamma \leq \gamma_{cr}$. Further implementation details are provided in the official documentation.

3.3. *Scalar advection-diffusion*

A number of environmental processes, such as pollutant, temperature or nutrient transport, can be modelled assuming the passive advection and diffusion of a scalar quantity, in the form of dissolved or particulated species (e.g. Vanzo et al., 2016). The scalar advection-diffusion module allows for the simultaneous simulation of multiple passive species, up to a maximum of 5. The maximum number of species is limited for purely computational efficiency reasons, i.e. to limit the memory requirements of this module. Based on our experience we consider 5 species a suitable limit for a broad range of applications. The transport of a generic species c can be described by the following advection-diffusion equation:

$$\partial_t q_c + \partial_x \left[\frac{q_x q_c}{h} - h\left(K_{xx}\partial_x\phi_c + K_{xy}\partial_y\phi_c\right)\right] + \partial_y \left[\frac{q_y q_c}{h} - h\left(K_{yx}\partial_x\phi_c + K_{yy}\partial_y\phi_c\right)\right] = S_{\phi_c}, \text{with } c = [1,5], \tag{8}$$

where the unknown is q_c, the specific mass of the species c. It can be expressed as $q_c = h\phi_c$, with ϕ_c the volumetric concentration and h the water depth. The term S_{ϕ_c} is a net source of c and K_{ij} [m^2/s] are the components of the 2D diffusion tensor.

3.3.1. *Scalar advection-diffusion closure relationships*

For the scalar advection-diffusion module, the closure relationships are used to model the contribution of external scalar input and output. In particular, the term S_{ϕ_c} represents an additional scalar mass flux that can be added within portions (regions) of the computational domain. The source can be specified either as an imposed concentration value or a total volumetric flux [m^3/s]. The behavior is analogous to the case of hydro- and morphodynamic sources S_h and S_b (§3.1.1 and §3.2.1).

The terms K_{ij} of the diffusion tensor vary considerably with respect to the physical nature of the transported species. Diffusive transport is modelled in terms of both molecular diffusion K^m and turbulent dispersion K^t_{ij}, such that $K_{ij} = K^m I_{ij} + K^t_{ij}$, with I_{ij} the identity matrix. The molecular diffusion is assumed as an isotropic Fickian process with constant coefficient K^m. Turbulent dispersion is anisotropic (K^t_{ij}) and scales with the friction velocity $u_* = \|\mathbf{u}\|/c_f$ and water depth via a longitudinal α_L and transversal α_T non-dimensional coefficients. Suitable values for open channel flows in natural environments are $\alpha_L = 13$ and $\alpha_T = 1.2$ (Vanzo et al., 2016).

4. Numerical solution

The numerical solution of the governing equations (1), (3) and (8) is sought in a finite volume framework, with a spatial discretization based

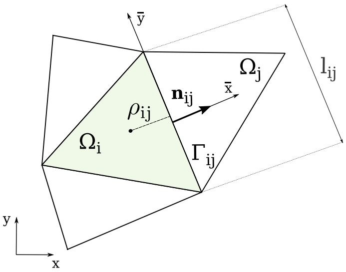


Fig. 2. Sketch of the triangular discretization. Main notations adopted for the generic computational cell i and its j-th neighbour (with $j = 1,2,3$).

on unstructured meshes (§4.1). For the temporal integration, an explicit first order Euler scheme is used. In its basic configuration, the temporal integration proceeds in a synchronous-decoupled way for all the modules, meaning that the modules are independently integrated in time with the same timestep (§4.5). The following sections detail the domain discretization strategy and the adopted numerical solver for the fluxes calculation of the three basic modules. The interested reader should refer to the provided references for specific implementation details.

4.1. Domain discretization

The problem is discretized adopting a finite volume approach over unstructured triangular meshes. A conforming triangulation T_Ω of the computational domain $\Omega \subset \mathbb{R}^2$ by elements Ω_i such that $T_\Omega = \bigcup \Omega_i$, is assumed. Given a finite volume element Ω_i (Fig. 2), $j = 1, 2, 3$ is the set of indexes such that Ω_j is a neighbour of Ω_i. Γ_{ij} is the common edge of two neighbour cells Ω_i and Ω_j, and l_{ij} its length. $\mathbf{n}_{ij} = (n_{ij,x}, n_{ij,y})$ is the unit vector which is normal to the edge Γ_{ij} and points toward the cell Ω_j.

4.2. Hydrodynamics

The system of governing equation (1) can be cast in vectorial form as

$$\partial_t \boldsymbol{U} + \partial_x \boldsymbol{F}_x + \partial_y \boldsymbol{F}_y = \boldsymbol{S}, \tag{9}$$

where left-handside terms of (9) are

$$\boldsymbol{U} = \begin{bmatrix} H \\ q_x \\ q_y \end{bmatrix}, \quad \boldsymbol{F}_x = \begin{bmatrix} q_x \\ \dfrac{q_x^2}{h} + \dfrac{1}{2} g H^2 - g H z_B \\ \dfrac{q_x q_y}{h} \end{bmatrix}, \quad \boldsymbol{F}_y = \begin{bmatrix} q_y \\ \dfrac{q_x q_y}{h} \\ \dfrac{q_y^2}{h} + \dfrac{1}{2} g H^2 - g H z_B \end{bmatrix}. \tag{10}$$

The vector of source terms can be written as $\mathbf{S}(\mathbf{U}) = \mathbf{S}_h + \mathbf{S}_{fr}(\mathbf{U}) + \mathbf{S}_{bed}(U)$, where

$$\boldsymbol{S}_h = \begin{bmatrix} S_h \\ 0 \\ 0 \end{bmatrix}, \quad \boldsymbol{S}_{fr} \begin{bmatrix} 0 \\ = -ghS_{fx} \\ -ghS_{fy} \end{bmatrix}, \quad \boldsymbol{S}_{bed} \begin{bmatrix} 0 \\ = -gH\partial_x z_B \\ -gH\partial_y z_B \end{bmatrix} \tag{11}$$

By integrating the governing system of equation (9) in the control volume $V = [\Omega_i] \times [t^n, t^{n+1}]$, we obtain the general update formula for the triangular element i:

$$\boldsymbol{U}_i^{n+1} = \boldsymbol{U}_i^n - \frac{\Delta t}{|\Omega_i|} \sum_{j=1}^{3} l_{ij} \left[\boldsymbol{F}_{ij}\right] + \Delta t \boldsymbol{S}_i. \tag{12}$$

Problem unknowns at cell i and discrete time n are represented by cell averages $\boldsymbol{U}_i^n$; the numerical solution sought at time $t^{n+1} = t^n + \Delta t$ is denoted by $\boldsymbol{U}_i^{n+1}$. In (12), $\boldsymbol{F}_{ij}$ are the hydrodynamic fluxes estimated at the cell interface ij (Fig. 2).

To compute the fluxes $\boldsymbol{F}_{ij}$ for the hydrodynamic system (9), several well-established solvers are available. Here we adopt the well-known HLLC approximate Riemann solver (Toro et al., 1994) which is a modification of the basic HLL scheme to account for the influence of intermediate contact waves. Further details on the HLLC approach are available in Chapter 10 of (Toro, 2001). The solver is proved to be robust and efficient in simulating unsteady flows and the advection of passive tracers (Vanzo et al., 2016).

The numerical discretization of the three terms of **S(U)** (11) is conducted separately, according to the nature of each term. The external inflow/outflow contribution $\mathbf{S}_h$ is added explicitly to the continuity equation, as it is not a function of the problem unknowns. The stiff friction source terms $\mathbf{S}_{fr}(\boldsymbol{U})$ are integrated with Runge-Kutta 2 (e.g. Toro, 2009) in a semi-implicit fashion after adopting a splitting technique. The implementation is analogous to the ones proposed in (Siviglia et al., 2013; Vanzo et al., 2016; Vanzo, 2015). The topographical terms $\mathbf{S}_{bed}(\boldsymbol{U})$ are discretized using the modified-state approach proposed by (Duran et al., 2013). This results in an easy and robust treatment of complex topographies and wetting and drying problems (Vanzo et al., 2016).

4.3. Morphodynamics

The Exner equation is solved in a synchronous-decoupled way with respect to the shallow water problem (§4.2), meaning that the numerical integration of the Exner equation (3) uses the same integration timestep Δt of the hydrodynamic problem. The general update formula for the Exner problem reads:

$$z_{Bi}^{n+1} = z_{Bi}^{n} - \frac{1}{1-p}\left[\frac{\Delta t}{|\Omega_i|}\sum_{j=1}^{3} l_{ij}\left[q_{Bij}\right] + \Delta t S_{bi}\right], \tag{13}$$

with the same symbols introduced for (3). The term q_{Bij} represents the normal sediment flux at the cell interface ij (Fig. 2).

For the numerical estimation of the term q_{Bij}, a number of approaches are available in literature. In the current version, BASEMENT implements an Approximate Riemann Solver of HLL-type (*sensu* Toro, 2001), as in (Soares-Frazão and Zech, 2011). The sediment flux is thus calculated as

$$q_{Bij} = \frac{\lambda_s^+ q_{Bi} - \lambda_s^- q_{Bj} + \lambda_s^+ \lambda_s^- (z_{Bj} - z_{Bi})}{\lambda_s^+ - \lambda_s^-}, \tag{14}$$

where pedix i (j) refers to quantities evaluated at the corresponding cell (Fig. 2), and λ_s^+, λ_s^- are speed estimations of the morphological problem. We adopt the following speed estimates (Soares-Frazão and Zech, 2011):

$$\lambda_s^- = min(\lambda_{1i}, \lambda_{1j}) \ , \lambda_s^+ = max(\lambda_{2i}, \lambda_{2j}). \tag{15}$$

The expression for the terms λ_1 and λ_2, calculated for both cell i or j reads:

$$\lambda_{1,2} = \frac{1}{2}\left(u_n - c \pm \sqrt{(u_n - c)^2 + 4\frac{\partial q_{B,n}}{\partial q_n} c^2}\right), \tag{16}$$

where u_n is the normal velocity at the cell interface ij (Fig. 2) and $c = \sqrt{gh}$ is the so-called wave celerity.

4.4. Scalar advection-diffusion

The scalar advection-diffusion problem is solved in a synchronous-decoupled way with respect to the shallow water problem (§4.2). We reformulate the governing equation (8) via a Cattaneo-type relaxation technique, as proposed by (Vanzo et al., 2016). Two additional scalar conservation equations are then added to (8), namely

$$\partial_t \psi_x^c - \partial_x \frac{\phi_c}{\varepsilon} = -\frac{\psi_x^c}{\varepsilon} \qquad \partial_t \psi_y^c - \partial_y \frac{\phi_c}{\varepsilon} = -\frac{\psi_y^c}{\varepsilon} \tag{17}$$

where ε is a positive and small relaxation time, whilst ψ_x^c and ψ_y^c are two auxiliary variables that recover $\partial_x \phi_c$ and $\partial_y \phi_c$, respectively, for a sufficiently small ε (Vanzo et al., 2016). After a trivial substitution of $\psi_x^c \approx \partial_x \phi_c$ and $\psi_y^c \approx \partial_y \phi_c$ into (8), the system composed by (8) and (17) can be rewritten in vectorial form as

$$\partial_t \boldsymbol{Q} + \partial_x \boldsymbol{A}_x + \partial_y \boldsymbol{A}_y + \partial_x \boldsymbol{D}_x + \partial_y \boldsymbol{D}_y = \boldsymbol{S}_c + \boldsymbol{S}_{rel} \tag{18}$$

where the vectors $\boldsymbol{Q}$, $\boldsymbol{A}_x$, $\boldsymbol{D}_x$, $\boldsymbol{S}_c$ and $\boldsymbol{S}_{rel}$ read

$$\boldsymbol{Q} = \begin{bmatrix} q_c \\ \psi_x^c \\ \psi_y^c \end{bmatrix}, \quad \boldsymbol{A}_x = \begin{bmatrix} \dfrac{q_c q_x}{h} \\ 0 \\ 0 \end{bmatrix}, \quad \boldsymbol{D}_x = \begin{bmatrix} -h\left(K_{xx}\psi_x^c + K_{xy}\psi_y^c\right) \\ -\dfrac{q_c}{\varepsilon h} \\ 0 \end{bmatrix},$$
$$\mathbf{S}_c = \begin{bmatrix} S_{\phi_c} \\ 0 \\ 0 \end{bmatrix}, \quad \boldsymbol{S}_{rel} = \begin{bmatrix} 0 \\ -\dfrac{\psi_x^c}{\varepsilon} \\ -\dfrac{\psi_y^c}{\varepsilon} \end{bmatrix}, \tag{19}$$

with $\boldsymbol{Q}$ representing the conserved scalar quantities, whilst $\boldsymbol{A}_x$ is the advective fluxes vector and $\boldsymbol{D}_x$ is the diffusive-relaxed fluxes vector, both in x direction. The scalar source terms are $\boldsymbol{S}_c$, whilst the source terms arising form the relaxation are $\boldsymbol{S}_{rel}$. For brevity, we omit the formulation for the y direction ($\boldsymbol{A}_y$ and $\boldsymbol{D}_y$), which is analogous. The interested reader can refer to (Vanzo et al., 2016) for a step-by-step derivation.

The scalar fluxes in (19) are solved through the SVT solver introduced by (Vanzo et al., 2016). The scheme presents a flux-splitting approach combining the advective and diffusive-relaxed fluxes, evaluated with different solvers. The HLLC solver, applied for the hydrodynamic fluxes (§4.2) provides the advective component of the scalar fluxes at the cell interface $\boldsymbol{A}_{ij}$. For the diffusive-relaxed component, the SVT technique derives the fluxes at the interface $\boldsymbol{D}_{ij}$ directly from the Riemann invariants of a two non-linear waves Riemann problem.

Similarly to the hydro- and morphodynamic problems, the control volume $V = [\Omega_i] \times [t^n, t^{n+1}]$ is used to integrate the governing system (18), in order to obtain the following scalar update formula at the element i:

$$\boldsymbol{Q}_i^{n+1} = \boldsymbol{Q}_i^n - \frac{\Delta t}{|\Omega_i|}\sum_{j=1}^{3} l_{ij}\left[\boldsymbol{A}_{ij} + \boldsymbol{D}_{ij}\right] + \Delta t(\boldsymbol{S}_c + \boldsymbol{S}_{rel})_i, \tag{20}$$

where the fluxes $\boldsymbol{A}_{ij}$ and $\boldsymbol{D}_{ij}$ are computed at each cell interface ij (Fig. 2).

The numerical integration of the two source term vectors is conducted separately, according to the nature of the terms. The scalar sources $\boldsymbol{S}_c$ are computed with a first-order explicit Euler scheme, while the stiff relaxation source terms $\boldsymbol{S}_{rel}(\boldsymbol{Q})$ are integrated by means of a locally implicit Euler method.

4.5. Stability condition

Numerical integration proceeds with a dynamic timestep Δt, evaluated at each time loop (Fig. 6b) that fulfills the well-known Courant-Friedrichs-Lewy stability condition (Toro, 2001). In the current implementation, the condition is expressed as:

$$\Delta t = CFL\min_{1 \le i \le N}\left(\min_{1 \le j \le 3}\left(\frac{\rho_{ij}}{\lambda_{ij}}\right)\right), \tag{21}$$

where ρ_{ij} is twice the distance between the edge j and the centroid of the cell i (Fig. 2), and N is the total number of domain elements. The term λ_{ij} is an estimation of the largest eigenvalue of the hydrodynamic problem (1), namely $\lambda_{ij} = |u_n| + \sqrt{gh}$ with the symbology already introduced. The CFL coefficient ranges between 0 and 1: by default it is set to 0.9, if not specified otherwise.

5. Initial and boundary conditions

5.1. Initial conditions

All modules require the user to define the initial conditions of the simulation. Two types of initial conditions are similarly available for all the modules:

- region defined: user explicitly defines the initial values of the problem unknowns (e.g. water depth and specific discharge for hydrodynamics). Different values can be assigned to different region of the computational domain;
- continue: values are taken from the result file of previous simulations.

In addition, the hydrodynamic module allows also to set dry conditions (no water in the domain) as initial conditions. In this case, the domain will progressively fill with water, in relation to the assigned inflow boundary conditions or internal sources (§3.1.1).

5.2. Boundary conditions

The boundary conditions (hereinafter BCs) have different specifications for each core module (see following Sections), but they all classify in three common types: external *standard*, external *linked* and *internal* BCs.

Fig. 3 exemplifies the main concepts adopted for the BCs. The computational domain Ω is defined by the domain boundaries, as $\Gamma_{1,2,3}$. An external *standard* BC is dependent only on the local flow conditions and on some user-defined rules. This represents the most common case, for example to define impermeable walls or river inflows and outflows. By default, all the external boundaries are set as wall. The *wall* BC consists of a fixed, frictionless (inviscid), reflective impermeable wall. In external *linked* BCs instead, the local BCs are defined also with information from a *linked* boundary. Typical example is a weir, where the flow discharge at the downstream side of the weir depends on the water stage on the upstream side.

The third type of BCs, *internal*, are defined within the computational domain Ω, and not at the edges (Fig. 3). This BC type comes in handy in case of very large domain application, because it allows to test different

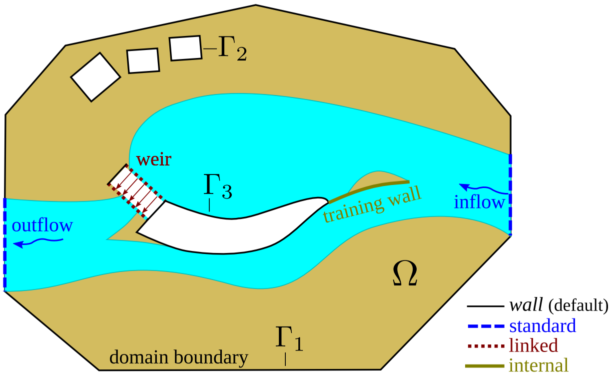


Fig. 3. Example of modelling domain with different types of boundary conditions. The computational domain Ω can include the river channels but also the surrounding floodplains. The domain is delimited by the external BCs: impermeable walls (default type, $\Gamma_{1,2,3}$) are depicted in solid black, while standard inflow and outflow are in dashed blue. A weir (dotted brown) is modelled with an external linked BC. A training wall (thick solid green) is modelled with an internal BC.

configurations of hydraulic structures (e.g. different locations of a weir or training wall), without the need of regenerate the entire computational mesh for every configuration.

A summary of the main features of the BCs for the three core modules follows here. The interested reader can refer to the official documentation for further details.

5.2.1. *Hydrodynamic BCs*

The hydrodynamic module implements different types of BCs, with a different level of customization. Depending on the BC type, user-assigned data is requested, as single constant value in time (e.g. lake level, constant discharge), as time series (e.g. hydrograph), or as set of parameters describing a dynamic behaviour (e.g. weir activation rule). In particular:

- Standard BCs: in addition to *wall* BC, inflows (upstream BCs) and outflows (downstream BCs) can be assigned. As standard inflows, three options are provided: with *uniform* or *explicit* options, the user provides a total volume discharge Q [m^3/s], whilst with *zhydrograph* the water surface elevation [m] must be provided. For the standard outflows a value for the water depth h must be specified. Possible options are: uniform conditions, hydraulic weir, rating curve, zhydrograph (i.e. water surface elevation) and zero_gradient (i.e. Neumann BC). It is worth remarking that the specific type of upstream and downstream BCs should be selected depending on the local flow conditions (i.e. sub- or super-critical).
- Linked BCs: this type of boundaries establishes a *link* between two certain region of the domain where the governing equations are not solved. It is specifically designed to simulate the behaviour of hydraulic structures within the river channel, such as weirs, gates, bridges, spillways.
- Internal BCs: they are fictitious boundaries defined as segments at the interfaces of some computational cells. On these segments, three different conditions can be enforced, instead of the solution of the SWE (1). Options are: *static* walls, *dynamic* walls and *rating curve*. With the static wall, the standard *wall* condition is applied on both sides of the internal boundary. This option is useful, for example, for easily testing the presence of barriers (e.g. training wall in Fig. 3) without the need of reconstructing the numerical domain. With the dynamic wall, the wall conditions are applied until a given threshold is reached, after which the wall is removed, and the SWE are solved instead. The threshold can be set as a time value (i.e. wall removal at a given time) or as water depth (removal when a given water depth is reached on one side of the BC). This feature comes in handy to simulate, for example, the collapse of some hydraulic structures. With the rating curve option (i.e. *h-Q relation*) an unidirectional flow is applied through the internal boundary based on the water stage in the upstream side of the boundary, as for the standard and linked rating curve BCs.

5.2.2. *Morphodynamic BCs*

The sediment flow is defined as a specific bedload flux, which is averaged and evenly distributed at the domain boundary conditions over the boundary length. In analogy with the hydrodynamic module, the morphodynamic boundaries are of type external *standard* and *linked*.

- Standard BCs: for the upstream BCs, BASEMENT implements three versions that allow to simulate: i) a given input of sediment as time series (i.e. sedimentograph), ii) a sediment input derived from the hydrodynamic conditions under transport capacity conditions or iii) bed equilibrium condition, where the upstream bed elevation is kept constant. Two downstream BCs are available, allowing the simulation of i) equilibrium condition and ii) check-dam. In this second option an equilibrium boundary condition is activated only if the bed level reaches a given threshold value, otherwise a wall type boundary is assumed.
- Linked BCs: one BC is available. It allows for the simulation of sediment transport through given hydrodynamic linked conditions, hence to ensure sediment continuity in the simulated channel.

5.2.3. *Scalar advection-diffusion BCs*

Scalar BCs are defined in terms of concentration of total volumetric rate [m^3/s], evenly distributed throughout the length of the relevant domain boundary. The implemented types are:

- Standard BCs: three types are available. i) scalar inflow as a constant value; (ii) scalar inflow as a time-series and (iii) zero_gradient (i.e. Neumann BC) outflow.

6. Software design

6.1. *Modelling workflow*

The standard modelling procedure involves three phases: the pre-processing, the numerical simulation and the post-processing phase (Fig. 4). BASEMENT is designed to integrate into this workflow. Moreover, the entire workflow relies on open-source or freeware tools. In the following we list the phases and provide a short description of the different configuration and results file formats as used by BASEMENT.

1. Pre-processing: in this phase the user is required to define the model domain and the input data. The mesh file (customized 2dm format, `MyMesh.2dm` in Fig. 4) contains the description of the triangular unstructured computational mesh. The file can be generated with BASEmesh, an open-source Python module as well as a QGIS plugin (see the official repository for details), or via grid generator software that supports the 2dm format. In addition, further input data such as time series of water and sediment discharge (or other quantities) to be used as BCs can be provided (ASCII format, `MyData.txt` in Fig. 4).
2. Numerical simulation: the actual simulation can be run via either CLI or GUI. In BASEMENT version 3, the numerical simulation is split into three steps (description follows in §6.2). The final simulation results are stored in a general purpose binary container (Hierarchical Data Format HDF5, www.hdfgroup.org). BASEMENT generates also an XDMF file (eXtensible Data Model and Format, http://www.xdmf.org) which contains a machine-readable description of the data stored in the HDF5 file.
3. Post-processing: the XDMF file (`output.xdmf` in Fig. 4) can be opened with the Crayfish plugin for QGIS or with Paraview for final results visualization and further post-processing. In addition, *ad-hoc* Python scripts can be used to manipulate results directly from the binary container (some scripts are provided at the software website).

6.2. *Simulation steps*

The numerical simulation phase consists of three steps: the pre-simulation, the simulation, and the post-simulation (Fig. 5). Each step can be completed by running a corresponding BASEMENT executable via GUI or CLI. This modular design allows a customization of the simulation workflow by the user and an efficient batch processing of BASEMENT steps. For instance, the programs can be run from a scripting language like Python.

The different executables are configured using a dedicated command file in standardized JSON file format (JavaScript Object Notation) (Fig. 5). The BASEMENT GUI is designed to support the user with creating the command files, and running and monitoring the three simulation steps. In particular, the GUI validates the configuration parameters and automatically adds required parameters where default values are available. The three simulation steps are detailed as follows.

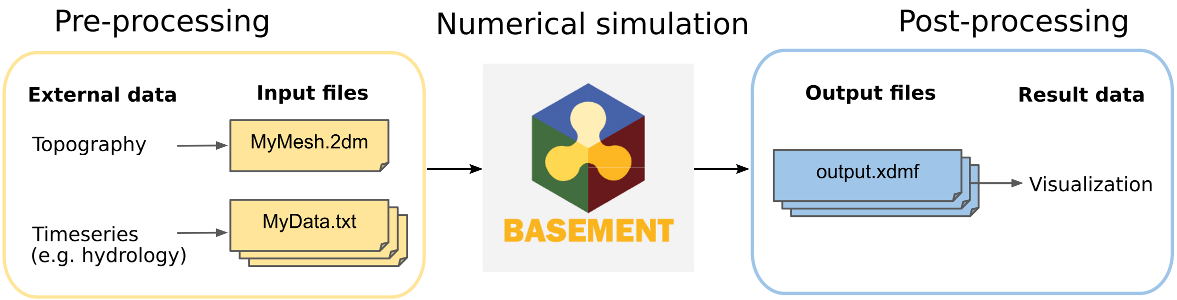


Fig. 4. Modelling workflow. Pre-processing: generation of the computational mesh from topographical data and definition of input time series. Numerical simulation: BASEMENT. Post-processing: elaboration and visualization of the results.

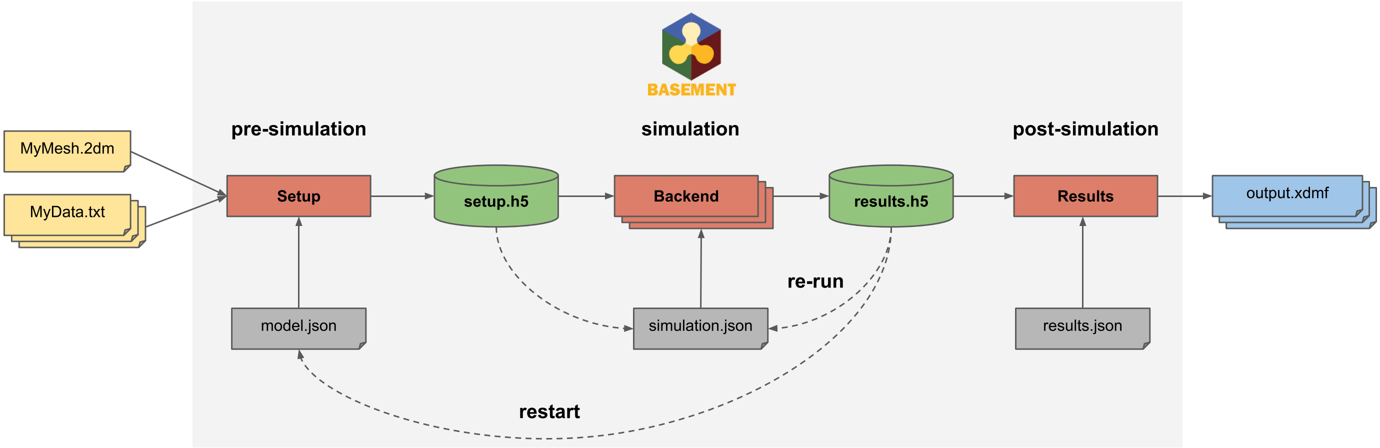


Fig. 5. Software components and simulation steps. The BASEMENT software is composed of a set of executables (red rectangles) driven by JSON configuration files (grey labels). Data is stored in HDF5 containers (green cylinders). In dashed arrows: the special actions of simulation *re-run* and *restart*.

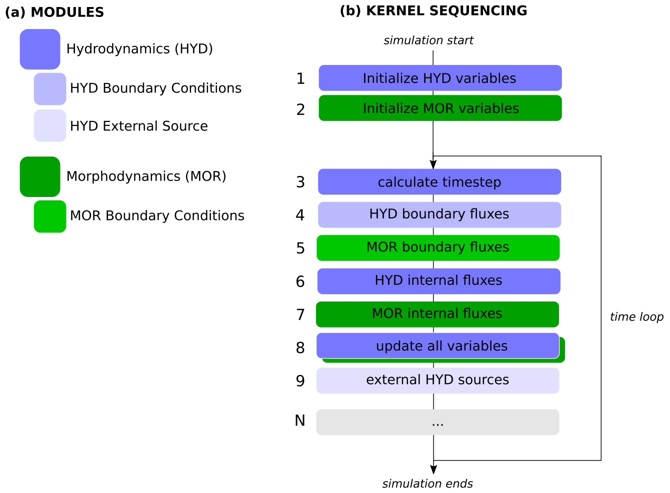


Fig. 6. Examples of modules and associated kernels. (a) At model setup, depending on the simulated physical problem, the user triggers the activation of a set of modules. **(b)** The active modules trigger a unique kernel sequence to be executed at simulation time to correctly simulate the requested processes. Each module corresponds to a different set of kernels.

1. The pre-simulation step focuses on the model definition. In particular, the `model.json` command file contains: i) physical properties, ii) initial conditions and iii) boundary conditions of the physical problem, and further iv) numerical parameters. The setup executable first reads the computational mesh `MyMesh.2dm`, the external required data `MyData.txt` and the command file `model.json`. Then it validates and stores the model inside the binary container `setup.h5`.
2. The simulation is carried out on a selected computational backend (§6.4). It is driven by the command file `simulation.json` that contains the simulation parameters such as the total simulation time, the output timestep (constant) and the desired output quantities. The program reads and executes the model `setup.h5` generated in the previous step. The results of the simulation are stored in a second binary container: `results.h5`. The user can monitor the simulation execution on the GUI log terminal, where the simulation progress (as %), the current integration timestep and an estimation of the Real Time Speed (RTS) of the simulation (ratio of simulated time over computational time) are provided.
3. The post-simulation step is configured using the command file `results.json` that contains the selected output format (currently only XDMF is supported). The output is then available for the post-processing phase (§6.1).

When it is necessary to run a new simulation starting from the results of a previous one, two options are available: *Restart* and *Re-run*. When performing a *Restart* the *pre-simulation* step is executed again, i.e. a new model is generated from scratch with (potentially) a new set of parameters. The user indicated an existing `results.h5` file that is to be used to fetch the initial conditions for the new model. The *Re-run* option does not generate a new model, but uses the existing model (`setup.h5`) with initial conditions taken from the current results file. In this scenario the user can only modify the simulation and results parameters (i.e. duration, output timestep and output type), but not the model parameters. This second option is particularly useful in the case of large models (i.e. millions of computational cells), because the pre-simulation step can take up to tens of minutes. If the user only needs to extend the simulation

Table 3
Description of available backend types for BASEMENT v3.

Type	Description
seq	sequential execution on the CPU
omp	multi-threading using OpenMP technology
cuda	GPU
cudaC	GPU with some kernels running sequentially on the CPU
cudaO	GPU with some kernels running in parallel (OpenMP) on the CPU

duration, for example, then the *Re-run* option allows to skip the pre-simulation step.

6.3. Modularity and sequencing

BASEMENT aims to simulate different river processes with a high level of flexibility and efficiency: with this in mind, we designed the software adopting a modular approach. The two core concepts of this design are *modules* and *kernels*, described as follows.

Modules take care of the simulation of specific river processes (e.g. module "Hydrodynamics" in Fig. 6a). They can be nested to simulate processes with an increasing level of detail/complexity (e.g. module "HYD External Source", Fig. 6a). Modules are activated by the user in the pre-simulation step. An activated module triggers the execution of a number of *kernels* throughout the simulation.

A *kernel* is a set of operations to be executed on each entity (e.g. a cell i or an edge j, Fig. 2) of the computational domain or a subset of it. Depending on the specific task, kernels can be scheduled for a single execution (i.e. initialization kernels) or for repeated execution in each iteration of the integration time loop (Fig. 6b). The global time loop is executed with a timestep Δt that satisfies the stability condition of the hydrodynamic problem (§4.5).

Some modules and their associated kernels, can be scheduled by the user for a delayed start or for execution at different time intervals larger than the global integration timestep Δt. For example, the user can set a delayed starting time (in seconds) of morphodynamic and scalar transport modules. Typical usage is to ensure steady hydrodynamic initial conditions before starting the morphodynamic (or advection-diffusion) simulation. For other modules, such as the gravitational transport or the flow curvature calculation, the user can define also the execution interval (in seconds): this feature allows to reduced the computational efforts when the integrated processes are not subject to the hydrodynamic timestep constrain (§4.5). By default, all modules are executed from the beginning of the simulation and at each global integration timestep.

The architecture based on modules and kernels has two main advantages. First, it is *flexible* in that it allows users (and software developers) to easily add or remove specific modules without interfering with other existing modules. In particular, this permits an integration of further modules as development continues (§8). Second, it is *efficient*, because only the necessary kernels are scheduled for execution at setup time (pre-simulation step).

6.4. Parallelization strategy and computational backends

The parallelization strategy of the BASEMENT numerical core addresses two main aspects: i) the use of different technologies (i.e. computational backends) generated from the same, unique software source code. This allows for an easier source code maintenance and integration of future/different backends. ii) An efficient and heavy parallelization of the numerical core following the concept of data parallelism. To this end the numerical core of BASEMENT integrates OP2 (Mudalige et al., 2012; Giles et al., 2012), which is an open-source framework for the development of unstructured grid applications. Using source-to-source translation, OP2 generates the appropriate code for different target platforms by introducing an additional level of abstraction between the numerical algorithm and its execution. It supports multi-core CPUs, GPUs, and even clusters via MPI (Message Passing Interface, http://www.mpi-forum.org).

BASEMENT currently supports multi-core CPUs and GPUs. When starting the simulation, the user can select to compute on the CPU, the GPU, or a combination of both. All the currently supported backends (Table 3) are available for both Windows and Linux (Ubuntu) operating systems. It is important to note that the choice of graphics processing units is currently limited to Nvidia (CUDA) cards. The precise requirements are provided in the official documentation. All the backends can execute the numerical simulations in double (default) or single precision, with different performance (§7.7).

7. Results

A set of selected test cases (T1-T6) are proposed here to test the robustness, accuracy and efficiency of the three basic modules. Table 4 summarizes the key features of each test case. The interested reader can refer to the official software documentation for further examples. Finally, §7.7 focuses on the software performance and scalability. All the test cases are freely available at ETH Zürich Research Collection, doi https://doi.org/10.3929/ethz-b-000482308.

7.1. T1: Malpasset dam collapse

The scope of this test is to assess the robustness and accuracy of the hydrodynamic solver when simulating a shock-type hydrodynamic wave travelling on a highly irregular and dry domain. The collapse of the Malpasset dam, in the Reyran River Valley (Fréjus, France), represents a well-established hydrodynamic benchmark for numerical models (e.g. Hervouet and Petitjean, 1999; Singh et al., 2011; Valiani et al., 2002). In 1959, the 66.5 m high dam collapsed almost instantaneously, generating an up to 40 m high flood wave that propagated down the Reyran valley, destroyed the two villages Malpasset and Bozon and reached the Mediterranean Gulf 21 min later (Valiani et al., 2002). The propagation of the flood wave was reconstructed via the maximum water level and the flood arrival time, recorded at multiple locations. In particular, the maximum water level is available from a police survey for 17 survey points, marked as P1 to P17 in Fig. 7 and the flood arrival time is known from three electric transformer stations which have been destroyed by

Table 4
Main features of the adopted benchmarks.

ID	Module	Comparison	Key features
T1	hydrodynamic	field data	i) highly unsteady shock wave generation and propagation, ii) wet-and-dry processes, iii) performance of the hydrodynamic module
T2	morphodynamic	lab data	i) sediment transport with a transcritical 1D flow, ii) upstream BCs: uniform inflow, iii) downstream BCs: imposed water level
T3	morphodynamic	lab data	i) 2D dam-break in a complex domain, ii) sediment transport with an advancing wet-and-dry front, iii) downstream BCs: free outflow, iv) performance of the morphodynamic module
T4	morphodynamic	lab data	i) erosion and deposition in a channel bend, ii) sediment transport direction correction, iii) upstream BCs: unsteady hydrograph
T5	scalar advection-diffusion	numerical sol	i) 1D strong rarefaction waves, ii) conservation of a steady discontinuity of the scalar quantity
T6	scalar advection-diffusion	numerical sol	i) 2D complex domain, ii) advection and diffusion of two different scalar quantities, iii) conservation and mixing of the scalars, iv) performance of the scalar advection-diffusion module

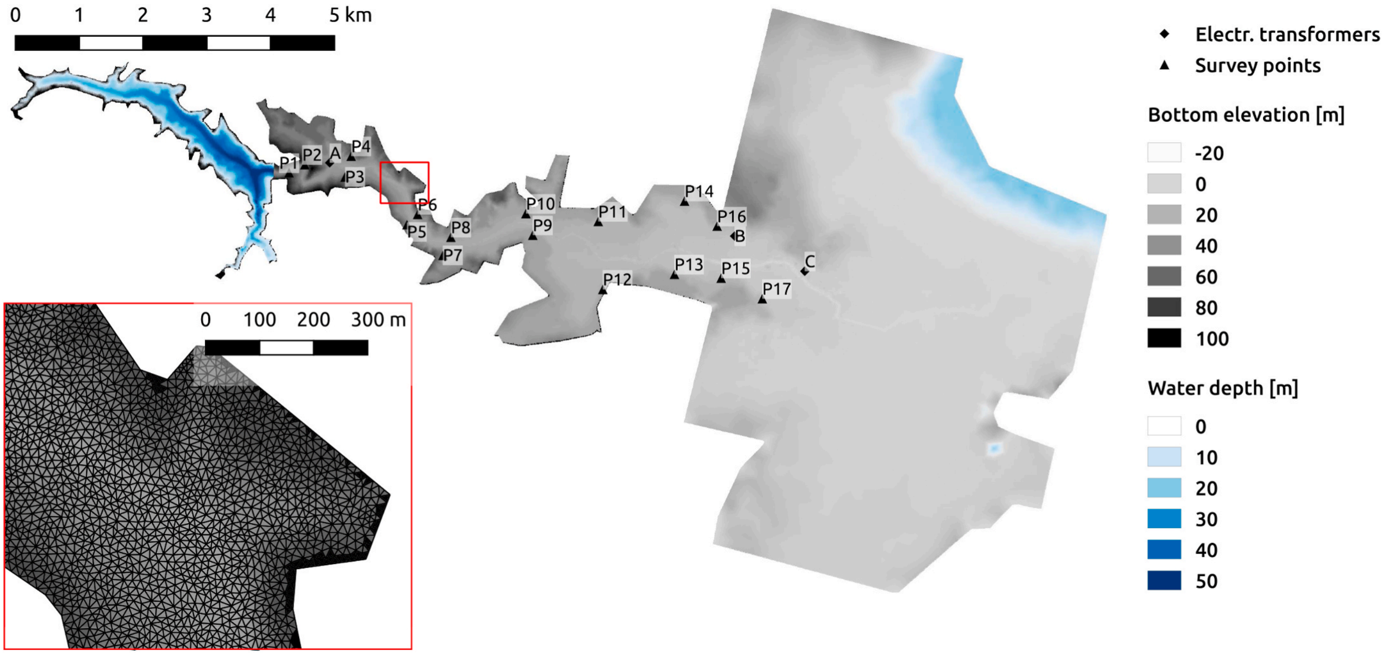


Fig. 7. T1: planar view of the Malpasset test case. Visualization of the computational domain illustrating the bottom elevation (grey scale) and the initial water depth (blue scale). Letters indicate the location of the transformer stations (A–C), and the maximum water level survey points (P1–P17). The left box shows a magnification of the computational mesh, as reference.

Table 5

T1: Malpasset dam-break wave arrival times. Observed and simulation time of flood arrival (TFA) and relative error (Err) for the three electrical transformer stations (ET) destroyed by the flood wave.

ET [-]	x [m]	y [m]	TFA_{obs} [s]	TFA_{sim} [s]	Err [%]
A	5550	4400	100	103	3
B	11900	3250	1240	1287	3.8
C	13000	2700	1420	1435	1

the flood wave. The locations of the transformer stations are indicated as A, B and C in Fig. 7. Coordinates and recorded arrival times are listed in Table 5. We make use of such field data to test the performance of the hydrodynamic module.

The computational domain is discretized with 499,059 triangular elements. The domain boundaries are set to *walls* (§5.2.1), with exception of the downstream boundary located in the Mediterranean Gulf, where a fixed water level was set to 0 m (zhydrograph). The initial conditions are a fixed water surface elevation of 100 m in the reservoir, and dry conditions in the rest of inland domain (region defined ICs). The initial velocity was set to 0.0 m/s in the entire domain. In accordance with (Hervouet and Petitjean, 1999), the Manning's friction coefficient was set to 0.033 $m^{-1/3}s$ for the whole domain. The CFL number was set to 0.9.

The simulated maximum water levels are compared to the 17 field observations in Fig. 8, with overall good agreement. The average relative error is 7.15%, with the largest observed at P13, with an overestimation of 30.6%. To highlight the effects of the topographical approximations of the Digital Elevation Model (and hence of the computational mesh), we compared recorded and simulated water level values as follows. For each punctual maximum water level recorded in field observations (blue triangles in Fig. 8), we compared the maximum simulated values of the spatial mean, maximum and minimum among the computational cell containing the observation point and its three neighbours (black and red series in Fig. 8). We expect lower discrepancies between recorded and simulated values where the numerical values, hence the topographical elevations, are spatially homogeneous.

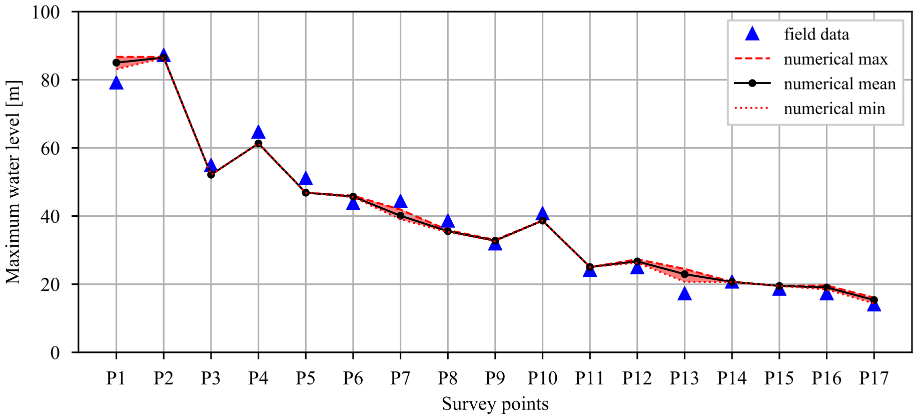


Fig. 8. T1: Malpasset dam-break wave maximum water level. Numerical values are compared to the field data at the survey points P1 to P17: simulated values are given as maximum, minimum (red lines) and mean value (black line) of the computational cell containing the survey point and its three neighbours.

As a matter of fact, points P1, P7 and P13 (Fig. 8) have the large discrepancy between measured and simulated values, but also the largest spatial variability of the numerical values (red shaded area). This suggests that such discrepancies relates more to the local topographical approximations of the DTM rather than to the numerical model.

Observed and simulated times of flood arrival are given in Table 5. Simulated values are in good agreement with measured ones for all the electrical transformer stations (ET). Simulated arrival times have a maximum relative error of 3.8% for ET B, corresponding to an absolute delay of 47 s. It is worth mentioning that the friction value influences the simulated arrival times.

7.2. *T2: Propagation of a sediment bore*

Scope of the test is to assess the robustness of the de-coupled hydro-morphodynamic solver approach, particularly when simulating the sediment transport over a transcritical flow. This represents a critical test, especially when adopting de-coupled approaches (e.g. Cordier et al., 2011). Moreover, the simulation tests the morphological solver capability in well reproducing the dynamics of an advancing sediment bore.

In this test case, the flume experiment proposed in (Bellal et al., 2003, run 2) is reproduced numerically. The computational domain is a composed by a straight 6.9 times 0.5 m channel, representing the lower part of the original experimental flume, and it is discretized with 24,612 triangular elements. The sediment has a characteristic diameter of 1.65 mm. The water and sediment discharge at the upstream boundary are set to 0.012 $m^3\ s^{-1}$ (uniform BC) and 0.196 $m^3\ s^{-1}$ (with porosity, sedimentograph BC) respectively. The flume is at initial uniform flow conditions, characterized by a supercritical flow. At $t = 0$ s, a fixed water level of 0.2093 m is imposed at the downstream boundary (zhydrograph) and sediment transport out of the domain is stopped (wall). This results in the formation of an hydraulic jump moving upstream in the flume, and a subsequent downstream propagation of a sediment bore. The CFL number is set to 0.9, the bed porosity is assumed constant and equal to 0.42, and the simulation duration is 500 s.

Fig. 9 shows the initial and final profiles of the simulated bed and water elevations. The solver reproduces well the sharp transition between super- and sub-critical flow conditions. The position of the sediment front in time is shown in Fig. 10, with good agreement between simulated and experimental values.

7.3. *T3: Dam-break over a mobile bed with a sudden enlargement*

Scope of the test is to assess code robustness in simulating sediment transport at wet-dry interface, and the accuracy in reproducing scour/deposition patterns. The experiment illustrated in (Goutiere et al., 2011) represents a well-know morphodynamic test for numerical models (e.g. Juez et al., 2014; Siviglia et al., 2013; Soares-Frazão and Zech, 2011). The domain consists of a flat flume with a non-symmetrical sudden enlargement (Fig. 11). The bed is composed of a coarse uniform sand with a median diameter of $d_m = 1.82$ mm. The initial conditions are defined by an horizontal layer of fully saturated sand of thickness 0.1 m over the whole domain and an initial water storage of depth 0.25 m upstream of the dam, located at section $x = 3.0$ m (region defined ICs). At time $t = 0$ s, the dam is suddenly removed, resulting in the propagation of a dam break wave with consequent sediment transport.

The computational domain is discretized by unstructured triangular cells at different resolutions (follows in Table 6). Inviscid wall boundary conditions are set at the upstream and lateral domain boundaries, while a free-outflow condition (zero_gradient) and constant bed elevation (equilibrium) conditions are used at the downstream outlet. The Manning coefficient is set to 0.0167 $m^{-1/3}s$, the sediment density and porosity are set to 2680 kg/m^3 and 0.47, respectively. The sediment transport is evaluated with the MPM-like formula (Table 2), setting $\theta_{cr} = 0.0495$ for the critical Shields stress, $\alpha = 3.97$ and $m = 1.5$ for the remaining parameters (Wong and Parker, 2006). The CFL number is set to 0.9. The numerical simulations last 12 s.

The evolution of the water elevation during the simulation is shown in Fig. 12 for the six survey points. The simulated series show a fairly good agreement with the experimental values: the dam break wave arrival time is well captured and the maximum elevation values are comparable with the measured ones. Moreover, the simulated series show minor discrepancies with the experimental ones after the arrival of the first wave. As already pointed out by previous works (Siviglia et al., 2013; Xia et al., 2010), discrepancies are due to the extremely complex flow pattern generate by multiple wave reflections while simulation proceeds in time, which potentially generate tri-dimensional flow structures. Nevertheless, obtained series are coherent with the ones of (Siviglia et al., 2013), where a second-order accuracy model was employed.

Numerical bed elevations after 12 s are compared to the experimental results in Fig. 13. The simulated scour and deposition patterns are well reproduced. The magnitude of the scour at cross section CS1 (at $y \approx 0.25$ m) matches well, with an underestimation of the deposition pattern (at $y \approx 0.35$ m). At cross section CS2, the simulated deposition magnitude matches well with the experimental one, but with a small shift toward the lateral boundary.

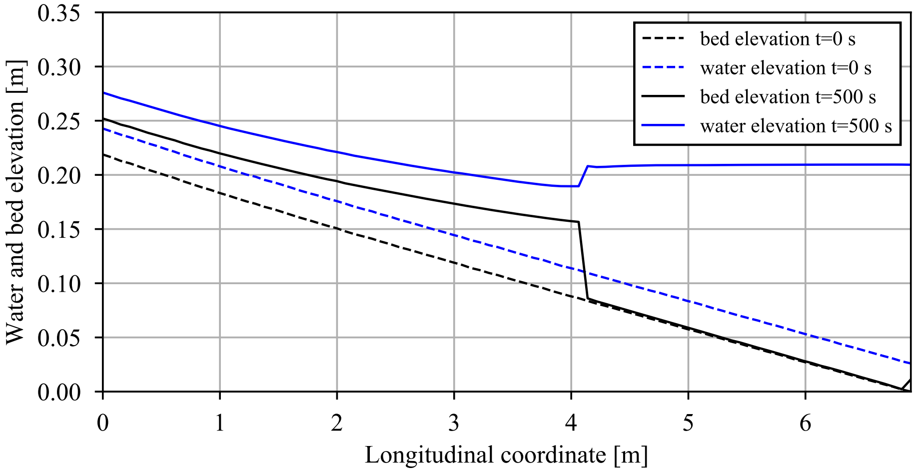


Fig. 9. T2: propagation of a sediment bore. Initial (dashed lines) and final (solid lines) longitudinal profiles of bed elevation (black) and water elevation (blue) for the propagation of a sediment bore test.

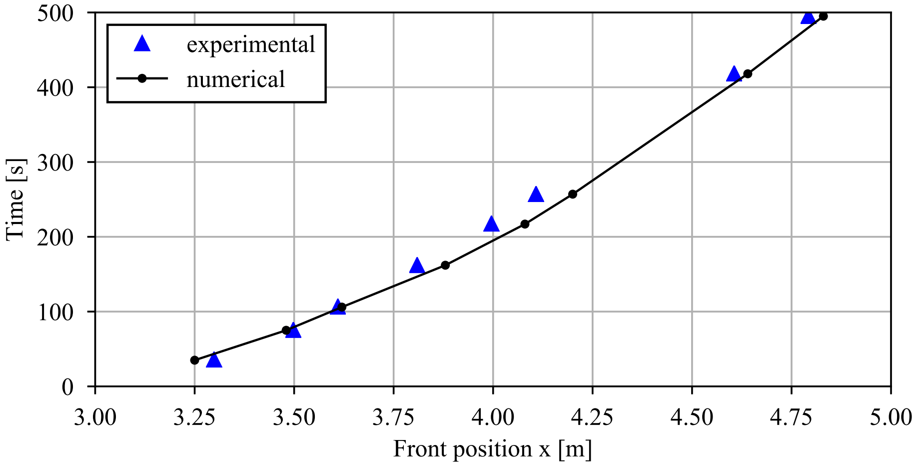


Fig. 10. T2: evolution in time of the sediment front position. Blue triangles are the experimental values (Bellal et al., 2003), while the black line is the numerical solution.

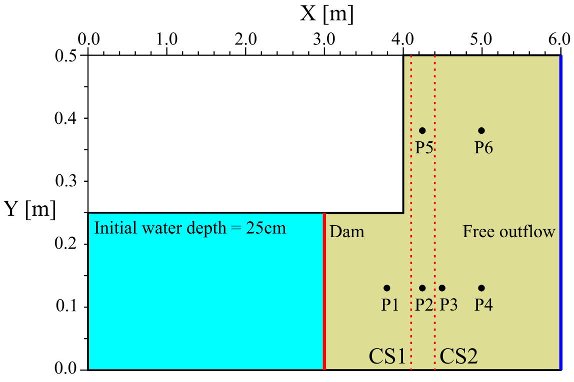


Fig. 11. T3: planar view of the dam-break over mobile bed setup (deformed axis). Experimental and numerical results are compared at survey points P1 to P6 during the simulation, and at cross-sections CS1 and CS2 at the end of simulation.

Table 6
Number of computational cells for the performance and scalability benchmarks.

Mesh ID	T1	T3	T6
1	24 945	27 444	24 388
2	52 102	47 187	49 155
3	101 417	109 344	98 163
4	499 060	218 912	196 829

7.4. T4: Scour and deposition on a channel bend

The scope of the simulation is to test the correct reproduction, both in term of positioning and magnitude, of a river point bar generated by a channel bend. In this test we numerically reproduced one experiment from (Yen and Lee, 1995), already adopted as morphodynamic benchmark test (e.g. Villaret et al., 2013). The flume is U-shaped, with a bend of 180° having a constant radius along the center line of $R_c = 4$ m. The cross section is rectangular with width $W = 1$ m and slope $S = 0.2\%$. The two straight reaches before and after the bend are 11.5 m long. The median diameter of the bed material was $d_m = 1$ mm. In the experimental run the flume was fed with a simplified (triangular) flood hydrograph, having a base flow of 0.02 $\mathrm{m}^3\,\mathrm{s}^{-1}$, and peak flow of 0.053 $\mathrm{m}^3\,\mathrm{s}^{-1}$. The rising and falling limbs last 100 and 200 min, respectively. Afterwards a constant baseflow was kept for another 100 min. During the experimental run, a steady point bar in the inner side of the bed develops and grows, with a corresponding erosion on the outer side.

In the numerical setup, the domain is discretized with 24,523 computational cells. We set the porosity to 0.4, and used the MPM-like formula with parameters from (Wong and Parker, 2006), as in Table 2. The lateral slope factor N_l is equal to 1.4 and the curvature factor N_* is set to 11 (6 and 7). At the numerical domain boundaries uniform flow and equilibrium sediment transport conditions are imposed. The simulation, as the experimental run, lasts 400 min.

A planar view comparison between numerical and experimental run is depicted in Fig. 14. The final bed change with respect to the initial flat configuration Δz_b is scaled with the approaching (i.e. upstream reach) flow depth h_0. The magnitude of scours and depositions for the numerical run ranges between −0.75 and 0.75, matching fairly well with the experimental values. Also the positioning of the point bar, with the maximum deposition anticipating the middle of the bend (90°) is well reproduced numerically.

Fig. 15 shows the cross-sectional profile of the relative bed change for the numerical simulation and the experimental run in the middle of the flume bend (at 90°). The numerical profile reproduces well the experimental trend. This test demonstrates the software capability in simulating an unsteady morphological process, i.e. a point bar development in a meander during a flood. Such process can be well reproduced only by implementing suitable corrections of the sediment transport direction due to gravity and curvature, as presented in §3.2.1.

7.5. T5: Steady scalar discontinuity with two diverging hydrodynamic waves

The test assesses the correct advection of scalar concentration. This is assessed with a challenging test: a steady discontinuity of a scalar concentration subjected to strongly variable flow conditions. The chosen test is an idealized one-dimensional problem, but nevertheless it is particularly challenging for a pletora of numerical schemes (Toro, 2001). The domain is a simple, flat-bed, channel 100 m long and 0.1 m wide. The domain is deliberately chosen very narrow, to mimic a 1D setup, given that the exact solution of the problem is available in one-dimension case. As initial conditions (region defined ICs), the water depth h is set even in all the domain, whilst the initial longitudinal specific discharge q_x and the concentration of a generic scalar ϕ present a discontinuity:

$$\left\{\begin{array}{lclllll} q_x = & -3.0\ \mathrm{m}^2/\mathrm{s} & \text{if } x & < 50\ \mathrm{m}, & q_x = & 3.0\ \mathrm{m}^2/\mathrm{s} & \text{otherwise}, \\ \phi = & 1.0 & \text{if } x & < 50\ \mathrm{m}, & \phi = & 0.0 & \text{otherwise}, \\ h = & 1.0\ \mathrm{m} & \forall x. & & & & \end{array}\right. \tag{22}$$

The domain is discretized with 1362 computational cells, lateral

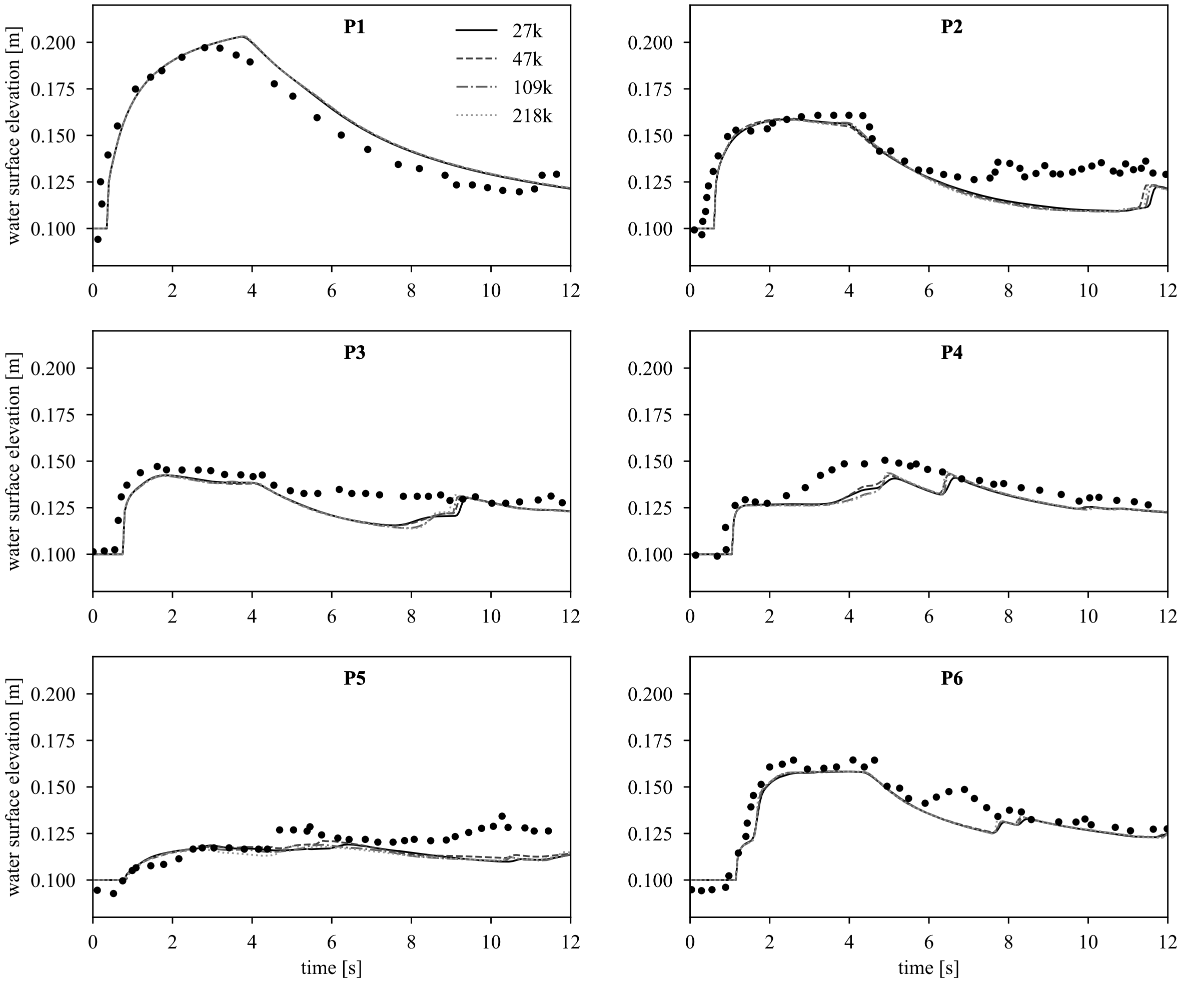


Fig. 12. T3: experimental and numerical water elevation at six survey points. Sub-panels represent the survey points P1 to P6 as in Fig. 11: experimental points are given as full circles, whilst numerical results as lines, with four different mesh sizes (27k, 47k, 109k, 218k computational cells).

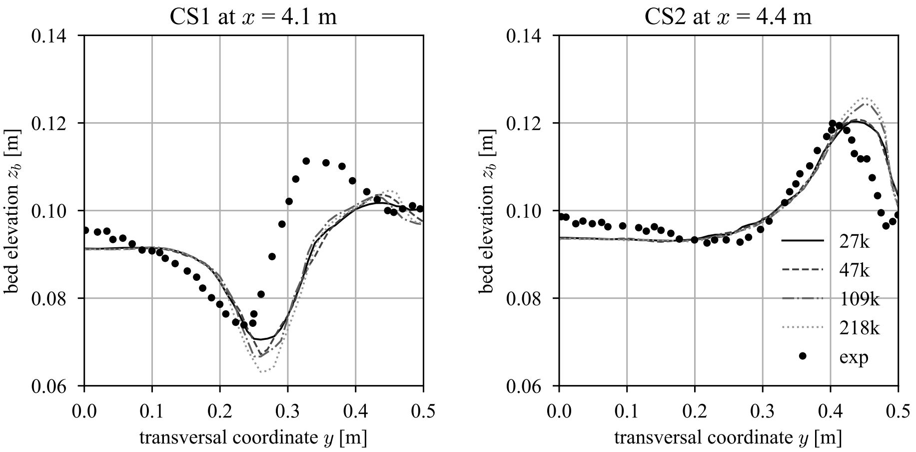


Fig. 13. T3: experimental and numerical bed elevation at different cross-sections. Sub-panels represent cross-section CS1 and CS2 as in Fig. 11: experimental points are given as full circles, whilst numerical results as lines, with four different mesh sizes (27k, 47k, 109k, 218k computational cells).

walls are reflective and inviscid, whilst transparent boundary conditions (zero_gradient) are set at beginning and end of the channel, for both hydrodynamic and scalar transport modules. The CFL is set to 0.95 and the simulation timeout is $t = 2.5$ s. As the simulation starts, two strong rarefaction waves start diverging from the center of the domain towards the two extremities, suddenly forming a water depression in the center. Despite the strong unsteadiness of the hydrodynamic quantities during the simulation, a steady contact wave persists in the domain, avoiding the scalar quantity to mix in the domain.

The numerical solution at simulation timeout is compared with the exact solution of the problem in Fig. 16. The hydrodynamic exact solution is obtained by resolving the two-rarefaction Riemann Problem

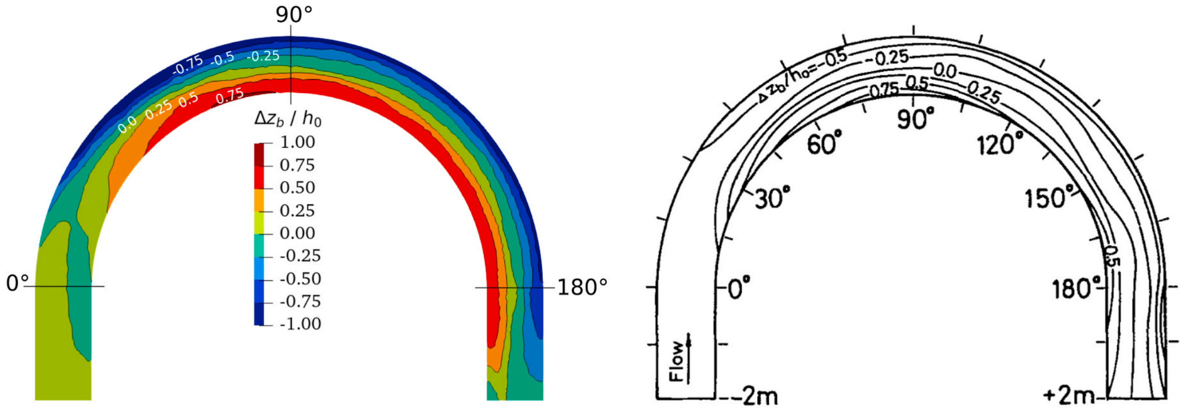


Fig. 14. T4: scour and deposition on a channel bend. Planar view of the relative bed change $\Delta z_b/h_0$, i.e. the difference between final and initial bed elevation (Δz_b) over the approaching flow depth (h_0). Numerical results on the left, laboratory results from (Yen and Lee, 1995) on the right.

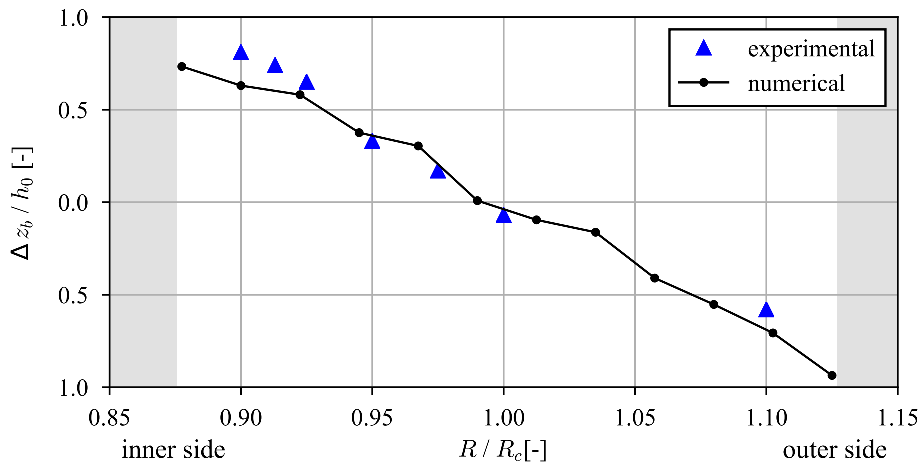
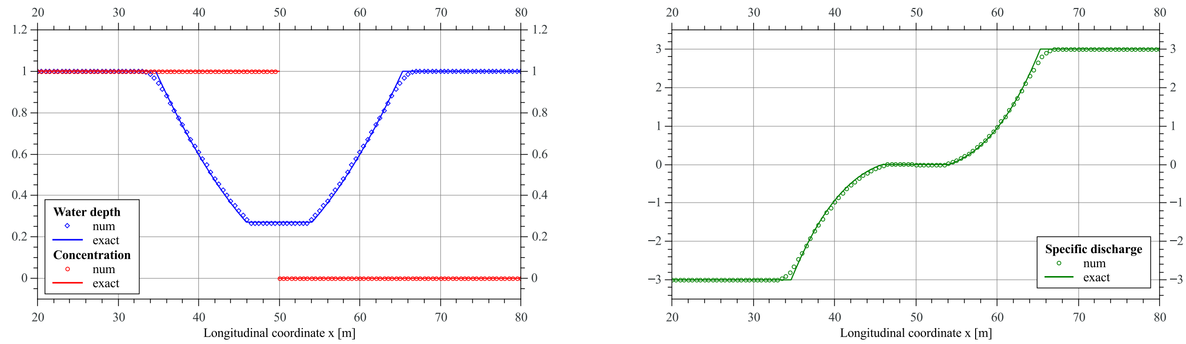


Fig. 15. T4: scour and deposition at one cross-section. Cross-sectional view (at 90°) of the relative bed change $\Delta z_b/h_0$ (difference between final and initial bed elevation over approaching flow depth h_0). In the x-axis the radial coordinate R is scaled with the center-line curvature radius R_c; black line is the numerical solution, blue triangles are from the laboratory experiment of (Yen and Lee, 1995).



(a)

(b)

Fig. 16. T5: comparison between numerical and exact solution of a steady contact wave. Solution is given at timeout $t = 2$ s. Exact solutions are depicted with lines, whereas numerical values are symbols, decimated for the sake of visualization. Panel **(a)** shows the water depth [m] (blue diamonds) and the scalar volumetric concentration (red circles), panel **(b)** shows the specific discharge [m^2/s] on the x direction (green circles).

(Toro, 2001), whilst the exact solution for the scalar advection is identical to the given initial conditions. Fig. 16 underlines how the numerical solution correctly approximates the exact solution in all the domain sections. The scalar discontinuity is perfectly maintained throughout the simulation, confirming the accurate resolution of the steady contact wave.

7.6. T6: Scalar advection and diffusion in a dam-break over a complex domain

With this test we assess the solver capability in correctly preserving the liquid and scalar species mass when simulating the advection and diffusion of species during a dam-break phenomena. The test is particularly harsh, due to the presence of fast wetting-drying fronts and multiple discontinuous flow regions.

The test is an *ad-hoc* setup inspired to a common benchmark for hydrodynamic codes (e.g. Brufau et al., 2002; Vanzo et al., 2016). The domain is composed by a rectangle [0; 75] × [−15; 15] m. The bottom $\eta(x, y)$ is fixed during the simulation, and defined as

$$\begin{aligned} &\eta(x,y) = \max(0, \eta_1(x,y), \eta_2(x,y), \eta_3(x,y)), \text{ with} \\ &\eta_1(x,y) = 1 - \frac{1}{8}\sqrt{(x-30)^2 + (y+9)^2}, \\ &\eta_2(x,y) = 1 - \frac{1}{8}\sqrt{(x-30)^2 + (y-9)^2}, \\ &\eta_3(x,y) = 3 - \frac{3}{10}\sqrt{(x-47.5)^2 + y^2}. \end{aligned} \tag{23}$$

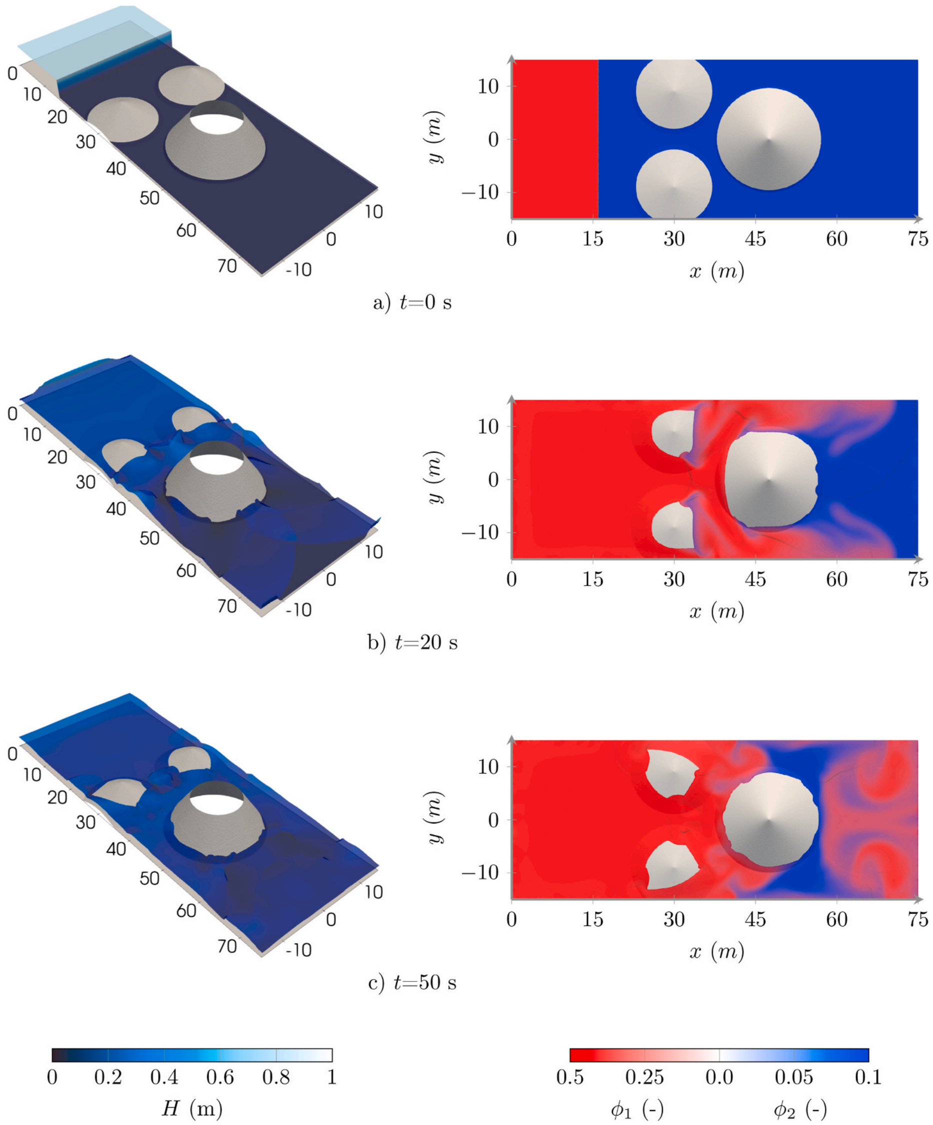
The initial conditions are given by:

$$\begin{cases} h = & 1.0\ \text{m} & \text{if } x < 16\ \text{m}, & h = & 0.125\ \text{m} & \text{otherwise}, \\ \phi_1 = & 0.5 & \text{if } x < 16\ \text{m}, & \phi_1 = & 0 & \text{otherwise}, \\ \phi_2 = & 0 & \text{if } x < 16\ \text{m}, & \phi_2 = & 0.1 & \text{otherwise}, \\ q_x = & q_y = 0\ \text{m}^2/\text{s} & \forall (x,y), & & & \end{cases} \tag{24}$$

presenting a virtual dam at $x = 16$ m separating two discontinuous volumes of water and scalar mass. Here the domain is discretized with 492,277 triangular cells with a maximum of characteristic length of 0.1 m. The hydrodynamics setup features reflective wall boundaries, a CFL of 0.95 and frictional sources compatible with a Manning coefficient of $n = 0.01\ \text{m}^{-1/3}\text{s}$. The scalar setup features two initially unmixed species, both with a constant and isotropic diffusion coefficient $K_c = 0.25\ \text{m}^2/\text{s}$. Fig. 17a illustrates the hydrodynamic (left) and scalar solutions (right) at the initial condition ($t = 0$ s).

At simulation start, the virtual dam collapses instantaneously, with an advancing wave that overtops the two small lateral humps, fully circumvents the larger hump and reaches the opposite wall in about $t = 20$ s. At this time the interface between the two species, in what would otherwise be a contact discontinuity on flat topography, is still lagging by approximately 15 m (Fig. 17b). At this point, the reflected bores propagate upstream and further mix both species, symmetrically around the x axis. By $t = 50$ s these bores overcome the two smaller obstacles and propagate upstream on flat ground (Fig. 17c).

After a continuous sloshing and interaction of reflected waves, topography and lateral walls, the friction sources gain relevance and dissipate most of the kinetic energy in the flow, with a near-static



a) t=0 s

b) t=20 s

c) t=50 s

Fig. 17. T6: dam break over complex topography. 3D visualization of the water surface elevation H (left panels) and planar view of concentration distribution (ϕ_1 and ϕ_2) for both scalar species 1 and 2 (right panes). Subpanels (a,b,c) report different simulation timeout.

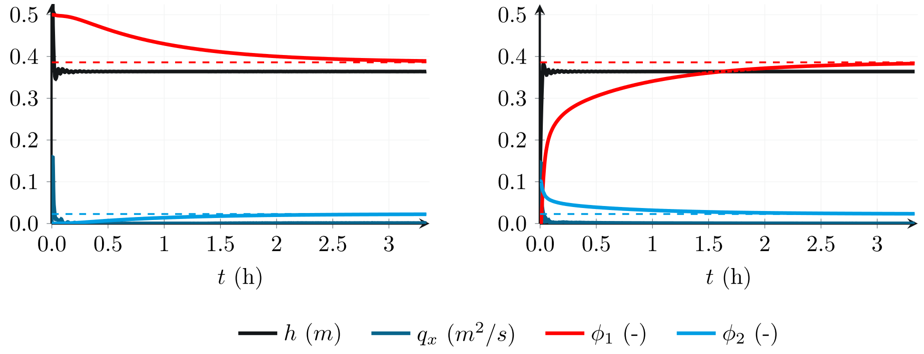


Fig. 18. T6: time series at two locations. Evolution of hydrodynamic quantities (h and q_x) and scalar concentrations ϕ_1 (red) and ϕ_2 (blue) in time at position $(x, y) = (7.5, 0.0)$ m (left) and $(x, y) = (67.5, 0.0)$ m (right). Dashed lines represent theoretical scalar concentration values, at rest.

solution being obtained at approximately $t = 20$ min. The scalars continue to mix, now due mostly to molecular diffusion, in what is a much slower process, that only vanishes at around 3.5 h as both scalars become fully homogeneous across the domain (Fig. 18).

The model is fully conservative, with the total liquid and scalar mass preserved during the entire simulation. As the simulation approaches the lake-at-rest conditions, the observed quantities correctly converge to their resting values of $h = 0.364$ m, $\phi_1 = 0.386$ and $\phi_2 = 0.023$ (Fig. 18).

7.7. Performance and scalability

The performance and scalability of the software depends not only on the implemented parallelization strategies but also on the physical model to be reproduced. In general, models with only few simulated physical processes, are likely to show higher computational performances. To test BASEMENT's computational performance, we selected the benchmarks T1 (hydrodynamic), T3 (morphodynamic) and T6 (scalar advection-diffusion).

Each of the selected numerical experiments (T1, T3, T6) has been conducted with four different computational meshes. The sizes of these meshes are given in Table 6 and have been chosen to cover a broad range of spatial resolutions, ranging from thousands to hundred of thousands computational cells. The simulations have been run with a set of computational backends. In particular, CPU-based simulations have been performed on an Intel Xeon Gold 6154 (3.00 GHz) workstation equipped with 36 cores (two sockets with 18 physical cores each), whilst GPU-based simulations have been run on three GPUs (GeForce GTX 1050 Ti, GeForce GTX 1080 Ti, and Tesla P100; see Table 7 for the main characteristics). Moreover, the simulations have been benchmarked in both single and double precision mode. The GPUs were integrated in a workstation with a 32-core Intel Xeon Gold 5218 (2.30 GHz) processor (two sockets with 16 physical cores each).

For a given mesh size, the speedup achieved by a parallelized backend p is computed using the formula speedup $= T_s/T_p$, where T_s (T_p) is the total computational time used by the serial (parallel) backend. The results for all the investigated benchmarks are depicted in Fig. 19.

As anticipated, the speedup depends on the simulated processes. Comparing the speedup values among different benchmarks in Fig. 19, cases T1 (hydrodynamics) and T3 (morphodynamics) show on average higher values than T6 (advection-diffusion). Such results are expected, given the increased complexity (number of equations and operations to be solved) of T6.

The performance benefits of BASEMENT's parallelization can be evaluated in more detail by comparing the speedup values along the vertical axis of the plots. In the following we focus on benchmark T3 (morphodynamics) which shows an "intermediate" scalability among the three benchmarks (Fig. 19b). Looking at the mesh with 47k elements as an example, the CPU-based family (i.e. OpenMP on multiple cores) shows a speedup efficiency (i.e. Speedup over number of cores) of 87% with 2 CPU cores (Speedup = 1.7), and of 58% with 32 cores (Speedup = 18.5), with an average efficiency of 74%. BASEMENT performs even better on some GPU cards. The least performing card (GTX 1050 Ti with double precision) has a speedup of about 8. However, note that speedup jumps to 20 when using the single precision version. Overall, the speedup provided by the tested GPUs ranges between 7 and 60.

The benchmarks in Fig. 19 also show how the maximum speedup changes with mesh size, computational backend and simulated processes. For all three cases, the CPU-based parallelizations show a mild speedup increase with an increasing number of computational cells. The dependency on the mesh size is slightly more pronounced when the number of computational cores is increased. This reflects the fact that CPU-based solutions have shared memory and minimal overhead (for multi-threading handling), thus the domain size (i.e. the data size) does not represent a potential performance bottleneck. Focusing on T3 (Fig. 19b), the parallelization efficiency for 2–4 CPU cores is almost constant for all mesh sizes and above 80%. On the other hand, the efficiency for 16–32 CPU cores is larger than 70% only for mesh 4 (218k).

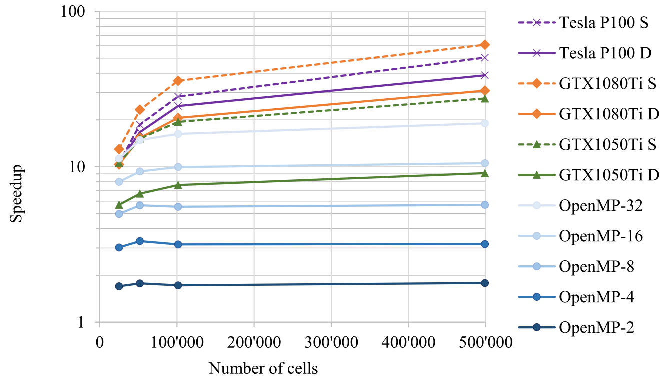
The speedup of the GPU-accelerated solutions shows not only a more marked dependency on the problem size, but also on the simulated processes. In benchmark T1 (Fig. 19a), the speedup clearly increases with problem size. This can be explained with the overhead of GPU parallelization, which becomes more and more negligible with increasing domain size. Conversely, benchmark T6 (Fig. 19c) shows little impact of the domain size on the speedup. In this case the scalability is limited by data transfers, i.e. data bandwidth: test T6 has the largest amount of data (compared to test T1 and T3), because of the simulation of the 5 species (§3.3). Due to the current design of the OP2 framework, dataset dimensions (e.g. the maximum number of species) are statically assigned at compile time. Increasing this maximum value would be at the expense of scalability of the computational backends.

It is worth remarking that the GPU-accelerated backends show an average speedup difference greater than 10 between the single and double precision versions. Of course, the adequate choice depends on the requirements of the specific application.

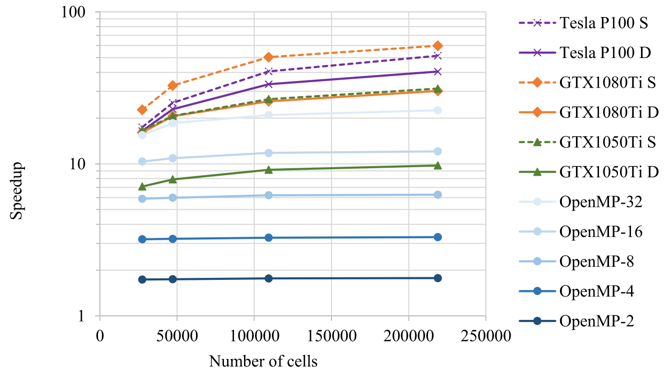
The analysis above shows BASEMENT's performance on different computational backends and underlines the differences when simulating different processes. The results summarized in Fig. 19 can also serve as a guideline for the interested reader/user when choosing an appropriate computational configuration for a given application. Finally it is worth

Table 7
Characteristics of the benchmarked Nvidia GPU cards. Further specifications available at www.nvidia.com/en-gb/geforce/10-series/ and www.nvidia.com/en-gb/data-center/tesla-p100/.

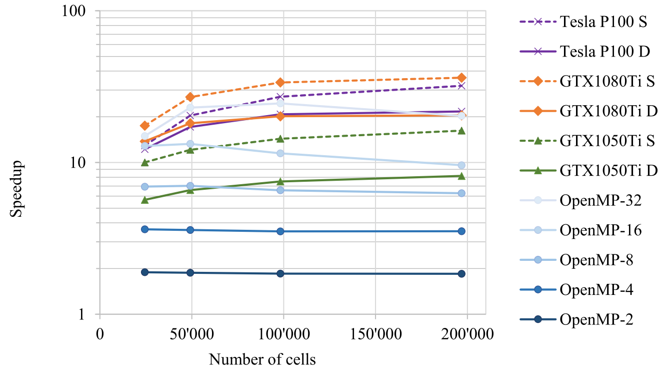
Type	Architecture	CUDA Cores
GeForce GTX 1050 Ti	Pascal	768
GeForce GTX 1080 Ti	Pascal	3584
Tesla P100	Pascal	3584



(a) T1



(b) T3



(c) T6

Fig. 19. Speedup of backends for varying mesh size, test cases T1, T3 and T6. The benchmarks were executed with different degrees of parallelism on the CPU (using OpenMP) and the GPU. The final “S" and “D" in GPU series denote single and double precision, respectively.

highlighting that all the tested hardware configurations can be easily installed in standard office workstations.

8. Conclusions

In this paper we introduced the main features of BASEMENT version 3, a freeware tool for river simulation. BASEMENT allows the simulation of a wide variety of hydro-, morphodynamic, and scalar advection-diffusion scenarios. As illustrated with the test cases, the software is able to efficiently capture large scale hydrodynamic processes modelled with several hundreds of thousand elements in good agreement with the measurements. On the opposite end of the spectrum, the morphological solver is able to handle demanding sediment transport scenarios well, albeit with known limitations. With the scalar advection-diffusion module a further set of physical processes such as the fate of river pollutants can be accurately modelled.

The impact of this flexibility on the software performance is minimized by activating modules on request in BASEMENT’s pre-simulation step. The advantage of this approach is that only the required kernels are scheduled for execution. This, together with OP2’s ability to generate executable code for both multi-core CPUs and GPUs, permit BASEMENT to scale with both available features and available computational power. Such advantages are reflected in the presented benchmarks. Given a large enough domain, the software shows a good parallel efficiency on the CPU and an even higher speedup when using GPUs.

The BASEMENT project is in continuous advancement to optimize and include further features in the existing basic modules. As an example, the modelling of the sediment transport in presence of non uniform sediment size and the simulation of water temperature dynamics are in implementation phase. On the other hand, efforts are dedicated also to develop novel modelling solutions for river processes such as the biomorphodynamic feedback between vegetation and sediment transport. Table 2 of the Supplementary Material provides an overview of under development features. The modularity of the development framework allows also for further refinement of single specific numerical solvers and the implementation of high-order schemes when needed.

Overall, the combination of different river processes that can be modelled, the computational efficiency, the flexibility in the backend choice, but also the availability of a light GUI, make BASEMENT a valuable tool for a broad family of river modelers in both Academia and Practice.

Funding

The development of the software BASEMENT is financially supported by the Swiss Federal Office for the Environment (FOEN).

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.

Acknowledgments

The Authors thank the reviewers for the fruitful comments and suggestions.

The Authors greatly thank the many former collaborators and developers within the BASEMENT project. Particular thanks to Aurélie Koch, for her valuable contribution in testing and documenting the software.

The design of BASEMENT was conducted by DFV, DV, SP, LV. The software prototyping, development and implementation was done by DV, SP, LV, MB, MW, with the coordination and supervision of DFV and AS. Implementation and testing of the reported features was done by MB, DV, MW and DC. The manuscript was conceptualized by DV, AS and DFV, and drafted by DV, with support of MB and MW. All Authors contributed to the manuscript review.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.envsoft.2021.105102.

References

Afzal, A., Ansari, Z., Faizabadi, A.R., Ramis, M.K., 2016. Parallelization strategies for computational fluid dynamics software: state of the art review. Arch. Comput. Methods Eng. 24 (2), 337–363. https://doi.org/10.1007/s11831-016-9165-4.

Armanini, A., 2018. Principles of River Hydraulics. Springer.

Beckers, F., Heredia, A., Noack, M., Nowak, W., Wieprecht, S., Oladyshkin, S., 2020. Bayesian calibration and validation of a large-scale and time-demanding sediment transport model. Water Resour. Res. 56 (7), e2019WR026966 https://doi.org/10.1029/2019wr026966.

Bellal, M., Spinewine, B., Savary, C., Zech, Y., 2003. Morphological evolution of steep-sloped river beds in the presence of a hydraulic jump: experimental study. In: XXX IAHR Congress, Citeseer, 133–140.

Bezzola, G.R., 2002. Fliesswiederstand und Sohlenstabilität natürlicher Gerinne, Ph.D. thesis. Eidgenössische Technische Hochschule Zürich.

Brewer, S.K., Worthington, T.A., Mollenhauer, R., Stewart, D.R., McManamay, R.A., Guertault, L., Moore, D., 2018. Synthesizing models useful for ecohydrology and ecohydraulic approaches: an emphasis on integrating models to address complex research questions. Ecohydrology 11 (7), e1966. https://doi.org/10.1002/eco.1966. ISSN 19360584.

Brodtkorb, A.R., Sætra, M.L., Altinakar, M., 2012. Efficient shallow water simulations on GPUs: implementation, visualization, verification, and validation. Comput. Fluids 55 (1–12), 00457930. https://doi.org/10.1016/j.compfluid.2011.10.012.

Brufau, P., Vázquez-Cendón, M.E., García-Navarro, P., 2002. A numerical model for the flooding and drying of irregular domains. Int. J. Numer. Methods Fluid. 39 (3), 247–275. https://doi.org/10.1002/fld.285.

Carraro, F., Vanzo, D., Caleffi, V., Valiani, A., Siviglia, A., 2018. Mathematical study of linear morphodynamic acceleration and derivation of the MASSPEED approach. Adv. Water Resour. 117, 40–52. https://doi.org/10.1016/j.advwatres.2018.05.002.

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. Compt. Rendus Mec. 339 (2–3), 165–184. https://doi.org/10.1016/j.crme.2010.12.004. ISSN 16310721.

Chen, X., Ma, J., Dey, S., 2010. Sediment transport on arbitrary slopes: simplified model. J. Hydraul. Eng. 136 (5) https://doi.org/10.1061/(ASCE)HY.1943-7900.0000175, 311–317, ISSN 0733-9429.

Cordier, S., Le, M., de Luna, T.M., 2011. Bedload transport in shallow water models: why splitting (may) fail, how hyperbolicity (can) help. Adv. Water Resour. 34 (8), 980–989. https://doi.org/10.1016/j.advwatres.2011.05.002.

Costabile, P., Macchione, F., 2015. Enhancing river model set-up for 2-D dynamic flood modelling. Environ. Model. Software 67. https://doi.org/10.1016/j.envsoft.2015.01.009, 89–107, ISSN 13648152.

Crosato, A., Saleh, M.S., 2011. Numerical study on the effects of floodplain vegetation on river planform style. Earth Surf. Process. Landforms 36 (6), 711–720. https://doi.org/10.1002/esp.2088.

Dazzi, S., Vacondio, R., Palù, A.D., Mignosa, P., 2018. A local time stepping algorithm for GPU-accelerated 2D shallow water models. Adv. Water Resour. 111, 274–288. https://doi.org/10.1016/j.advwatres.2017.11.023.

Dazzi, S., Vacondio, R., Mignosa, P., 2020. Internal boundary conditions for a GPU-accelerated 2D shallow water model: implementation and applications. Adv. Water Resour. 137, 103525. https://doi.org/10.1016/j.advwatres.2020.103525.

Dugdale, S.J., Hannah, D.M., Malcolm, I.A., 2017. River temperature modelling: a review of process-based approaches and future directions. Earth Sci. Rev. 175, 97–113. https://doi.org/10.1016/j.earscirev.2017.10.009.

Duran, A., Liang, Q., Marche, F., 2013. On the well-balanced numerical discretization of shallow water equations on unstructured meshes. J. Comput. Phys. 235, 565–586. https://doi.org/10.1016/j.jcp.2012.10.033. ISSN 00219991.

Engelund, F., 1974. Flow and bed topography in channel bends. J. Hydraul. Div. ASCE 100 (11), 1631–1648.

Engelund, F., Hansen, E., 1972. A Monograph on Sediment Transport in Alluvial Streams. Teknisk Forlag, Copenhagen.

Exner, F.M., 1925. Ueber die Wechselwirkung zwischen Wasser und Geschiebe in Fluessen, Tech. Rep., Akademie der Wissenschaften. Mathematische Naturwissenschaft Abt. IIa, Wien, Austria.

García-Feal, O., González-Cao, J., Gómez-Gesteira, M., Cea, L., Domínguez, J., Formella, A., 2018. An accelerated tool for flood modelling based on Iber. Water 10 (10), 1459. https://doi.org/10.3390/w10101459.

Giles, M.B., Mudalige, G.R., Sharif, Z., Markall, G., Kelly, P.H.J., 2012. Performance analysis and optimization of the OP2 framework on many-core architectures. Comput. J. 55 (2), 168–180. https://doi.org/10.1093/comjnl/bxr062. ISSN 0010-4620.

Giles, D., Kashdan, E., Salmanidou, D.M., Guillas, S., Dias, F., 2020. Performance analysis of Volna-OP2 – massively parallel code for tsunami modelling. Comput. Fluids 209, 104649. https://doi.org/10.1016/j.compfluid.2020.104649.

Gilvear, D.J., Greenwood, M.T., Thoms, M.C., Wood, P.J. (Eds.), 2016. River Science. John Wiley & Sons, Ltd. https://doi.org/10.1002/9781118643525.

Goutiere, L., Soares-Frazão, S., Zech, Y., 2011. Dam-break flow on mobile bed in abruptly widening channel: experimental data. J. Hydraul. Res. 49 (3), 367–371. https://doi.org/10.1080/00221686.2010.548969.

Graf, W.H., 1966. On the determination of the roughness coefficient in natural and artificial waterways, International Association of Scientific Hydrology. Bulletin 11 (1), 59–68. https://doi.org/10.1080/02626666609493443.

Grass, A.J., 1981. Sediment Transport by Waves and Currents, University College. Dept. of Civil Engineering, London.

Guan, M., Wright, N., Sleigh, P., Ahilan, S., Lamb, R., 2016. Physical complexity to model morphological changes at a natural channel bend. Water Resour. Res. 52 (8), 6348–6364. https://doi.org/10.1002/2015WR017917.

Hervouet, J.-M., Petitjean, A., 1999. Malpasset dam-break revisited with two-dimensional computations. J. Hydraul. Res. 37 (6), 777–788. https://doi.org/10.1080/00221689909498511.

Horváth, Z., Perdigão, R.A., Waser, J., Cornel, D., Konev, A., Blöschl, G., 2016. Kepler shuffle for real-world flood simulations on GPUs. Int. J. High Perform. Comput. Appl. 30 (4), 379–395. https://doi.org/10.1177/1094342016630800.

Hou, J., Kang, Y., Hu, C., Tong, Y., Pan, B., Xia, J., 2020. A GPU-based numerical model coupling hydrodynamical and morphological processes. Int. J. Sediment Res. 35 (4), 386–394. https://doi.org/10.1016/j.ijsrc.2020.02.005.

Ikeda, S., 1982. Lateral bed load transport on side slopes. J. Hydraul. Div. 108 (11), 1369–1373. https://doi.org/10.1061/jyceaj.0005937.

Juez, C., Murillo, J., García-Navarro, P., 2014. A 2D weakly-coupled and efficient numerical model for transient shallow flow and movable bed. Adv. Water Resour. 71, 93–109. https://doi.org/10.1016/j.advwatres.2014.05.014.

Juez, C., Lacasta, A., Murillo, J., García-Navarro, P., 2016. An efficient GPU implementation for a faster simulation of unsteady bed-load transport. J. Hydraul. Res. 54 (3) https://doi.org/10.1080/00221686.2016.1143042, 275–288, ISSN 0022-1686.

Jung, Y., Merwade, V., 2014. Estimation of uncertainty propagation in flood inundation mapping using a 1-D hydraulic model. Hydrol. Process. 29 (4), 624–640. https://doi.org/10.1002/hyp.10185.

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. Adv. Eng. Software 78 (1–15), 09659978. https://doi.org/10.1016/j.advengsoft.2014.08.007.

Lacasta, A., Juez, C., Murillo, J., García-Navarro, P., 2015. An efficient solution for hazardous geophysical flows simulation using GPUs. Comput. Geosci. 78, 63–72. https://doi.org/10.1016/j.cageo.2015.02.010.

Maddock, I., 1999. The importance of physical habitat assessment for evaluating river health. Freshw. Biol. 41 (2), 373–391. https://doi.org/10.1046/j.1365-2427.1999.00437.x.

Marcus, W.A., Fonstad, M.A., 2010. Remote sensing of rivers: the emergence of a subdiscipline in the river sciences. Earth Surf. Process. Landforms 35 (15), 1867–1872. https://doi.org/10.1002/esp.2094.

Meyer-Peter, E., Müller, R., 1948. Formulas for bed-load transport. In: IAHSR 2nd Meeting, Stockholm, Appendix 2. IAHR, –.

Morgan, J.A., Kumar, N., Horner-Devine, A.R., Ahrendt, S., Istanbullouglu, E., Bandaragoda, C., 2020. The use of a morphological acceleration factor in the simulation of large-scale fluvial morphodynamics. Geomorphology 356, 107088. https://doi.org/10.1016/j.geomorph.2020.107088.

Mudalige, G., Giles, M., Reguly, I., Bertolli, C., Kelly, P., 2012. OP2: an active library framework for solving unstructured mesh-based applications on multi-core and many-core architectures. In: Innovative Parallel Computing (InPar). IEEE. ISBN 978-1-4673-2633-9, 1–12, doi:10.1109/InPar.2012.6339594, 2012.

Nahorniak, M., Wheaton, J., Volk, C., Bailey, P., Reimer, M., Wall, E., Whitehead, K., Jordan, C., 2018. How do we efficiently generate high-resolution hydraulic models at large numbers of riverine reaches? Comput. Geosci. 119 (November 2017), 80–91. https://doi.org/10.1016/j.cageo.2018.07.001. ISSN 00983004.

Owens, J., Houston, M., Luebke, D., Green, S., Stone, J., Phillips, J., 2008. GPU computing. Proc. IEEE 96 (5), 879–899. https://doi.org/10.1109/jproc.2008.917757.

G. Parker, 1D sediment transport morphodynamics with applications to rivers and turbidity currents: E-book, Minneapolis, MN, available from:: http://hydrolab.illinois.edu/people/parkerg/morphodynamics_e-book.htm.

Pasternack, G., 2011. 2D Modeling and Ecohydraulic Analysis. University of California at Davis, California, ISBN 9781466320093.

Petaccia, G., Leporati, F., Torti, E., 2016. OpenMP and CUDA simulations of Sella Zerbino Dam break on unstructured grids. Comput. Geosci. 20 (5), 1123–1132. https://doi.org/10.1007/s10596-016-9580-5.

Peter, S.J., 2017. Dam Break Analysis under Uncertainty. Ph.D. thesis, ETH Zurich, Zurich, Switzerland. https://doi.org/10.3929/ethz-b-000209879.

Powell, K.G., Roe, P.L., Quirk, J., 1993. Adaptive-mesh algorithms for computational fluid dynamics. Algorithmic Trends in Computational Fluid Dynamics. Springer, New York, pp. 303–337. https://doi.org/10.1007/978-1-4612-2708-3_18.

Reguly, I.Z., Mudalige, G.R., Bertolli, C., Giles, M.B., Betts, A., Kelly, P.H., Radford, D., 2016. Acceleration of a full-scale industrial CFD application with OP2. IEEE Trans. Parallel Distr. Syst. 27 (5), 1265–1278. https://doi.org/10.1109/TPDS.2015.2453972. ISSN 1045-9219.

Reguly, I.Z., Giles, D., Gopinathan, D., Quivy, L., Beck, J.H., Giles, M.B., Guillas, S., Dias, F., 2018. The VOLNA-OP2 tsunami code (version 1.5). Geosci. Model Dev. (GMD) 11 (11), 4621–4635. https://doi.org/10.5194/gmd-11-4621-2018.

Rozovskii, I.L., 1961. Flow of Water in Bends of Open Channels, Academy of Science of the Ukrainian S.S. .R, Institute of Hydrology and Hydraulic Engineering.

Sanders, B.F., 2008. Integration of a shallow water model with a local time step. J. Hydraul. Res. 46 (4), 466–475. https://doi.org/10.3826/jhr.2008.3243.

Savage, J.T.S., Pianosi, F., Bates, P., Freer, J., Wagener, T., 2016. Quantifying the importance of spatial resolution and other factors through global sensitivity analysis of a flood inundation model. Water Resour. Res. 52 (11), 9146–9163. https://doi.org/10.1002/2015wr018198.

Sharma, D., Kansal, A., 2012. Assessment of river quality models: a review. Rev. Environ. Sci. Biotechnol. 12 (3), 285–311. https://doi.org/10.1007/s11157-012-9285-8.

Shields, A., 1936. Anwendungen der Ähnlichkeitsmechanik und der Turbulenzforschung auf die Geschiebebewegungen, Tech. Rep., Mitteilung der Preussischen Versuchsanstalt für Wasserbau und Schiffbau. Berlin, Deutschland.

Shimizu, Y., Nelson, J., Ferrel, K.A., Asahi, K., Giri, S., Inoue, T., Iwasaki, T., Jang, C.-L., Kang, T., Kimura, I., Kyuka, T., Mishra, J., Nabi, M., Patsinghasanee, S., Yamaguchi, S., 2019. Advances in computational morphodynamics using the International River Interface Cooperative (iRIC) software. Earth Surf. Process. Landforms 45 (1), 11–37. https://doi.org/10.1002/esp.4653.

Singh, J., Altinakar, M.S., Ding, Y., 2011. Two-dimensional numerical modeling of dam-break flows over natural terrain using a central explicit scheme. Adv. Water Resour. 34 (10), 1366–1375. https://doi.org/10.1016/j.advwatres.2011.07.007.

Siviglia, A., Crosato, A., 2016. Numerical modelling of river morphodynamics: latest developments and remaining challenges. Adv. Water Resour. 93, 1–3. https://doi.org/10.1016/j.advwatres.2016.01.005.

Siviglia, A., Stecca, G., Vanzo, D., Zolezzi, G., Toro, E.F., Tubino, M., 2013. Numerical modelling of two-dimensional morphodynamics with applications to river bars and bifurcations. Adv. Water Resour. 52, 243–260. https://doi.org/10.1016/j.advwatres.2012.11.010. ISSN 03091708.

Smart, G.M., Jaeggi, M.N.R., 1983. Sediment Transport on Steep Slopes, VAW-Mitteilung 64, Versuchsanstalt für Wasserbau,Hydrologie und Glaziologie (VAW). Zürich. ETH Zürich.

Smith, L.S., Liang, Q., 2013. Towards a generalised GPU/CPU shallow-flow modelling tool. Comput. Fluids 88, 334–343. https://doi.org/10.1016/j.compfluid.2013.09.018. ISSN 00457930.

Soares-Frazão, S., Zech, Y., 2011. HLLC scheme with novel wave-speed estimators appropriate for two-dimensional shallow-water flow on erodible bed. Int. J. Numer. Methods Fluid. 66 (8), 1019–1036. https://doi.org/10.1002/fld.2300. ISSN 02712091.

Stecca, G., Measures, R., Hicks, D., 2017. A framework for the analysis of noncohesive bank erosion algorithms in morphodynamic modeling. Water Resour. Res. 53 (8), 6663–6686. https://doi.org/10.1002/2017WR020756.

Talmon, A.M., Struiksma, N., Van Mierlo, M., 1995. Laboratory measurements of the direction of sediment transport on transverse alluvial-bed slopes. J. Hydraul. Res. 33 (4) https://doi.org/10.1080/00221689509498657, 495–517, ISSN 0022-1686.

Teng, J., Jakeman, A.J., Vaze, J., Croke, B.F.W., Dutta, D., Kim, S., 2017. Flood inundation modelling: a review of methods, recent advances and uncertainty analysis. Environ. Model. Software 90 (201–216). https://doi.org/10.1016/j.envsoft.2017.01.006. ISSN 13648152.

Toro, E., 2001. Shock-capturing Methods for Free-Surface Shallow Flows. Wiley and Sons Ltd, ISBN 0471987662. https://doi.org/10.1080/00221680309499935.

Toro, E.F., 2009. Riemann Solvers and Numerical Methods for Fluid Dynamics. Springer-Verlag GmbH, ISBN 978-3-540-49834-6. https://doi.org/10.1007/978-3-540-49834-6.

Toro, E.F., Spruce, M., Speares, W., 1994. Restoration of the contact surface in the HLL-Riemann solver. Shock Waves 4 (1), 25–34. https://doi.org/10.1007/bf01414629.

Vacondio, R., Palù, A.D., Mignosa, P., 2014. GPU-enhanced Finite Volume Shallow Water solver for fast flood simulations. Environ. Model. Software 57 (60–75). https://doi.org/10.1016/j.envsoft.2014.02.003. ISSN 1364-8152.

Vacondio, R., Dal Palù, A., Ferrari, A., Mignosa, P., Aureli, F., Dazzi, S., 2017. A non-uniform efficient grid type for GPU-parallel Shallow Water Equations models. Environ. Model. Software 88 (119–137). https://doi.org/10.1016/j.envsoft.2016.11.012. ISSN 13648152.

Valiani, A., Caleffi, V., Zanni, A., 2002. Case study: Malpasset dam-break simulation using a two-dimensional finite volume method. J. Hydraul. Eng. 128 (5), 460–472 doi:10.1061/(asce)0733-9429(2002)128:5(460).

van Rijn, L.C., 1989. Handbook Sediment Transport by Current and Waves. Delft Hydraulics Laboratory, Delft, The Netherlands.

Vanzo, D., 2015. Eco-hydraulic Quantification of Hydropeaking and Thermopeaking: Development of Modeling and Assessment Tools. Ph.D. thesis, University of Trento, Trento, Italy.

Vanzo, D., Siviglia, A., Toro, E.F., 2016. Pollutant transport by shallow water equations on unstructured meshes: hyperbolization of the model and numerical solution via a novel flux splitting scheme. J. Comput. Phys. 321, 1–20. https://doi.org/10.1016/j.jcp.2016.05.023.

Villaret, C., Hervouet, J.-M., Kopmann, R., Merkel, U., Davies, A.G., 2013. Morphodynamic modeling using the Telemac finite-element system. Comput. Geosci. 53, 105–113. https://doi.org/10.1016/j.cageo.2011.10.004.

Williams, R.D., Brasington, J., Hicks, D.M., 2016. Numerical modelling of braided river morphodynamics: review and future challenges. Geogr. Compass 10 (3), 102–127. https://doi.org/10.1111/gec3.12260.

Wong, M., Parker, G., 2006. Reanalysis and correction of bed-load relation of Meyer-Peter and Müller using their own database. J. Hydraul. Eng. 132 (11), 1159–1168 doi:10.1061/(asce)0733-9429(2006)132:11(1159).

Wyrick, J., Senter, A., Pasternack, G., 2014. Revealing the natural complexity of fluvial morphology through 2D hydrodynamic delineation of river landforms. Geomorphology 210, 14–22. https://doi.org/10.1016/j.geomorph.2013.12.013.

Xia, J., Lin, B., Falconer, R.A., Wang, G., 2010. Modelling dam-break flows over mobile beds using a 2D coupled approach. Adv. Water Resour. 33 (2), 171–183. https://doi.org/10.1016/j.advwatres.2009.11.004.

Yen, C., Lee, K.T., 1995. Bed topography and sediment sorting in channel bend with unsteady flow. J. Hydraul. Eng. 121 (8), 591–599 doi:10.1061/(asce)0733-9429(1995)121:8(591).

Zischg, A.P., Mosimann, M., Bernet, D.B., Röthlisberger, V., 2018. Validation of 2D flood models with insurance claims. J. Hydrol. 557, 350–361. https://doi.org/10.1016/j.jhydrol.2017.12.042.