Article:

Kesserwani, G. and Liang, Q. (2014) RKDG2 shallow-water solver on non-uniform grids with local time steps: Application to 1D and 2D hydrodynamics. Applied Mathematical Modelling, 39 (3-4). pp. 1317-1340. ISSN 1872-8480

https://doi.org/10.1016/j.apm.2014.08.009

RKDG2 shallow-water solver on non-uniform grids with local time steps: application to 1D and 2D hydrodynamics

Georges Kesserwani [a], Qiuhua Liang [b]

[a] *Department of Civil and Structural Engineering, University of Sheffield, Mappin St., Sheffield S1 3JD, UK*

[b] *School of Civil Engineering and Geosciences, Newcastle University, Newcastle upon Tyne NE1 7RU, UK*

Summary: This paper investigates Local Time Stepping (LTS) with the RKDG2 (second-order Runge-Kutta Discontinuous Galerkin) non-uniform solutions of the inhomogeneous SWEs (Shallow Water Equations) with source terms. A LTS algorithm – recently designed for homogenous hyperbolic PDE(s) – is herein reconsidered and improved in combination with the RKDG2 shallow-flow solver (LTS-RKDG2) including topography and friction source terms as well as wetting and drying. Two LTS-RKDG2 schemes that adapt 3 and 4 levels of LTSs are configured on 1D and/or 2D (quadrilateral) non-uniform meshes that, respectively, adopt 3 and 4 scales of spatial discretization. Selected shallow water benchmark tests are used to verify, assess and compare the LTS-RKDG2 schemes relative to their conventional Global Time Step RKDG2 alternatives (GTS-RKDG2) considering several issues of practical relevance to hydraulic modelling. Results show that the LTS-RKDG2 models could offer (depending on both the mesh setting and the features of the flow) comparable accuracy to the associated GTS-RKDG2 models with a savings in runtime of up to a factor of 2.5 in 1D simulations and 1.6 in 2D simulations.

Key-words: Shallow water equations; RKDG2 schemes; temporal adaptivity, non-uniform grids; conservative scheme; friction terms, computational efficiency, 1D and 2D hydraulic modelling.

*Corresponding author:
E-mails: g.kesserwani@sheffield.ac.uk; (G. Kesserwani) Qiuhua.Liang@ncl.ac.uk (Q. Liang)

1. Introduction

Explicit finite volume (FV) Godunov-type methods solving the shallow water equations (SWEs) are relevant to simulate hydraulic problems because they excel in a distinctive numerical formulation that incorporates widest range of spatial flow transients including discontinuities [1, 2]. These models have received numerous developments [3, 4] and some robust Godunov-type shallow water solvers have been successfully applied to support practical applications [5, 6], From an applied perspective, it is well-accepted that a *robust* Godunov-type numerical solver should be able to maintain its stability and consistency when a flow discontinuity develops, steep terrain gradients are present, a wet/dry front occurs, and high roughness values are combined with very small water depths. In spite of all these advances, it is still desirable to reduce the runtime of these explicit FV models. Parallelization has alleviated this issue using extrinsic parallel computers [7, 8] as well as the intrinsic shared-memory architecture of GPUs [9, 10]. However, the expanding power of parallelism remains rather stagnant and is not without problems as such [11]. For example, the small memory size of GPU computing cannot yet afford refined uniform-mesh simulations over large spatial domain coverage. Thus, the size of the system in terms of the number of cells remains a problem and, generally, to the interest of computational cost, allowing coarser cell size in a form of a non-uniform mesh is certainly a benefit.

In this context, it is expected that the efficiency of an explicit numerical scheme may suffer as the size of their time steps is restricted by the Courant-Friedrich-Lewy (CFL) stability condition [12]. This criterion provides the maximum allowable Global Time Step (GTS) permitted, which reduces proportional to a local increase in the velocity magnitude or a local decrease in the cell size. Few refined cells may dictate a restrictive time step on the whole non-uniform mesh, which may compromise by significantly longer runtimes. Temporal adaptivity, or a local time step method (LTS), whereby the solutions on different cell sizes are advanced by different time steps, may thus be beneficial to increase the computational efficiency. In so doing, in the FV context, a local first-order Godunov-type numerical formulation operating on a *small calculation stencil* appeared to be the most accommodating setup to favour temporal data exchange between those heterogeneous cells of the mesh [2]. However, first-order models are well-known to be diffusive – namely on coarse potions of the mesh. Thus, the design of a higher-order accurate Godunov-type shallow water model with a LTS algorithm could be beneficial and is the aim of this paper.

One convenient choice to do this is the use of a *local* spatial Discontinuous Galerkin (DG) approximations paired with an explicit multi-stage Runge-Kutta (RK) time mechanism (RKDG). RKDG schemes are reported to be convenient for (spatial) adaptive meshing techniques and demonstrated to deliver converged solutions on coarse meshes better than equally-accurate FV alternatives [13, 14]. An RKDG formulation can be regarded as an extension to the original FV Godunov philosophy in the sense that inter-elemental flux exchange evolves a finite series of local coefficients (spanning a polynomial solution) on each mesh element; thus allows keeping the *calculation stencil small* despite the desired order of accuracy. Practically speaking, the level of complexity, *robustness* and operational efficiency of an RKDG formulation drastically increase with the desired formal accuracy-order and the choice of the 2D mesh. A second-order accurate RKDG formulation (RKDG2) is therefore sensible to deliver a shallow water model that handles flow simulations involving topographic and friction effects, and flooding and drying processes [15-17]. Worth also mentioning the work of Wirasaet et al. [18] that identified the suitability –in both accuracy and efficiency– of quadrilateral meshes for low-order RKDG schemes over triangular meshes.

Quite few published papers dealt with the design, implementation and verification of LTS algorithms with Godunov-type shallow water solvers. Crossley and Wright [19] first probed LTS algorithms in 1D hydrodynamic modelling using uniform meshes and based on hypothetical test cases. Their findings revealed that LTS not only adds value in reducing runtimes but also in augmenting the quality of the numerical solution. Later, Sanders [20] explored a LTS method with a robust Godunov-type shallow water solver on 2D unstructured triangular meshes and considering more challenging test cases, i.e. with frictional flow over irregular topographies with wetting and drying. His conclusions reported a potential conflict between the implicit friction term discretization (IFTD) –commonly used practice to stabilize water flow simulations– and the LTS algorithm. Both of these investigations considered first-order FV Godunov-type models recommended using a maximum level of four LTSs to avoid introducing significant loss in accuracy or conservation relative to a conventional GTS formulation. More recently, second-order accurate LTS methods have been integrated with RKDG2 shallow water models following the multirate approach of Constantinescu and Sandu [21]. Seny et al. [22] explored one LTS-RKDG2 approach on unstructured triangular meshes; their approach considered flux monitoring to ensure conservation across interface cells but was concluded to be not entirely stable and did not include source terms. Their findings also point out that the multirate model is non-conservative for higher than second-order LTS-RKDG formulation. Taran and Dawson [23] modified the multirate model to produce a triangular mesh LTS-RKDG2 shallow water model that accommodates complex topography domains and wetting and drying – albeit at introducing theoretical loss of accuracy. In both of these papers, second-order mesh convergence was observed in ideal conditions (i.e. frictionless and flat topography without wetting and drying) and speed up efficiency was reported to be highly dependent on the mesh (with indications that it can accelerate efficiency up to 2X).

In this work, a different LTS-RKDG2 shallow water solver is proposed and tested with a particular focus on the applied aspects of hydraulic modelling and considering the case of uniform but structured meshes in 1D and 2D (i.e. quadrilateral). The LTS algorithm of Krivodonova [24] – particularly designed for RKDG2 schemes solving homogenous conservation laws – is newly extended to the case of the (nonhomogeneous) SWEs, i.e. with source terms and including wetting and drying [15, 17]. In Krivodonova [24], no information was provided on the gain of efficiency owed to such an LTS-RKDG2 model and flux conservation (in time) was enforced by a correction step adjusting the solution coefficients (i.e. at large interface cells). Here, the extended LTS-RKDG2 algorithm is newly reformulated so that: (i) it includes latest features relevant to applied hydraulic modelling (e.g., local slope control [25], well-balanced property [26] and depth-positivity preserving condition [27, 28]), (ii) flux conservation enforcement (in time) is dealt with by acting upon the fluxes and (iii) new measures to minimize certain knock-on effects of the IFTD are introduced. Another novel character of this paper is to systematically explore the ability of the proposed LTS-RKDG2 shallow water solver relating to applied hydraulic modelling including the issues of runtime efficiency and conservation on 1D *vs.* 2D mesh settings, convergence of accuracy-order and towards a steady state, frictional flows and shock capturing. In so doing, 1D and 2D implementations the proposed LTS-RKDG2 flow model are verified and explored according to two different non-uniform meshes comprising respectively three and four LTSs, and jointly with the conventional GTS-RKDG2 counterpart.

2. Depth-averaged Shallow Water Equations (SWEs)

From the principles of mass and momentum conservation, the mathematical model of SWEs can be cast in a 2D conservative matrix form that involve as the main flow variables the free-surface elevation (i.e. $\eta = h + z$) and the x-direction and y-direction components of unit-width discharge, which are denoted, respectively, by hu and hv. Where h is the water depth, u and v

are, respectively, the velocity components in the x-direction and y-direction, and z the bed topography.

$$\partial_t \mathbf{U} + \partial_x \mathbf{F}(\mathbf{U}) + \partial_y \mathbf{G}(\mathbf{U}) = \mathbf{S}(\mathbf{U}) \quad (1)$$

Where, (x , y) represent the Cartesian coordinates and t is the time. $\mathbf{U} = [\eta, hu, hv]^{\mathrm{T}}$ is the vector of the conserved quantities or of flow variables, $\mathbf{F} = [hu, hu^2 + 0.5g(\eta^2 - 2\eta z), huv]^{\mathrm{T}}$ and $\mathbf{G} = [hv, huv, hv^2 + 0.5g(\eta^2 - 2\eta z)]^{\mathrm{T}}$ are flux vectors relative to x- and y- directions, and $\mathbf{S}$ is a vector containing the source terms. The source term vector $\mathbf{S}$ can be further partitioned into $\mathbf{S} = \mathbf{S_b} + \mathbf{S_f}$ where $\mathbf{S_b} = [0, -g\eta\ \partial_x z, -g\eta\ \partial_y z]^{\mathrm{T}}$ and $\mathbf{S_f} = \left[0, S_{fx}, S_{fy}\right]^{\mathrm{T}}$, where $S_{fx} = -C_f u\sqrt{u^2+v^2}$ and $S_{fy} = -C_f v\sqrt{u^2+v^2}$, with $C_f = gn_M^2 / h^{1/3}$ (n_M is the Manning coefficient and g the constant gravitational acceleration).

In practical computation of flow hydrodynamics, the incorporation of the free-surface elevation variable η in the numerical discretization has proved useful to properly treat steep topographic slope (especially with the presence of a slope-limiter in the context of the RKDG2 framework [29]) and to implement a wetting and drying condition [30]. Therefore, recasting the SWEs so that [31] are the main the flow variables – whereas {h, u, v} are the secondary variables obtained from the main variables – ensures better stability and convenience to integrate a wetting and drying condition [27].

3. Non-uniform structured mesh

Firstly, a problem domain is discretized using a coarse baseline mesh consisting of $M \times N$ cells of size $\Delta x \times \Delta y$, which consists of coarsest cells, *i.e.* assigned a level of spatial refinement equal to '0'. Secondly, the baseline mesh is locally refined to enable higher level of spatial refinement varying from '1' up to a maximum of 'lev_{max}' (where lev_{max} is a positive natural number). The refinement is performed in a fractal manner, *i.e.* the cell size reduces by a factor of two whenever the refinement level increases '1'. Finally, the mesh is regularized so that it does not contain adjacent cells with sizes differing by more than a factor of two.

After these steps, a mesh embraces cells with different levels of refinement varying between '0' and 'lev_{max}', where those with level '0' are the largest and those with level 'lev_{max}' are the smallest. Thus a cell I_i with a level of refinement '$lev(i)$' ($0 \leq lev(i) \leq lev_{max}$) can be expressed as: $I_i = [x_{i-1/2};\, x_{i+1/2}] \times [y_{i-1/2};\, y_{i+1/2}]$, where $x_{i\pm1/2} = x_i \pm \Delta x_i / 2$ and $y_{i\pm1/2} = y_i \pm \Delta y_i / 2$, in which (x_i, y_i) represents the cell centre and $(\Delta x_i, \Delta y_i) = \left(\frac{\Delta x}{2^{lev(i)}}, \frac{\Delta y}{2^{lev(i)}}\right)$ is its size, which is level-dependent.

4. Review of the Global Time Stepping RKDG2 scheme (GTS-RKDG2)

Over a cell 'I_i', the GTS-RKDG2 method solves for a *local planar* solution to (1), denoted by $\mathbf{U_h} = [\eta_h, (hu)_h, (hv)_h]^T$ that is engendered by three *local* coefficients, one cell-averaged data and two 1st-order-slope data (spanning the *x*- and *y*- directions). For consistency, these coefficients are denoted by $\mathbf{U}_i^0(t)$, $\mathbf{U}_i^{1x}(t)$ and $\mathbf{U}_i^{1y}(t)$, respectively [32, 33]. Using these coefficients, the local planar solution is expanded, i.e. $\mathbf{U_h}(x, y, t)\big|_{I_i} = \left\{\mathbf{U}_i^K(t)\right\}$ ($K = 0, 1x, 1y$), where it may be written as:

$$\mathbf{U_h}(x, y, t)\big|_{I_i} = \mathbf{U}_i^0(t) + \mathbf{U}_i^{1x}(t)\left(\frac{x - x_i}{\Delta x_i / 2}\right) + \mathbf{U}_i^{1y}(t)\left(\frac{y - y_i}{\Delta y_i / 2}\right) \qquad \left(\forall (x, y) \in I_i\right) \quad (2)$$

With given initial conditions, *i.e.* $\mathbf{U_0}(x,y) = \mathbf{U}(x,y,0)$, the local expansion coefficients can be initialized (*i.e.* at $t = 0$s) as

$$\begin{aligned} \mathbf{U}_i^0(0) &= \left[\mathbf{U_0}(x_{i+1/2}, y_i) + \mathbf{U_0}(x_{i-1/2}, y_i) + \mathbf{U_0}(x_i, y_{i+1/2}) + \mathbf{U_0}(x_i, y_{i-1/2})\right] / 4 \\ \mathbf{U}_i^{1x}(0) &= \left[\mathbf{U_0}(x_{i+1/2}, y_i) - \mathbf{U_0}(x_{i-1/2}, y_i)\right] / 2 \\ \mathbf{U}_i^{1y}(0) &= \left[\mathbf{U_0}(x_i, y_{i+1/2}) - \mathbf{U_0}(x_i, y_{i-1/2})\right] / 2 \end{aligned} \qquad (3)$$

The topography function must be similarly approximated (in space), within a local planar approximation, denoted here by $z_h(x, y)\,|_{I_i}$, to balance numerically flux gradients with the topographic gradients (the *well-balanced* property) [26]. In the context of an RKDG2 scheme, the local topography-associated expansion coefficients $\{z_i^K\}$ ($K = 0, 1x, 1y$) can be found in a similar way as described in (2) and (3) [29]. With this setting, the local bed slope gradient (within $\mathbf{S_b}$) writes $\left(\partial_x z_h(x,y)\big|_{I_i}, \partial_y z_h(x,y)\big|_{I_i}\right) = \left(2z_i^{1x}/\Delta x_i, 2z_i^{1y}/\Delta y_i\right)$.

4.1 Two-stage Runge-Kutta (RK) time stepping routine

On each local cell I_i, time evolution of the expansion coefficients, $\{\mathbf{U}_i^K(t)\}$, from 't' to '$t + \Delta t_{GTS}$' is performed by two-stage RK time stepping [34]. That is, denoting $\{\mathbf{U}_i^K\}^n$ and $\{\mathbf{U}_i^K\}^{n+1}$ the discrete coefficients at 't', and '$t + \Delta t_{GTS}$' (respectively) local RK update write:

$$\left\{\mathbf{U}_i^K\right\}^{n+1/2} = \left\{\mathbf{U}_i^K\right\}^n + \Delta t_{GTS}\left\{\mathbf{L}_i^K\right\}^n \tag{4}$$

$$\left\{\mathbf{U}_i^K\right\}^{n+1} = \frac{1}{2}\left[\left\{\mathbf{U}_i^K\right\}^n + \left\{\mathbf{U}_i^K\right\}^{n+1/2} + \Delta t_{GTS}\left\{\mathbf{L}_i^K\right\}^{n+1/2}\right] \tag{5}$$

To ease technical presentation (coming next), the RK stages in (4) and (5), respectively, are hereafter referred to RK1 and RK2, which are recalled below:

- RK1 uses the coefficients $\{\mathbf{U}_i^K\}^n$ (at time 't') to produce coefficients, $\{\mathbf{U}_i^K\}^{n+1/2}$, after halfway step of time (at '$t^* = t + \Delta t_{GTS}/2$').
- RK2 further uses the coefficients of $\{\mathbf{U}_i^K\}^{n+1/2}$ to produce coefficients, $\{\mathbf{U}_i^K\}^{n+1}$, after one time step (at '$t + \Delta t_{GTS}$').

In (4) and (5), $\{\mathbf{L}_i^K\}$ are locally-conservative **DG2** (Discontinuous Galerkin 2nd-order) spatial operators (details in Subsection 4.2) that are evaluated from the expansion coefficients; whereas Δt_{GTS} denotes the Global Time Step (GTS) that is restricted by the Courant-Friedrichs-Lewy condition (CFL) stability condition with a CFL number equal to 0.3 [33]. In this work, Δt_{GTS} is evaluated according to the coefficients of the cell-averaged data as described Eq. (6) below:

$$\Delta t_{GTS} = \mathrm{CFL} \times \min_i \left(\frac{\Delta x_i}{\left|\left(u_i^0\right)^n\right| + \sqrt{g\left(h_i^0\right)^n}}, \frac{\Delta y_i}{\left|\left(v_i^0\right)^n\right| + \sqrt{g\left(h_i^0\right)^n}} \right) \quad (6)$$

Obviously, from (6), on a non-uniform mesh, Δt_{GTS} is governed by the smallest cells (*i.e.* those with the highest refinement level) and tends to decrease when more depth in refinement level is allowed ($\Delta t \to 0$ when $lev_{max} \to \infty$).

4.2 Local DG2 space operators

After application of the finite element weak formulation, to (1), and the particular adoption of Legendre basis as *local* basis functions [32, 33], a decoupled set of ODEs is obtained for the spatial update of the time-derivative of each local coefficients, namely:

$$\left\{\partial_t \mathbf{U}_i^K(t)\right\} = \left\{\mathbf{L}_i^K\right\} \qquad (K = 0, 1x, 1y) \quad (7)$$

where, $\{\mathbf{L}_i^K\}$ are nonlinear vectors of space-functions representing the flux derivatives and the source terms in (1), which can be manipulated to:

$$\mathbf{L}_i^0 = -\frac{1}{\Delta x_i}\left(\tilde{\mathbf{F}}_i^{\mathrm{E}} - \tilde{\mathbf{F}}_i^{\mathrm{W}}\right) - \frac{1}{\Delta y_i}\left(\tilde{\mathbf{G}}_i^{\mathrm{N}} - \tilde{\mathbf{G}}_i^{\mathrm{S}}\right) + \mathbf{S}\left(\mathbf{U}_i^0, z_i^{1x}, z_i^{1y}\right) \quad (8)$$

$$\begin{aligned}\mathbf{L}_i^{1x} = -\frac{3}{\Delta x_i}\Bigg\{ & \tilde{\mathbf{F}}_i^{\mathrm{E}} + \tilde{\mathbf{F}}_i^{\mathrm{W}} - \mathbf{F}\left(\mathbf{U}_i^0 + \tfrac{\hat{\mathbf{U}}_i^{1x}}{\sqrt{3}}, z_i^0 + \tfrac{z_i^{1x}}{\sqrt{3}}\right) - \mathbf{F}\left(\mathbf{U}_i^0 - \tfrac{\hat{\mathbf{U}}_i^{1x}}{\sqrt{3}}, z_i^0 - \tfrac{z_i^{1x}}{\sqrt{3}}\right) \\ & - \frac{\Delta x_i \sqrt{3}}{6}\left[\mathbf{S}\left(\mathbf{U}_i^0 + \tfrac{\hat{\mathbf{U}}_i^{1x}}{\sqrt{3}}, z_i^{1x}\right) - \mathbf{S}\left(\mathbf{U}_i^0 - \tfrac{\hat{\mathbf{U}}_i^{1x}}{\sqrt{3}}, z_i^{1x}\right)\right]\Bigg\}\end{aligned} \quad (9)$$

$$\begin{aligned}\mathbf{L}_i^{1y} = -\frac{3}{\Delta y_i}\Bigg\{ & \tilde{\mathbf{G}}_i^{\mathrm{N}} + \tilde{\mathbf{G}}_i^{\mathrm{S}} - \mathbf{G}\left(\mathbf{U}_i^0 + \tfrac{\hat{\mathbf{U}}_i^{1y}}{\sqrt{3}}, z_i^0 + \tfrac{z_i^{1y}}{\sqrt{3}}\right) - \mathbf{G}\left(\mathbf{U}_i^0 - \tfrac{\hat{\mathbf{U}}_i^{1y}}{\sqrt{3}}, z_i^0 - \tfrac{z_i^{1y}}{\sqrt{3}}\right) \\ & - \frac{\Delta y_i \sqrt{3}}{6}\left[\mathbf{S}\left(\mathbf{U}_i^0 + \tfrac{\hat{\mathbf{U}}_i^{1y}}{\sqrt{3}}, z_i^{1y}\right) - \mathbf{S}\left(\mathbf{U}_i^0 - \tfrac{\hat{\mathbf{U}}_i^{1y}}{\sqrt{3}}, z_i^{1y}\right)\right]\Bigg\}\end{aligned} \quad (10)$$

When evaluating the **DG2** operators (8)-(10), a number of essential (spatial) treatments must be considered to maintain stability and robustness for realistic flow modelling applications. These treatments are summarized here (to save space) as their details can be found in Kesserwani and Liang [17]. First, local slope coefficients (i.e. $\mathbf{U}_i^{1x}$ and $\mathbf{U}_i^{1y}$) that could cause numerical instability at sharp solution's gradient are identified and limited [25]; after slope coefficients control, they are appended with a "*hat*" (i.e. $\hat{\mathbf{U}}_i^{1x}$ and $\hat{\mathbf{U}}_i^{1y}$). Second, the discontinuous nature of the local approximate solution $\mathbf{U}^{\mathbf{h}}$, at the faces separating two adjacent cells, is incorporated via the HLLC approximate Riemann solver. The HLLC evaluations recall information from direct neighbour cells to then produce the numerical flux estimates $\tilde{\mathbf{F}}_i^{\mathrm{E}}$, $\tilde{\mathbf{F}}_i^{\mathrm{W}}$, $\tilde{\mathbf{G}}_i^{\mathrm{N}}$ and $\tilde{\mathbf{G}}_i^{\mathrm{S}}$ at, respectively, the eastern, western, northern and southern faces of each cell I_i [2]. Third, conservative spatial flux computation of these fluxes needed to ensured when cell I_i shares an edge (or more) with two finer cells (on a 2D mesh) [17]. Last, it is important to ensure the positivity of the flow variables with time evolution, which is here done based on the wetting and drying condition described in [35] (applied to revise the coefficients prior to evaluating any of the components in Eqs. (8)-(10)).

4.3 Implicit Friction Term Discretization (IFTD)

When modelling water flow over dry zone with high roughness, the water depth close to the wet/dry front can be very small and may lead to numerical instabilities if the friction source term $\mathbf{S_f}$ is explicitly discretized, within (8)-(10) [36]. Separate implicit discretization is largely recommended for handling the friction terms in order to avoid numerical instabilities. By denoting the local approximate friction term by $(\mathbf{S_f})_h$, the update due to the friction term is done by the following splitting implicit scheme:

$$\partial_t \mathbf{U}_h = (\mathbf{S_f})_h^{n+1} \tag{11}$$

Since the friction increment is zero for the continuity equation, only the momentum components are actually considered, i.e.

$$\partial_t (hu)_h = (S_{fx})_h^{n+1} \tag{12}$$

$$\partial_t (hv)_h = (S_{fy})_h^{n+1} \tag{13}$$

Eqs (12) and (13) may be respectively approximated by

$$\frac{(hu)_h^{n+1} - (hu)_h^n}{\Delta t_{GTS}} = (S_{fx})_h^n + \frac{\partial (S_{fx})_h^n}{\partial (hu)} \left[(hu)_h^{n+1} - (hu)_h^n \right] \tag{14}$$

$$\frac{(hv)_h^{n+1} - (hv)_h^n}{\Delta t_{GTS}} = (S_{fy})_h^n + \frac{\partial (S_{fy})_h^n}{\partial (hv)} \left[(hv)_h^{n+1} - (hv)_h^n \right] \tag{15}$$

From Eqs (14) and (15), the friction update formulae for the discharges components $(hu)_h$ and $(hv)_h$ may be produced

$$(hu)_h^{n+1} = (hu)_h^n + \Delta t_{GTS} \frac{(S_{fx})_h^n}{(Du)_h^n} \tag{16}$$

$$(hv)_h^{n+1} = (hv)_h^n + \Delta t_{GTS} \frac{(S_{fy})_h^n}{(Dv)_h^n} \tag{17}$$

in which Du and Dv are implicit coefficients that respectively given by

$$(Du)_h^n = 1 + \Delta t_{GTS} \left(\frac{C_f}{h} \frac{2u^2 + v^2}{\sqrt{u^2 + v^2}} \right)_h^n \tag{18}$$

$$(Dv)_h^n = 1 + \Delta t_{GTS} \left(\frac{C_f}{h} \frac{u^2 + 2v^2}{\sqrt{u^2 + v^2}} \right)_h^n \tag{19}$$

This IFTD automatically ensures $(hu)_h^{n+1} \times (hu)_h^n \geq 0$ and $(hv)_h^{n+1} \times (hv)_h^n \geq 0$, and will not predict reversed flow. In the current GTS-RKDG2 model, the splitting implicit scheme (16) and (17) are applied to each wet cell I_i to add the contribution of friction into the average coefficients $\left(hu\right)_i^0$ and $\left(hv\right)_i^0$, respectively, in a pointwise manner, prior to the RK1 stage and the RK2 stage. In order to add the friction contribution to the slope coefficients, i.e. $\left(hu\right)_i^K$ and $\left(hv\right)_i^K$ ($K \neq 0$), one simple way is to first perform a pointwise friction update at corresponding local Gaussian points and then deduce the slopes coefficients by a local *planar* P^1-projection [29, 33]. For instance, the friction increment within the slope coefficients $\left(hu\right)_i^K$, ($K \neq 0$), can be added as follows

$$\left(hu\right)_i^{1x} = \frac{\sqrt{3}}{2}\left[\left(hu\right)_{G1}^{n+1} - \left(hu\right)_{G2}^{n+1}\right] \tag{20}$$

$$\left(hu\right)_i^{1y} = \frac{\sqrt{3}}{2}\left[\left(hu\right)_{P1}^{n+1} - \left(hu\right)_{P2}^{n+1}\right] \tag{21}$$

$\left(hu\right)_{G1,G2}^{n+1}$ and $\left(hu\right)_{P1,P2}^{n+1}$ are pointwise output of the friction update (16) evaluated for $\left(hu\right)_{G1,G2}^{n} = \left[\left(hu\right)_i^0 \pm \left(hu\right)_i^{1x}/\sqrt{3}\right]^n$ and $\left(hu\right)_{P1,P2}^{n} = \left[\left(hu\right)_i^0 \pm \left(hu\right)_i^{1y}/\sqrt{3}\right]^n$, respectively. By analogy, the friction contribution can be added to $\left(hv\right)_i^K$, ($K \neq 0$).

Despite ensuring stability, the IFTD may lead to a loss in the discrete balance among fluxes and topographic source terms (i.e. *well-balanced* property [26, 28]), particularly when modelling steady flow problems over uneven topographies with non-zero velocities (refer to the detailed analysis in [37]). Furthermore, the IFTD relationships (16) and (17), which does not pose a problem with the GTS-RKDG2 scheme, may conflict with a LTS scheme (will be discussed in Subsection 5.3.1 and illustrated in Subsection 6.1).

4.4 Reduced 1D GTS-RKDG2 formulation

Neglecting the y-direction components, the vector **G** vanishes in (1) and the system reduces to two equations with two unknowns; now $\mathbf{U} = [\eta, hu]^{\mathrm{T}}$, $\mathbf{F} = [hu, hu^2 + 0.5g(\eta^2 - 2\eta z)]^{\mathrm{T}}$,

$\mathbf{S_b} = [0, -g\eta\ \partial_x z]^{\mathrm{T}}$ and $\mathbf{S}_f = [0, S_{fx}]^{\mathrm{T}}$. The 1D version of the GTS-RKDG2 scheme uses local *linear* solutions and topography approximations engendered by two coefficients (one cell-averaged and one for the monodirectional slope), i.e. $\{\mathbf{U}_i^K(t)\}$ and $\{z_i^K\}$ ($K = 0, 1x$). That is, over a 1D local cell $I_i = [x_{i-1/2};\ x_{i+1/2}]$ the flow solution (and similarly the topography apart from being static-in-time) expands as:

$$\mathbf{U_h}\left(x,t\right)\Big|_{I_i} = \mathbf{U}_i^0(t) + \mathbf{U}_i^{1x}(t)\left(\frac{x - x_i}{\Delta x_i / 2}\right) \tag{22}$$

The **DG2** spatial derivative operators reduce to two

$$\begin{aligned} \mathbf{L}_i^0 &= -\tfrac{1}{\Delta x_i}\left(\tilde{\mathbf{F}}_i^{\mathrm{E}} - \tilde{\mathbf{F}}_i^{\mathrm{W}}\right) + \mathbf{S}\left(\mathbf{U}_i^0, z_i^1\right) \\ \mathbf{L}_i^{1x} &= -\tfrac{3}{\Delta x_i}\left\{\tilde{\mathbf{F}}_i^{\mathrm{E}} + \tilde{\mathbf{F}}_i^{\mathrm{W}} - \mathbf{F}\left(\mathbf{U}_i^0 + \tfrac{\hat{\mathbf{U}}_i^{1x}}{\sqrt{3}}, z_i^0 + \tfrac{z_i^{1x}}{\sqrt{3}}\right) - \mathbf{F}\left(\mathbf{U}_i^0 - \tfrac{\hat{\mathbf{U}}_i^{1x}}{\sqrt{3}}, z_i^0 - \tfrac{z_i^{1x}}{\sqrt{3}}\right)\right. \\ &\qquad \left. - \tfrac{\Delta x_i \sqrt{3}}{6}\left[\mathbf{S}\left(\mathbf{U}_i^0 + \tfrac{\hat{\mathbf{U}}_i^{1x}}{\sqrt{3}}, z_i^1\right) - \mathbf{S}\left(\mathbf{U}_i^0 - \tfrac{\hat{\mathbf{U}}_i^{1x}}{\sqrt{3}}, z_i^1\right)\right]\right\} \end{aligned} \tag{23}$$

The RK1 and RK2 stages (4) and (5), together with the IFTD, apply straightforwardly to locally advance coefficients $\{\mathbf{U}_i^K(t)\}$ in time [35]. It is worth commenting that, relative to the 2D GTS-RKDG2 model, its 1D version is expected to be more efficient in that: first, it involves twice less inter-cell flux calculations; second, it needs twice less the number of operations to achieve the RK updates and has four times less operations in each call to the IFTD. Above all, the 1D version is not subjected to (extrinsic) inter-scales flux conservation reinforcement (in space) at heterogeneous cells [17]; thus could be also more conservative.

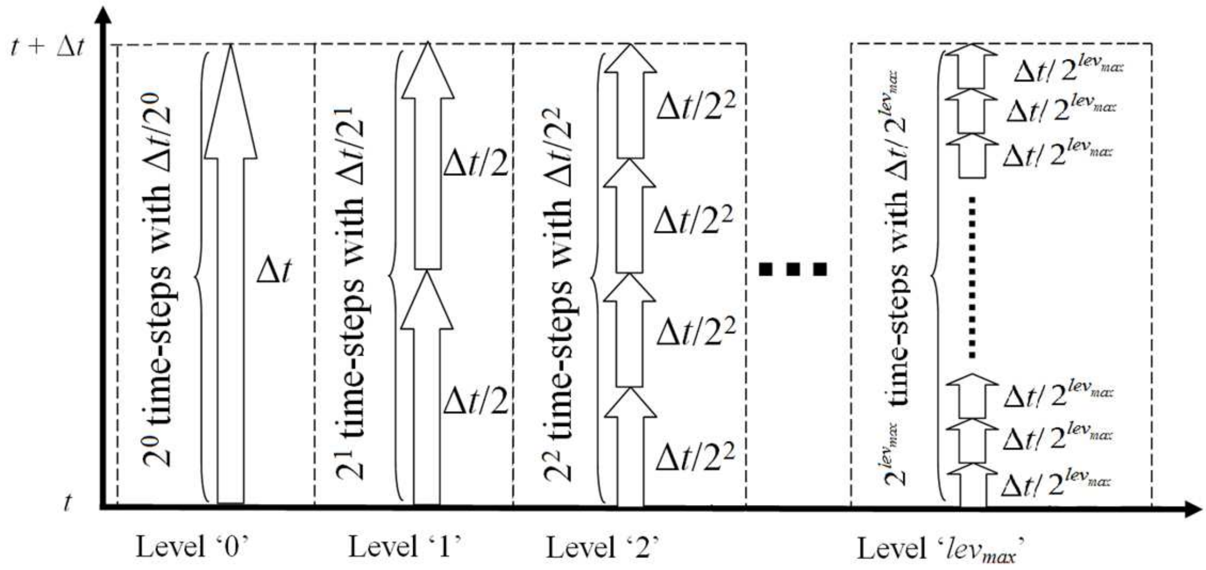


Fig. 1: LTS-RKDG2 calculation(s) to the coefficients from '*t*' to '$t + \Delta t$' on a mesh with levels of refinement '0', ..., 'lev_{max}', where a '***thick arrow***' = one LTS-RKDG2 calculation. The LTS-RKDG2 update is first achieved at cells with the level '0'. Then, the calculation moves to those cells with level '1', and so on until those cells with the highest level 'lev_{max}' are reached after $2^{lev_{max}}$ LTS-RKDG2 calculations.

5. New Local-Time-Stepping RKDG2 flow model (LTS-RKDG2)

In this section, the second-order LTS approach of Krivodonova [24] is integrated with the RKDG2 model [15] to form the so-called LTS-RKDG2 formulation. Their combination is here redesigned in order to accommodate the applied features of shallow flow simulations. For convenience of presentation, the LTS-RKGD2 method is described for the 1D version (as the description of the 2D version reads by analogy).

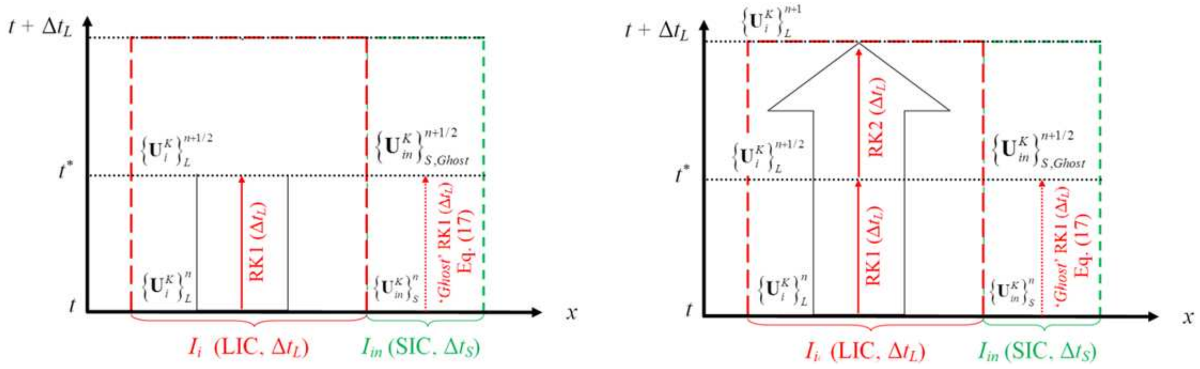
5.1 Basic concept

Assuming (for simplicity) that the maximum wave speed does not significantly influence the local CFL number, the LTS (Local Time Step) relative to cell I_i is solely dependent on its level of refinement $lev(i)$, or cell size $\Delta x_i = \Delta x/2^{lev(i)}$. Here, Δt denotes the maximum time step allowed that is yet relative to the coarsest resolution (cells with level '0' of refinement), i.e.

$$\Delta t = \Delta t_{GTS} \times 2^{lev_{\max}} \tag{24}$$

As illustrates in Fig. 1, LTS-RKDG2 calculation(s) are locally performed with the LTS Δt, $\Delta t/2$, $\Delta t/2^2$, ..., $\Delta t/2^{lev_{max}}$, orderly, on the cells with level '0', '1', '2', ..., 'lev_{max}' to progressively advance their coefficients 2^0 LTS, 2^1 LTSs, 2^2 LTSs, ..., $2^{lev_{max}}$ LTSs, respectively. At the first iteration, the LTS-RKDG2 calculation operates at cells with level '0' to directly lift their coefficients to time '$t + \Delta t$' (i.e. in one round). At the second iteration, LTS-RKDG2 calculations are undertaken at cells with level '1' (i.e. in two rounds), and so on, until the finest cells with level 'lev_{max}' are fully updated after $2^{lev_{max}}$ rounds. Therefore, cells are crossed according to their level of refinement on a mesh that comprises "*inner cells*" and "*interface cells*". When cell I_i has all of its neighbours of equal size, it will be an *inner cell*; otherwise, if at least one of its neighbours has different size, cell I_i will be an *interface cell* (so will the neighbour be). When I_i is an *inner cell*, LTS-RKDG2 calculation(s) are straightforward and actually stem from a series of GTS-RKDG2 calculation(s) using the LTS time step $\Delta t / 2^{lev(i)}$ (instead of Δt_{GTS}) across $2^{lev(i)}$ rounds.

However, when I_i is an *interface cell* at least one of its adjacent neighbours has a different refinement level. In what follows, to ease the details, we assume the eastern neighbour cell I_{in} is such a neighbour, which is also an *interface cell*. In this scenario, the LTS-RKDG2 calculation at *interface cells* $\{I_i, I_{in}\}$ faces different temporal resolutions on cells I_i and I_{in}. To accommodate this difference, synchronized '*ghost*' coefficients must be produced to complete the LTS-RKDG2 calculation(s) across first the inner RK1 and RK2 stages, and then the LTSs (as described in Section 5.2).



(a) **(b)**

Fig. 2. LTS-RKDG2 calculation at the LIC I_i (neighboured by a SIC I_{in}) to advance its coefficients from time 't' to time '$t + \Delta t_L$', where a ***'thin arrow'*** = one RK stage, ***'thick arrow'*** = one-time-step, ***'straight line'*** = '*actual*' advancement and ***'dashed line'*** = '*ghost*' advancement.

5.2 LTS-RKDG2 calculation(s) at the interface cells {I_i, I_{in}}

Since the mesh is regularized (see Section 3) and the calculation is recursive, it suffices to explain the LTS-RKDG2 calculation(s) when cells I_i and I_{in} are one refinement level different. Without loss of generality, assume cells I_i and I_{in} have, respectively, '0' and '1' as a refinement levels. Cells I_i and I_{in} can, respectively, be viewed as "*Large Interface Cell*" (LIC) and "*Small Interface Cell*" (SIC); consistently, their associated LTS, coefficients and fluxes will be appended with the subscripts 'L' and S', respectively. Firstly, *one* LTS-RKDG2 calculation is applied to update the '*actual*' coefficients at the LIC (I_i) while employing '*ghost*' synchronized coefficients from the SIC (I_{in}) [Subsection 5.2.1]. Next, *two* LTS-RKDG2 calculations are applied to update the '*actual*' coefficients are at the SIC (I_{in}) while using '*ghost*' coefficients from the LIC (I_i) [Subsection 5.2.2].

5.2.1 Coefficients update at the LIC (I_i)

At the LIC I_i, LTS-RKDG2 calculation starts from the coefficients at time 't', i.e. $\{\mathbf{U}_i^K\}_L^n$, with the LTS $\Delta t_L = \Delta t/2^0$. At '$t$', the coefficients at the SIC I_{in}, i.e. $\{\mathbf{U}_{in}^K\}_S^n$, are also available.

DG2 space operators on I_i, i.e. $\{\mathbf{L}_i^K\}_L^n$, can be obtained leading to (after RK1) the '*actual*' coefficients on I_i at '$t^* = t + \Delta t_L/2$', i.e. $\{\mathbf{U}_i^K\}_L^{n+1/2}$, (Fig. 2a — 'straight thin arrow' in the left-hand-side). Equally, RK1 is applied on on I_{in} but with the LTS Δt_L leading to time-matching '*ghost*' coefficients, i.e. $\{\mathbf{U}_{in}^K\}_{S,Ghost}^{n+1/2}$ (Fig. 2a — 'dashed thin arrow' in the right-hand-side), namely:

$$\left\{\mathbf{U}_{in}^K\right\}_{S,Ghost}^{n+1/2} = \left\{\mathbf{U}_{in}^K\right\}_S^n + \Delta t_L \left\{\mathbf{L}_{in}^K\right\}_S^n \tag{25}$$

Again, **DG2** space operators on I_i, i.e. $\{\mathbf{L}_i^K\}_L^{n+1/2}$, can be now obtained for evaluation in RK2 advancing thereby to produce the 'actual' coefficients to time '$t + \Delta t_L$', i.e. $\{\mathbf{U}_i^K\}_L^{n+1}$ (Fig. 2b — second 'straight thin arrow' and the 'thick arrow' in the left-hand-side).

5.2.2 Coefficients update at the SIC (I_{in})

Calculation restarts (time 't') at the SIC I_{in} with the LTS $\Delta t_S = \Delta t_L/2$; thus two LTS-RKDG2 calculations are needed to move its '*actual*' coefficients to '$t + \Delta t_L$' (i.e. across two rounds). Before detailing these calculations, it should be noted that any past 'ghost' information on I_{in} must be ignored; whereas some past 'actual' information on I_i are needed (i.e. the DG2 space operator records across inner time stages) to define the following quadratic function:

$$\left\{\phi_i^K(\tau)\right\} = \left\{\mathbf{U}_i^K\right\}_L^n + \left\{\mathbf{L}_i^K\right\}_L^n (\tau - t) + \frac{\left\{\mathbf{L}_i^K\right\}_L^{n+1/2} - \left\{\mathbf{L}_i^K\right\}_L^n}{2\Delta t_L} (\tau - t)^2 \tag{26}$$

that is needed to interpolate '*ghost*' coefficients on I_i at a fractional time-step $\tau \in [t; t + \Delta t_L[$ and an associated intermediate time-stage at $\tau^* \in [\tau; t + \Delta t_L[$, i.e.

$$\left\{\mathbf{U}_i^K(\tau)\right\}_{L,Ghost}^n = \left\{\phi_i^K(\tau)\right\} \tag{27}$$

$$\left\{\mathbf{U}_i^K(\tau^*)\right\}_{L,Ghost}^{n+1/2} = \left\{\mathbf{U}_i^K(\tau)\right\}_{L,Ghost}^n + \Delta t_S \frac{d}{d\tau}\left\{\phi_i^K(\tau)\right\} \tag{28}$$

- In the first LTS-RKDG2 calculation, coefficients over I_{in} are advanced one LTS to '$t_2 = t + \Delta t_S$'. Calculation starts from the coefficients available at 't', i.e. $\{\mathbf{U}_{in}^{K}\}_{S}^{n}$ and $\{\mathbf{U}_{i}^{K}\}_{L}^{n}$, that give the **DG2** operators on I_{in}, i.e. $\{\mathbf{L}_{in}^{K}\}_{S}^{n}$, which in turn (via RK1) yield the '*actual*' coefficients at '$t_1^* = t + \Delta t_S/2$', i.e. $\{\mathbf{U}_{in}^{K}\}_{S}^{n+1/2}$ (Fig. 3a — 'straight thin arrow' at the right-hand-side). Meanwhile, on I_i, synchronized '*ghost*' coefficients, i.e. $\{\mathbf{U}_{i}^{K}\}_{L,Ghost}^{n+1/2}$, are reconstructed (Fig. 3a — 'dashed thin arrow' at the left-hand-side) by [(27) and (28) evaluated at $\tau = t_1^*$]:

$$\left\{\mathbf{U}_i^K\right\}_{L,Ghost}^{n+1/2} = \left\{\mathbf{U}_i^K\right\}_L^n + \Delta t_S \left\{\mathbf{L}_i^K\right\}_L^n \tag{29}$$

Local **DG2** space operators $\{\mathbf{L}_{in}^{K}\}_{S}^{n+1/2}$ on I_{in} can be now evaluated to (via RK2) yield the '*actual*' coefficients at 't_2', i.e. $\{\mathbf{U}_{in}^{K}\}_{S}^{n+1}$ (Fig. 3c — second 'straight thin arrow' and the 'thick arrow' at the right-hand-side). Meanwhile, again, synchronized (at 't_2') '*ghost*' coefficients, on I_i, i.e. $\{\mathbf{U}_{i}^{K}\}_{L,Ghost}^{n+1}$, are reconstructed (Fig. 3b — second 'dashed thin arrow' and the overall 'thick dashed arrow' at the left-hand-side) [via (26) and (27) evaluated at $\tau = t_2$] by:

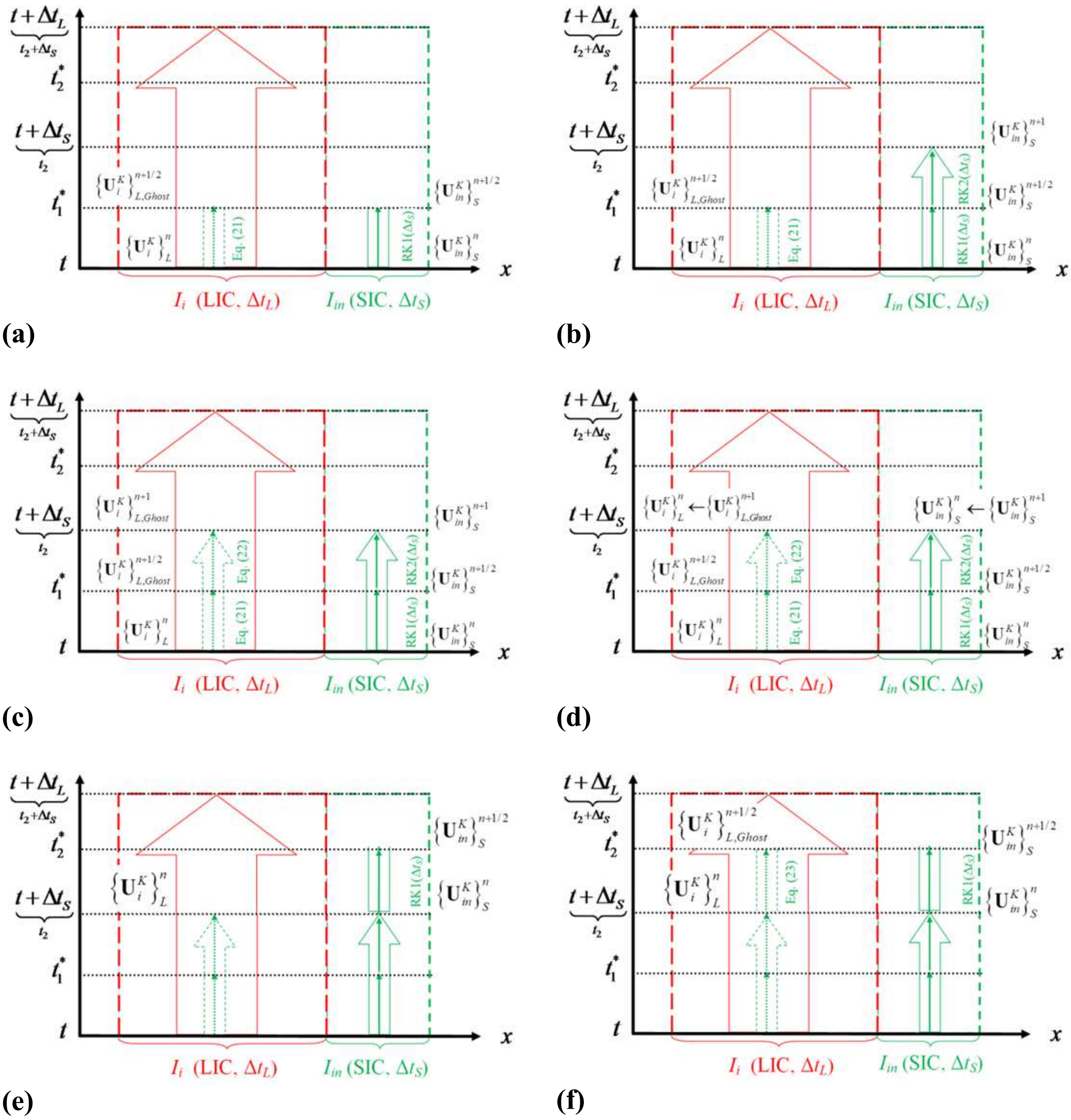
$$\left\{\mathbf{U}_i^K\right\}_{L,Ghost}^{n+1} = \left\{\phi_i^K(t_2)\right\} = \left\{\mathbf{U}_i^K\right\}_L^n + \Delta t_S \left\{\mathbf{L}_i^K\right\}_L^n + \left(\Delta t_S\right)^2 \frac{\left\{\mathbf{L}_i^K\right\}_L^{n+1/2} - \left\{\mathbf{L}_i^K\right\}_L^n}{2\Delta t_L} \tag{30}$$

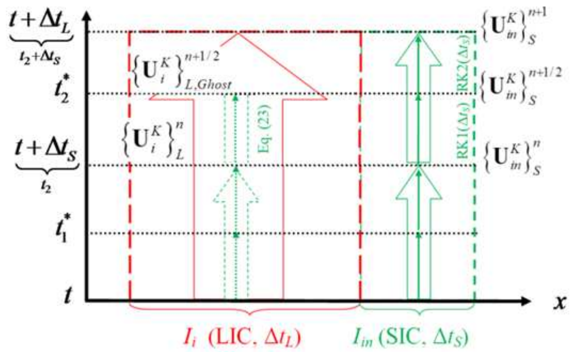
- Prior to the second LTS-RKDG2 calculation, both '*actual*' and '*ghost*' coefficients (at I_{in} and I_i) are reinitialized at 't_2' (see Fig. 3d): $\{\mathbf{U}_{in}^{K}\}_{S}^{n} \leftarrow \{\mathbf{U}_{in}^{K}\}_{S}^{n+1}$ & $\{\mathbf{U}_{i}^{K}\}_{L}^{n} \leftarrow \{\mathbf{U}_{i}^{K}\}_{L,Ghost}^{n+1}$ (all variable relevant to intermediate time-stage $\{\cdot\}^{n+1/2}$ can be now reused). Calculation starts from the initial coefficients at 't_2', i.e. $\{\mathbf{U}_{in}^{K}\}_{S}^{n}$ and $\{\mathbf{U}_{i}^{K}\}_{L}^{n}$, leading to (after calculation of $\{\mathbf{L}_{in}^{K}\}_{S}^{n}$ on I_{in} and then via and RK1) the '*actual*' coefficients at '$t_2^* = t_2 + \Delta t_S/2$', i.e. $\{\mathbf{U}_{in}^{K}\}_{S}^{n+1/2}$ (Fig. 3e—right part along the third 'straight thin arrow').

Meanwhile, once again, on I_i, synchronized (at t_2^*) '*ghost*' coefficients are reconstructed [using (26)-(28) evaluated at $\tau = t_2^*$], i.e. $\{\mathbf{U}_i^K\}_{L,Ghost}^{n+1/2}$, by (Fig. 3f—left part along the third 'dashed thin arrow'):

$$\left\{\mathbf{U}_i^K\right\}_{L,Ghost}^{n+1/2} = \left\{\mathbf{U}_i^K\right\}_L^n + \Delta t_S \left[\left\{\mathbf{L}_i^K\right\}_L^n + \frac{\left\{\mathbf{L}_i^K\right\}_L^{n+1/2} - \left\{\mathbf{L}_i^K\right\}_L^n}{\Delta t_L} \left(\Delta t_S\right) \right] \quad (31)$$

Finally, **DG2** operators, on I_{in}, i.e. $\{\mathbf{L}_{in}^K\}_S^{n+1/2}$, can be found and evaluated in RK2 to yield the '*actual*' coefficients at time '$t + \Delta t_L$', i.e. $\{\mathbf{U}_{in}^K\}_S^{n+1}$.





(g)

Fig. 3. LTS-RKDG2 calculation at the SIC I_i (neighboured by the LIC I_{in}) to advance its coefficients from time 't' to time '$t + \Delta t_L$' in two consecutive rounds. A ***'thin arrow'*** = one-time-stage, ***'thick arrow'*** = one-time-step, ***'straight line'*** = '*actual*' advancement and ***'dashed line'*** = '*ghost*' advancement.

5.3 Specific issues relevant to applied hydraulic modelling

During the LTS-RKDG2 calculation(s), slope-limiting and wetting and drying do not appear to pose any specific technical problems. In contrast, more computational work is found necessary to properly handle the IFTD (Subsection 5.3.1) and conserve the fluxes in time (Subsection 5.3.2) at *interface cells*.

5.3.1 Hybrid explicit-implicit discretization of the friction term

When using the implicit friction source term discretization (IFTD) [see Subsection 4.3] across the LTS-RKDG2 calculations, its aforementioned side effect of disturbing the *well-balanced* property may magnify at *inner cells* proportional to an increase in the refinement level (see also numerical experiments in Subsection 6.1). On the other hand, the different LTSs within the IFTD complicate its integration during the LTS-RKDG2 calculations at *interface cells* (i.e. to avoid duplicate use of the IFTD at the same *interface cell* with two different LTSs). This complication stems from the need to produce extra phases of '*ghost*' friction advancement, and removal, in line with the '*ghost*' coefficients advancement (outlined before in Subsections 5.2.1 and Subsection 5.2.2).

One convenient way to avoid this complication is to restrict the usability of the IFTD to those cells where the water height may potentially become infinitesimal; whereas elsewhere (at wet cells) use explicit friction source term discretization in the **DG2** operators (23) [free from any time-step dependence]. In this work, the IFTD is only applied locally at a cell I_i when a small water level occurs in the calculation stencil containing cell I_i and its direct neighbours, e.g. in the 1D when:

$$\min\left(h_{i-1}^0 h_i^0, h_{i+1}^0\right) \leq 3\% \times h^{\max}(t) \tag{32}$$

where $h^{\max}(t)$ represents the maximum water level spanning the wet domain at time 't'. The 3% is a user-selected threshold, which means that the IFTD will be active at, or around, those cell where the RKDG2 calculation involves, at least, a depth that is smaller than 3% of the maximum depth.

Now the IFTD implementation with LTS-RKDG2 calculation(s) is described, which could occur at either *inner cells* or *interface cells*. At *inner cells* the IFTD applies (recursively) a similar way as with the GTS-RKDG2 scheme. In contrast, at *interface cells* the IFTD needs a careful treatment across RK1 and RK2 stages where 'Ghost' data change for the different LTSs (Subsections 5.2.1 and 5.2.2). Here, we detail the application of the IFTD within the LTS-RKDG2 calculation(s) consistent with *interface cell* $\{I_i, I_{in}\}$.

- During the LTS-RKDG2 calculation at the LIC, the IFTD step (16) applies at I_i (resp. at I_{in}) to amend the '*actual*' (resp. '*ghost*') discharge coefficients within $\{\mathbf{U}_i^K\}_L^n$ and $\{\mathbf{U}_{in}^K\}_S^n$. Then, once coefficients $\{\mathbf{U}_i^K\}_L^{n+1/2}$ and $\{\mathbf{U}_{in}^K\}_{S,Ghost}^{n+1/2}$ are in place (Subsection 5.2.1), the IFTD step (16) is again applied at I_i (resp. at I_{in}) to amend their '*actual*' (resp. '*ghost*') discharge coefficients. However, once 'actual' coefficients at I_i are lifted to '$t + \Delta t_L$', it is necessary to restore their initial (*frictionless* discharge) relative to time 't'.

- During the LTS-calculations at the SIC I_{in}, no further treatment is here needed. In effect, after the LTS-RKDG2 calculation at the LIS I_i: (a) its initial discharge coefficients in $\{\mathbf{U}_i^K\}_L^n$ have been reset to frictionless; (b) the (saved) **DG2** operators $\{\mathbf{L}_i^K\}_L^n$ and $\{\mathbf{L}_i^K\}_L^{n+1/2}$ already include the '*actual*' effects due to friction. Thus, '*ghost*' coefficients at I_i, reconstructed by (29)-(31), are expected to include the contribution of friction.

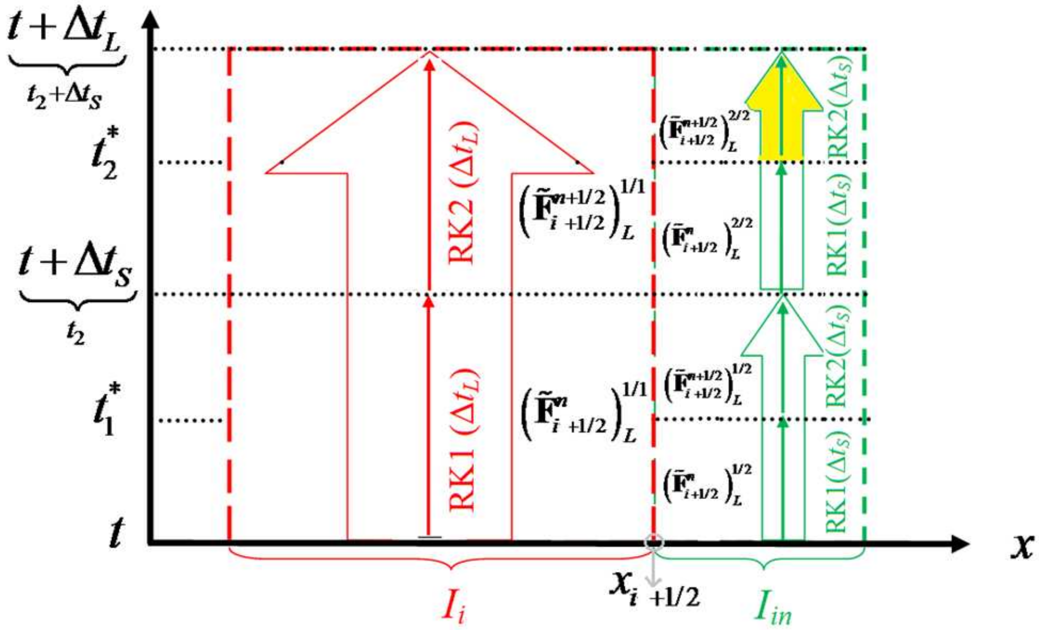


Fig. 4: History of the 'actual' inner RK stages of the LTS-RKDG2 calculations at the LIC I_i and the SIC I_{in} in terms of Riemann flux evaluations. Particular case (when $\Delta t_L = \Delta t$) where flux conservation reinforcement is needed and take action at the SIC within the RK2 stage of the last of LTS-RKDG2 calculation, using (25).

5.3.2 Flux conservation at interface cells

After achieving the LTS-RKDG2 calculations at the LIC I_i (Subsection 5.2.1) and the SIC I_{in} (Subsection 5.2.2), the sum of Riemann flux quantities cumulated between times 't' and '$t + \Delta t_L$' at the edge $x_{i+1/2}$ may not be equal. For instance, following the notations in Fig. 4, it may happen that

$$\left[\left(\tilde{\mathbf{F}}_{i+1/2}^{n}\right)_L^{1/1} + \left(\tilde{\mathbf{F}}_{i+1/2}^{n+1/2}\right)_L^{1/1}\right]_t^{t+\Delta t_L} \neq \left[\left(\tilde{\mathbf{F}}_{i+1/2}^{n}\right)_S^{1/2} + \left(\tilde{\mathbf{F}}_{i+1/2}^{n+1/2}\right)_S^{1/2}\right]_t^{t_2} + \left[\left(\tilde{\mathbf{F}}_{i+1/2}^{n}\right)_S^{2/2} + \left(\tilde{\mathbf{F}}_{i+1/2}^{n+1/2}\right)_S^{2/2}\right]_{t_2}^{t+\Delta t_L} \quad (33)$$

where $(\tilde{\mathbf{F}}_{i+1/2}^{n})_L^{1/1} + (\tilde{\mathbf{F}}_{i+1/2}^{n+1/2})_L^{1/1}$ is the sum of Riemann fluxes accumulated from the sole LTS-RKDG2 calculation at the LIC I_i (superscript '1/1'); whereas, $(\tilde{\mathbf{F}}_{i+1/2}^{n})_S^{1/2} + (\tilde{\mathbf{F}}_{i+1/2}^{n+1/2})_S^{1/2}$ and $(\tilde{\mathbf{F}}_{i+1/2}^{n})_S^{2/2} + (\tilde{\mathbf{F}}_{i+1/2}^{n+1/2})_S^{2/2}$ are the sum of Riemann fluxes accumulated during the first (superscript '1/2') and the second (superscript '2/2') LTS-RKDG2 calculations at the SIC I_{in}.

To alleviate this effect, flux conservation (in time) is reinforced at the SIC I_{in} and during the ***final*** of LTS-RKDG2 **calculation** and, more particularly, at the **RK2** stage (when the coefficients are pending one last step before reaching '$t + \Delta t$') [Fig. 4—right highlighted portion of the thick arrow). This can be done by exceptionally choosing the flux $(\tilde{\mathbf{F}}_{i+1/2}^{n+1/2})_S^{2/2}$ so as to ensure that the two sides of Eq. (33) remain equal, i.e.

$$\left[\left(\tilde{\mathbf{F}}_{i+1/2}^{n+1/2}\right)_S^{2/2}\right]_{t_2^*}^{t+\Delta t} = \left[\left(\tilde{\mathbf{F}}_{i+1/2}^{n}\right)_L^{1/1}+\left(\tilde{\mathbf{F}}_{i+1/2}^{n+1/2}\right)_L^{1/1}\right]_t^{t+\Delta t} - \left[\left(\tilde{\mathbf{F}}_{i+1/2}^{n}\right)_S^{1/2}+\left(\tilde{\mathbf{F}}_{i+1/2}^{n+1/2}\right)_S^{1/2}\right]_t^{t_2} - \left[\left(\tilde{\mathbf{F}}_{i+1/2}^{n}\right)_S^{2/2}\right]_{t_2}^{t_2^*} \qquad (34)$$

and then proceed with the conventional evaluation for the **DG2** space operators to complete the RK2 stage.

5.4 LTS-RKDG2 algorithm on a mesh with multiple refinement levels

5.4.1 Computational and memory demands

In the GTS-RKDG2 calculation, coefficients are moved from 't' to '$t + \Delta t$' in one round. Computational storage associated with this calculation (at cell I_i and for all K coefficients) are three matrices $\{\mathbf{U}_i^K\}^n$, $\{\mathbf{U}_i^K\}^{n+1/2}$ and $\{\mathbf{U}_i^K\}^{n+1}$ for storing coefficients at times 't', 't^*' and '$t + \Delta t_{GTS}$'; whereas any other variables/operations are local and/or momentary.

Calculations of the LTS-RKDG2 are recursive and occur across 2^k rounds for cells with level 'k' of refinement ($1 \le k \le lev_{max}$). Nevertheless, the same allocated matrices can be used subject to re-initialization at the beginning of each round, *i.e.* $\{\mathbf{U}_i^K\}^n \leftarrow \{\mathbf{U}_i^K\}^{n+1}$. Nonetheless, extra *local* storage is required to facilitate the calculations at *interface cells*, namely for recording the **DG2** operators at LICs, evolving sums of Riemann fluxes at *interface cells* and restoring frictionless discharge coefficients *interface cells*. Moreover, these storage demands become higher for the 2D version given the presence of an additional slope component and **DG2** operator, and two more direct neighbours.

5.4.2 LTS-RKDG2 calculations at interface cells {I_i, I_{in}}

Here, all the steps of LTS-RKDG2 calculations at {I_i, I_{in}} are combined including the specific features relevant to hydrodynamic modelling. At time 't', coefficients over I_i and I_{in} are available and Table 1 summarises the steps of the LTS-RKDG2 calculations for lifting coefficients of cells I_i and I_{in} to time '$t + \Delta t_L$' (in which subscripts 'L' and 'S' are overlooked for the coefficients and the DG2 operators).

Table 1: List of steps for the LTS-RKDG2 calculations at I_i (resp. I_{in}) with the LTS Δt_L (resp. $\Delta t_S = \Delta t_L/2$) to move its coefficients from time 't' to time '$t + \Delta t_L$' in one round (resp. in two rounds).

1. Start with the one round over the LIC I_i with the time step Δt_L.
A. Detect if an IFTD is needed. If so, save the initial frictionless discharge coefficients at I_i and I_{in}; using (16) with Δt_L, do an '*actual*' (reps. a '*ghost*') IFTD step at I_i (resp. I_{in}) to add friction effects to the discharge coefficients in $\{\mathbf{U}_i^K\}^n$ (resp. $\{\mathbf{U}_{in}^K\}^n$). Otherwise, omit Step 1-A.
B. Evaluate and save the Riemann flux at $x_{i+1/2}$. Then, evaluate, via (23), and save the **DG2** space operators $\{\mathbf{L}_i^K\}^n$.
C. Advance the coefficients at I_i one time stage, using (4) with the time step Δt_L, to produce $\{\mathbf{U}_i^K\}^{n+1/2}$ (*i.e.*, '*actual*' coefficients).
D. In a similar way, *i.e.* via (25), advance the coefficients over I_{in} one time stage, to produce '*ghost*' coefficients $\{\mathbf{U}_{in}^K\}_{S,Ghost}^{n+1/2}$. Set $\{\mathbf{U}_{in}^K\}^{n+1/2} \leftarrow \{\mathbf{U}_{in}^K\}_{S,Ghost}^{n+1/2}$.
E. If an IFTD is needed. Using (16) with Δt_L, do an '*actual*' (reps. a '*ghost*') IFTD step at I_i (resp. I_{in}) to add increment of friction in the discharge coefficients of $\{\mathbf{U}_i^K\}^{n+1/2}$ (resp. $\{\mathbf{U}_{in}^K\}^{n+1/2}$). Otherwise, omit Step 1-E.
F. Evaluate and save the Riemann flux at $x_{i+1/2}$. Then, evaluate, via (23), and save the **DG2** space operators $\{\mathbf{L}_i^K\}^{n+1/2}$.
G. Advance the coefficients over I_i another time stage, using (5) with the time step Δt_L, to produce $\{\mathbf{U}_i^K\}^{n+1}$.
H. Restore the (original) frictionless state for the coefficients $\{\mathbf{U}_i^K\}^n$ and $\{\mathbf{U}_{in}^K\}^n$ using the saved frictionless discharge coefficients in Step 1-A.
2. Then, two rounds over the SIC I_{in} with the time step $\Delta t_S = \Delta t_L/2$.

A. Detect if an IFTD is needed. If so, using (16) with Δt_S, do an '*actual*' IFTD step at I_{in} to add increment due to the friction effects to the discharge coefficients in $\{\mathbf{U}_{in}^K\}^n$. Otherwise, omit Step 2-A. B. Evaluate and save the Riemann flux at $x_{i+1/2}$. Then, evaluate, via (23) the **DG2** space operators $\{\mathbf{L}_{in}^K\}^n$. C. Advance the coefficients over I_{in} one time stage, using (4) with the time step Δt_S, to produce the '*actual*' coefficients $\{\mathbf{U}_{in}^K\}^{n+1/2}$; if an IFTD is needed, using (16) with Δt_S, do another '*actual*' IFTD step for $\{\mathbf{U}_{in}^K\}^{n+1/2}$. D. Produce '*ghost*' coefficients $\{\mathbf{U}_i^K\}_{L,Ghost}^{n+1/2}$ over I_i [*i.e.*, using (29) with $\{\mathbf{U}_i^K\}^n$ from Step 1-H and the previously saved $\{\mathbf{L}_i^K\}^n$ from Step 1-B]. Set $\{\mathbf{U}_i^K\}^{n+1/2} \leftarrow \{\mathbf{U}_i^K\}_{L,Ghost}^{n+1/2}$. E. Evaluate and save the Riemann flux at $x_{i+1/2}$. Then, evaluate, via (23), the **DG2** space operators $\{\mathbf{L}_{in}^K\}^{n+1/2}$. F. Advance the coefficients over I_{in} another time stage, using (5) with the time step Δt_S, to produce $\{\mathbf{U}_i^K\}^{n+1}$. G. Produce time-matching '*ghost*' coefficients $\{\mathbf{U}_i^K\}_{L,Ghost}^{n+1}$ over I_i (*i.e.*, using (30) with the same parameters used in (29) and by further involving $\{\mathbf{L}_i^K\}^{n+1/2}$ saved in Step 1-F). H. Re-initialize the coefficients at I_i and I_{in}: $\{\mathbf{U}_{in}^K\}^n \leftarrow \{\mathbf{U}_{in}^K\}^{n+1}$ and $\{\mathbf{U}_i^K\}^n \leftarrow \{\mathbf{U}_i^K\}_{L,Ghost}^{n+1}$. I. Do similar as Steps 2-A, 2-B and 2-C to reproduce the '*actual*' coefficients $\{\mathbf{U}_{in}^K\}^{n+1/2}$. J. Produce, via (31), '*ghost*' coefficients $\{\mathbf{U}_i^K\}_{L,Ghost}^{n+1/2}$ and reset $\{\mathbf{U}_i^K\}^{n+1/2} \leftarrow \{\mathbf{U}_i^K\}_{L,Ghost}^{n+1/2}$. K. Do similar as Step 2-E and Step 2-F to finally obtain the '*actual*' coefficients $\{\mathbf{U}_{in}^K\}^{n+1}$.
Remark (exceptional flux conservation Step 2-L) L. In the case where Step 2-I – Step 2-K take action at the very last round, which is lifting the coefficients over I_{in} to 't+Δt', Step 2-J should be removed and the flux in Step 2-K is directly estimated by the relationship (34).

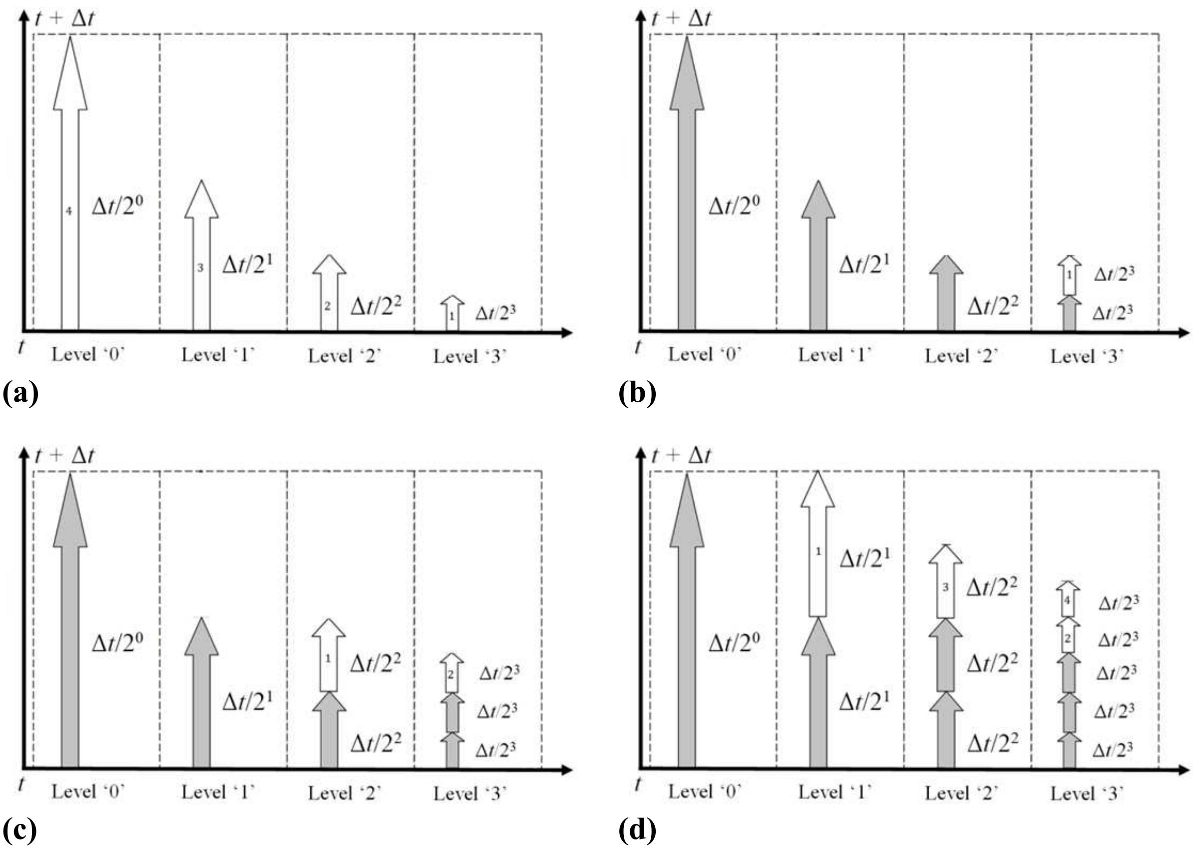
5.4.3 Generalized LTS-RKDG2 model

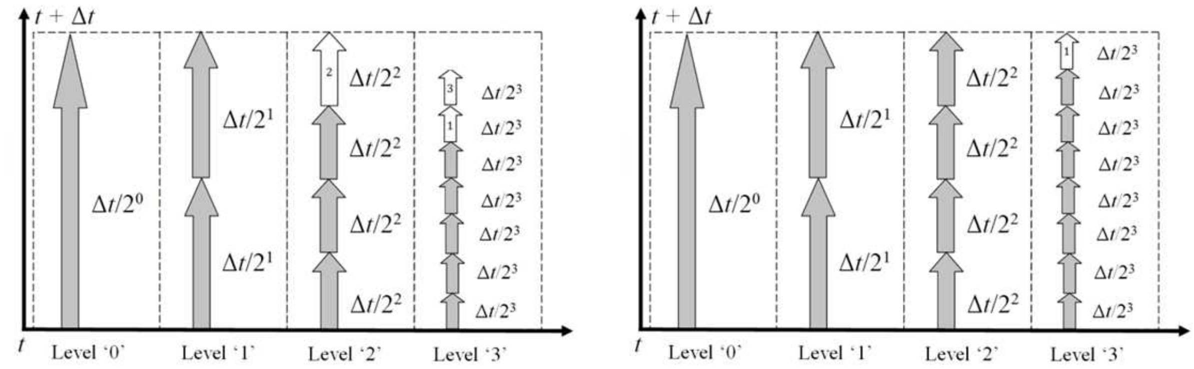
Following Krivodonova [24], the generalization of the LTS-RKDG2 scheme on a mesh with arbitrary depth of refinement stems from a recursive repetition of the steps in Table 1, so that to keep a "*staircase*" in time after each iteration. For simplicity, it is described for lev_{max} = 3 in Table 2 and correspondingly in Fig. 5. Here, a total of four iterations is needed to lift the coefficients over all cells from time 't' to time '$t + \Delta t$'. Evidently, after round #k (k = 1, 2, 3 and 4), the coefficients over cells with level k reaches '$t + \Delta t$'.

Table 2: List of steps for LTS-RKDG2 calculations at a mesh with four refinement levels of '0', '1', '2' and '3' using respectively the LTS Δt, $\Delta t/2$, $\Delta t/2^2$ and $\Delta t/2^3$.

<u>**Round #1:**</u> advance the coefficients one LTS over all cells using Steps (1-A)—(1-H) or Steps (2-A)—(2-F). As seen in Fig. 5a, the calculation starts orderly with the cells of level '3', '2',

‘1’ and then ‘0’ (*i.e.*, using respectively the LTS $\Delta t/2^3$, $\Delta t/2^2$, $\Delta t/2$ and Δt).
Round #2: first, advance the coefficients over cells with level ‘3’ one LTS using Steps (2-G)—(2-K); (*i.e.*, Fig. 5b). Second, advance the coefficients over cells with level ‘2’ one LTS using Steps (1-A)—(1-H); (*i.e.*, Fig. 5c) and revisit the cells with level ‘3’ to further advance their coefficients another LTS using Steps (2-G)—(2-K); (*i.e.*, Fig. 5c). Fourth, advance the coefficients over cells with level ‘1’ one LTS using Steps (2-G)—(2-K) while *enforcing flux conservation* via (34); (*i.e.*, Fig. 5d). Fifth, revisit the cells with level ‘3’ and further advance their coefficients one more LTS using Steps (2-G)—(2-K); (*i.e.*, Fig. 5d). Sixth, revisit the cells with level ‘2’ and further advance their coefficients one more LTS Steps (2-G)—(2-K); (*i.e.*, Fig. 5d). Finally, revisit the cells with level ‘3’ and again advance their coefficients one more LTS using Steps (2-A)—(2-F); (*i.e.*, Fig. 5d).
Round #3: first, advance the coefficients over cells with level ‘3’ one LTS using Steps (2-G)—(2-K); (*i.e.*, Fig. 5e). Second, advance the coefficients over cells with level ‘2’ one LTS using Steps (2-G)—(2-K) while *reinforcing flux conservation* via (34). Finally, revisit the cells with level ‘3’ and again advance their coefficients one more LTS using Steps (2-A)—(-F); (*i.e.*, Fig. 5e).
Round #4: now, the remaining step is to advance the coefficients over cells of level ‘3’ one LTS using Steps (2-G)—(2-K) while *enforcing flux conservation* via (34); (*i.e.*, Fig. 5f).



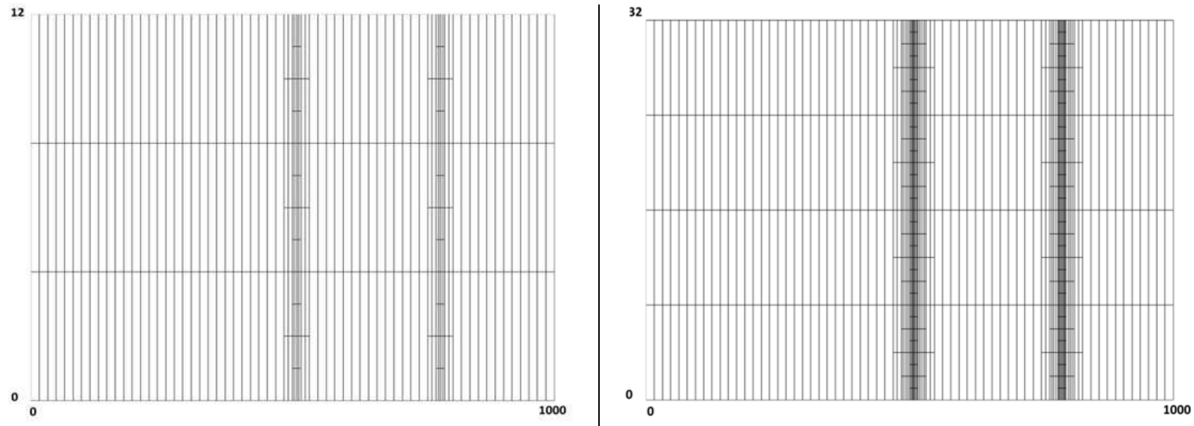


(e) **(f)**

Fig. 5: Schematic description of the LTS-RKDG2 calculations over the *interface cells* relative to mesh with four levels of refinement; "***Gray arrow***" = previous step(s) and "***Blank numbered arrow***" = present step(s) in successive order.

6. LTS-RKDG2 model's verification relative to the GTS-RKDG2 model

The 1D and 2D formulations of the LTS-RKDG2 scheme are verified for two non-uniform mesh configurations, refereed hereafter to as '*mesh-3LTSs*' and '*mesh-4LTSs*', which respectively involve '3' and '4' levels of local spatial-temporal discretization-scales (*i.e.*, $lev_{\max} = 2$ and $lev_{\max} = 3$, respectively). On the former mesh the LTS-RKDG2 framework coordinates the LTSs {Δt, $\Delta t/2$, $\Delta t/4$} while it coordinates the LTSs {Δt, $\Delta t/2$, $\Delta t/4$, $\Delta t/8$} on the latter mesh. Selected benchmark tests are employed to investigate the performance of the LTS-RKDG2 scheme (*i.e.,* 1D and/or 2D versions on both '*mesh-3LTSs*' and '*mesh-4LTSs*') with respect to the traditional GTS-RKDG2 scheme, while discussing/identifying several issues pertaining to computational hydraulics and quantifying the runtime saving (*i.e.,* the ratio 'runtime GTS'/'runtime LTS'). By default, transmissive (numerical) boundary conditions are used in the both RKDG2 models unless otherwise mentioned for specific test cases.

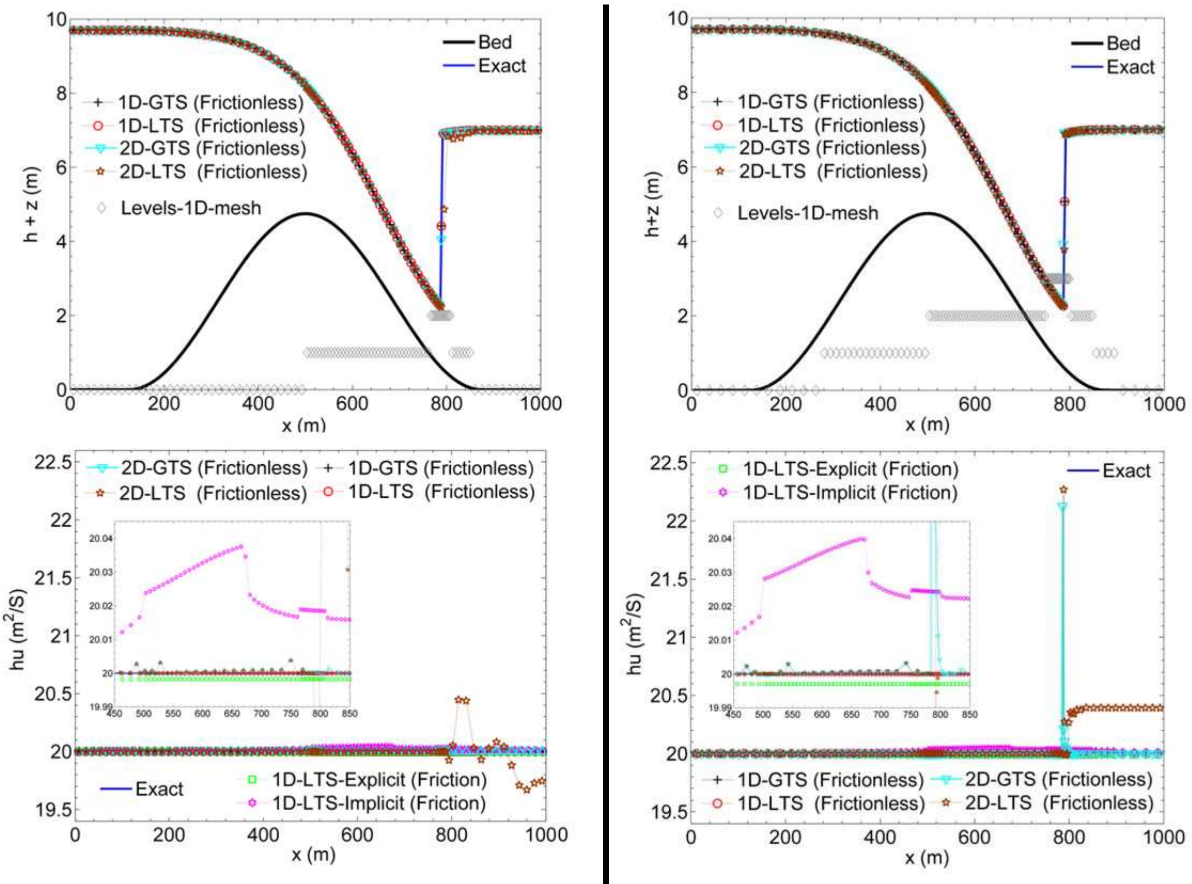


(a) **(b)**

Fig. 6: Transcritical flow over a hump with shock. 2D domains and meshes with local refinement around the point of transcritical flow and the local of the water jump; **(a)** lev_{max} = 2 and **(b)** lev_{max} = 3.

6.1 Steady transcritical flow over topography with shock

This test investigates moving steady transcritical flow over non-flat topography with a shock. It is usually employed to demonstrate the capability of a numerical method to converge towards a steady state, accurately balance the flux gradient with the topography gradient, and capture transcritical flow transitions and water jumps. The channel is 1000m long with a hump-shape topography located between x = 125m and x = 875m [38]. Inflow (physical) boundary condition is imposed through a unit discharge of 20m^2/s and the (physical) outflow boundary is a water level of 7m. Under these conditions, a steady transitional flow takes place where the flow changes from subcritical to supercritical at x = 500m. Downstream of the topography, a hydraulic jump occurs as the flow regime restores to subcritical. A simulation starts from an initial water height of 9.7m and is desired to stop after a relatively long time evolution (*i.e.,* t = 2000s). Simulations are done using the 1D and 2D versions of the GTS-RKDG2 and LTS-RKDG2 schemes. The 1D and 2D mesh characteristics are listed in Table 3; the 2D domains and associated mesh-refinement are described in Fig. 6, while the level of refinement used for the 1D meshes are marked in Fig. 7 (the *grey* diamond marker within the upper panel).



(a) **(b)**

Fig. 7: Transcritical flow over a hump with shock. LTS-RKDG2 calculations *vs.* GTS-RKDG2 calculations compared with the analytical solution; **(a)** $lev_{max} = 2$ and **(b)** $lev_{max} = 3$.

At first, the channel's bed is assumed frictionless. Fig. 7a and Fig. 7b display the corresponding steady state profiles acquired by the 1D and 2D versions of the RKDG2 solvers on *mesh-3LTSs* and *mesh-4LTSs*, respectively. It can be seen that the numerical water depths predictions match very well the analytical solution. For the momentum conservation predictions, in terms of steady discharge, the expected conservative state is reached by all the 1D-RKDG2 variants (GTS- and LTS-, and on both meshes) and the 2D-GTS-RKDG2 variant relative to *mesh-3LTSs*. In contrast, the 2D-LTS-RKDG2 variant shows deficit in achieving an fully conservative steady discharge profile; notable also, both 2D-RKDG2 (GTS- and LTS-) models on *mesh-4LTSs* shows the localized discharge spike (Fig. 7b) at the jump's location, which is suspected to occur as a result of a redundant call to the slope-limiter function [39]. However, these side effects remain rather localized and do not appear to affect the whole simulations. These findings indicate that the current LTS-RKDG2 model can maintain the well-balanced property [29] in the 1D formulation but tend to locally disturb momentum conservation in the 2D formulation increasingly with more refinement levels.

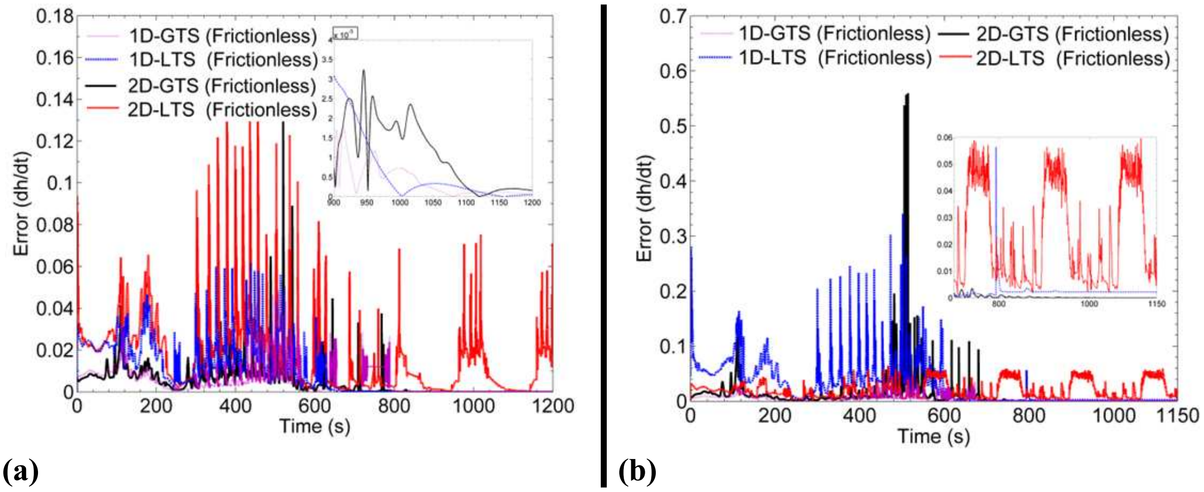


Fig. 8: Transcritical flow over a hump with shock. LTS-RKDG2 calculations *vs.* GTS-RKDG2 convergence rates; **(a)** $lev_{max} = 2$ and **(b)** $lev_{max} = 3$.

Up to t = 2000s, the LTS-RKDG2 model is spotted to reduce the GTS-RKDG2 runtime up to roughly 2X in 1D and 1.5X in 2D (see Table 3). In terms of convergence rates, the L^2-errors defined by the 'variations of the water depth between two successive iterations' were monitored and are illustrated in Fig. 8 (*i.e.,* relative to the output time when the L^2-error of the 2D-GTS-RKDG2 variant became $\leq 10^{-8}$). As shown in Fig. 8a, the convergence error produced by 2D-LTS-RKDG2 variant on *mesh-3LTSs* is seen to alternate steadily; whereas the errors acquired by the other variants appear to follow the expected exponential decay (see the zoom-in portion within the upper-right in Fig. 8a). However, on *mesh-4LTSs* (i.e., Fig. 8b) the 1D-LTS-RKDG2 variant's error appear to stagnate after a certain time while the 2D-LTS-RKDG2 variant's error produces again an alternating pattern (see the zoom-in portion within the upper-right in Fig. 8b). With these results, it appears that the RKDG2 framework risk losing its ability to delivering exponential convergence rates. It can be therefore argued that the present LTS-RKDG2 framework may compromise with either a delay or stagnation in reaching convergence for steady flow simulations (also depending on the dimensionality of the formulation and/or the depth of refinement levels [Fig. 8]).

Table 3: Mesh configurations and runtime ratios after 2000s for test-case 6.1

Simulation case	1D		2D	
Level of refinement	2	3	2	3
Baseline mesh	62	40	62×3	64×4
Domain	[0;1000]	[0;1000]	[0;1000]×[0;12]	[0;1000]×[0;32]
Runtime ratio (GTS/ LTS)	1.9X	2.3X	1.6X	1.5X

Secondly, this test case is used to further point out the inconvenience of the IFTD when solely implemented in conjunction with the LTS-RKDG2 scheme. Therefore, the 1D-LTS-RKDG2 method is reconsidered with a Manning factor of 0.033 s/m$^{1/3}$; the simulations are remade on the same non-uniform meshes (in Table 3) but now with a focus on comparing the IFTD discretization (i.e., time-dependent) *vs.* the explicit friction term discretization (i.e., independent of the time-step). The solution to the momentum equation, in terms of steady discharge numerical result, is appended within the discharge plots of Fig. 7a and 7b. As outlined before (Subsection 5.3.1), the use of the IFTD with the LTS-RKDG2 tends to magnify the impact of the IFTD by increasing the amount of numerical diffusion manifesting itself in form of disturbance in the well-balanced property of the RKDG2 scheme. Further, this side-effect is observed to increase in line with either an increase in the Manning factor (herein, zoom-in of discharge illustrations within Figs. 7a and 7b contains the results relative to the highest value of n_M that was tested, i.e., n_M = 0.033 s/m$^{1/3}$) or in the level of LTS (in that, the LTS-RKDG2-IFTD's discharge prediction in Fig. 7a is less diffusive than the one in Fig. 7b). As anticipated, the discharge solution reproduced by the LTS-RKDG2 scheme with the explicit friction discretization remain comparatively unaffected – despite an insignificant drop that is believed to occur as a results of coarsening the mesh at the boundary and also, perhaps, due to the heuristic nature of Manning's formula. These results justify the motivation to use the proposed hybrid explicit-implicit friction term discretisation (employed from now on for the test cases 6.2-6.5).

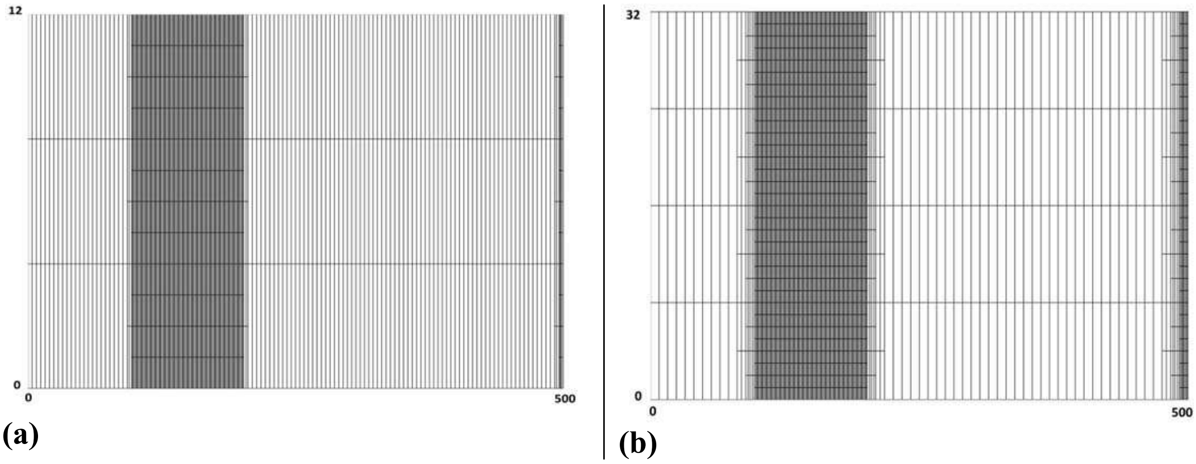


Fig. 9: Wet/dry front advancing and recessing over a rough topography. 2D domains and mesh configurations with local refinement around the steepest topogprahy gradient and at inflow boundary; **(a)** lev_{max} = 2 and **(b)** lev_{max} = 3.

6.2 Wet/dry front advancing and recessing over a rough topography

This synthetic tidal wave case was initiated by Heniche et al. [40] and is a commonly used test case to verify the stability and robustness of a numerical model when reproducing the movement of a wet/dry front over an uneven and rough topography. It can be regarded as a tidal wave running up and down over sloping beach in a 1D domain [0m; 500m] with a slope of -0.001 over [0m; 100m], -0.01 over]100m; 200m] and -0.001 over]200m ;500m]. The friction effects are quite significant as they associate to a Manning coefficient of n_M = 0.03. The flow is initially still with a constant surface elevation of 1.75m. The eastern end of the domain (x = 500m) is assumed to be the inlet where the varying water depth reads

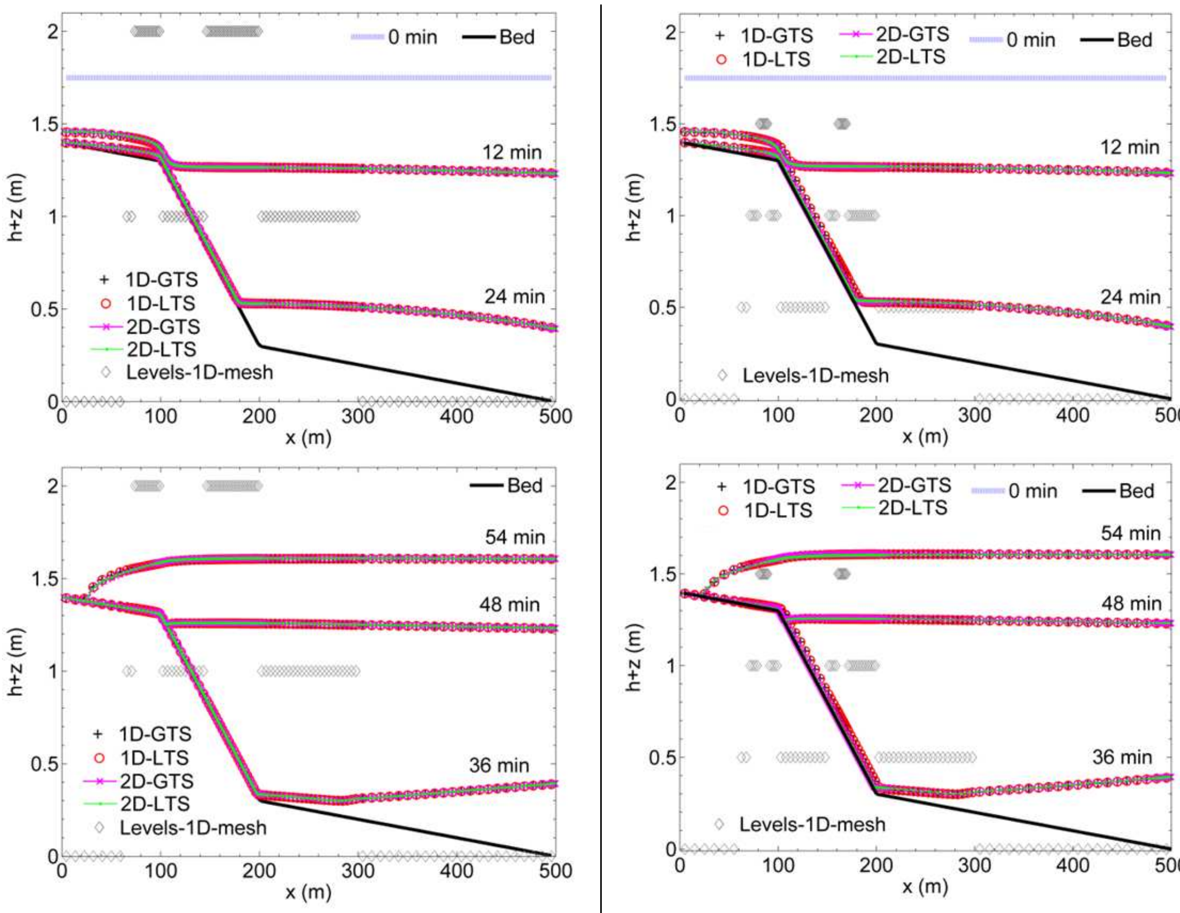
$$h(500,t) = 1 + 0.75\cos\left(\frac{2\pi t}{T}\right) \tag{35}$$

which mimics a tidal wave with T = 60min representing the period of a tidal cycle. The western end of the domain is a standing solid wall.

Table 4: Mesh configurations and runtime ratios after 60 min for test-case 6.2

Simulation case	1D		2D	
Level of refinement	2	3	2	3

Baseline mesh	62	50	124×3	62×4
Domain	[0;500]	[0;500]	[0;500]×[0;12]	[0;500]×[0;32]
Runtime ratio (GTS/ LTS)	1.4X	2.5X	1.18X	1.23X



(a) **(b)**

Fig. 10: Wet/dry front advancing and recessing over a rough topography. LTS-RKDG2 calculations *vs.* GTS-RKDG2 calculations **(a)** lev_{max} = 2 and **(b)** lev_{max} = 3.

1D and 2D, LTS- and GTS-, RKDG2 runs on the meshes configurations described in Table 4 are performed. The employed meshes, of type *mesh-3LTSs* and *mesh-4LTSs*, are displayed in Fig. 9 for the 2D case whereas for 1D case the meshes properties are marked within Fig. 10 (for convenience, the marker's plots in Fig. 10b are shrank by a factor of 0.5). The simulations output time is 60min (*i.e.,* one tidal cycle). The LTS- and GTS- RKDG2 solutions of the advancing and recessing shoreline, at t = 0, 12, 24, 36, 48 and 54min are presented in Fig. 10a and Fig. 10b, respectively, on *mesh-3LTSs* and *mesh-4LTSs*. Apparently, here, the LTS-RKDG2 and GTS-RKDG2 predictions agree very closely and also match those presented in literature (*e.g.*, in [27]). Nevertheless, for this test, as summarizes Table 4, the LTS-RKDG2 is found less costly than the GTS-RKDG2; namely the relative saving in runtime is about 1.2X in 2D and reached 2.5X for the 1D case on *mesh-4LTSs*.

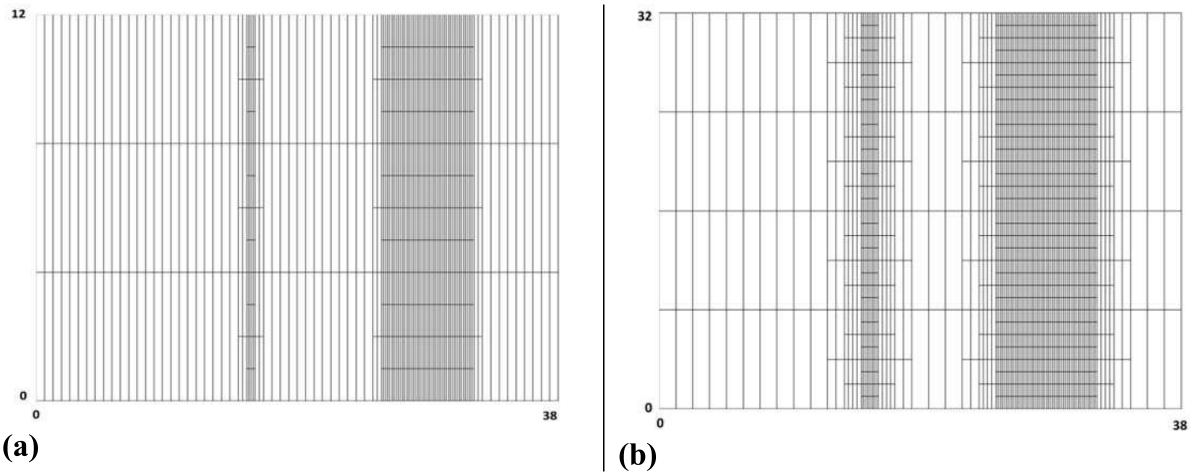


Fig. 11: Dam-break flow interacting with a triangular obstacle. 2D domains and mesh configurations with refinement at the local of the initial dam and around the triangular obstacle; **(a)** lev_{max} = 2 and **(b)** lev_{max} = 3.

6.3 Dam-break wave interacting with a triangular obstacle

The RKDG2 schemes are here assessed by replicating an experimental test case from the CADAM project [41]. It consist of a violent breaching wave propagating over an initially dry and rough floodplain, overtopping a triangular obstacle and then interacting with it. The length of the domain is 38m; the initial condition is a still water state of 0.75 m held by an imaginary dam (located at x = 15.5m) and a dry floodplain downstream of the dam (see Fig. 12). For this problem, measured time histories of the water depth are available at point G10, G11, G13 and G20 that are respectively located 10 m, 11 m, 13 m and 20 m downstream of the dam's location. The friction effects are associated to a Manning factor of 0.0125.

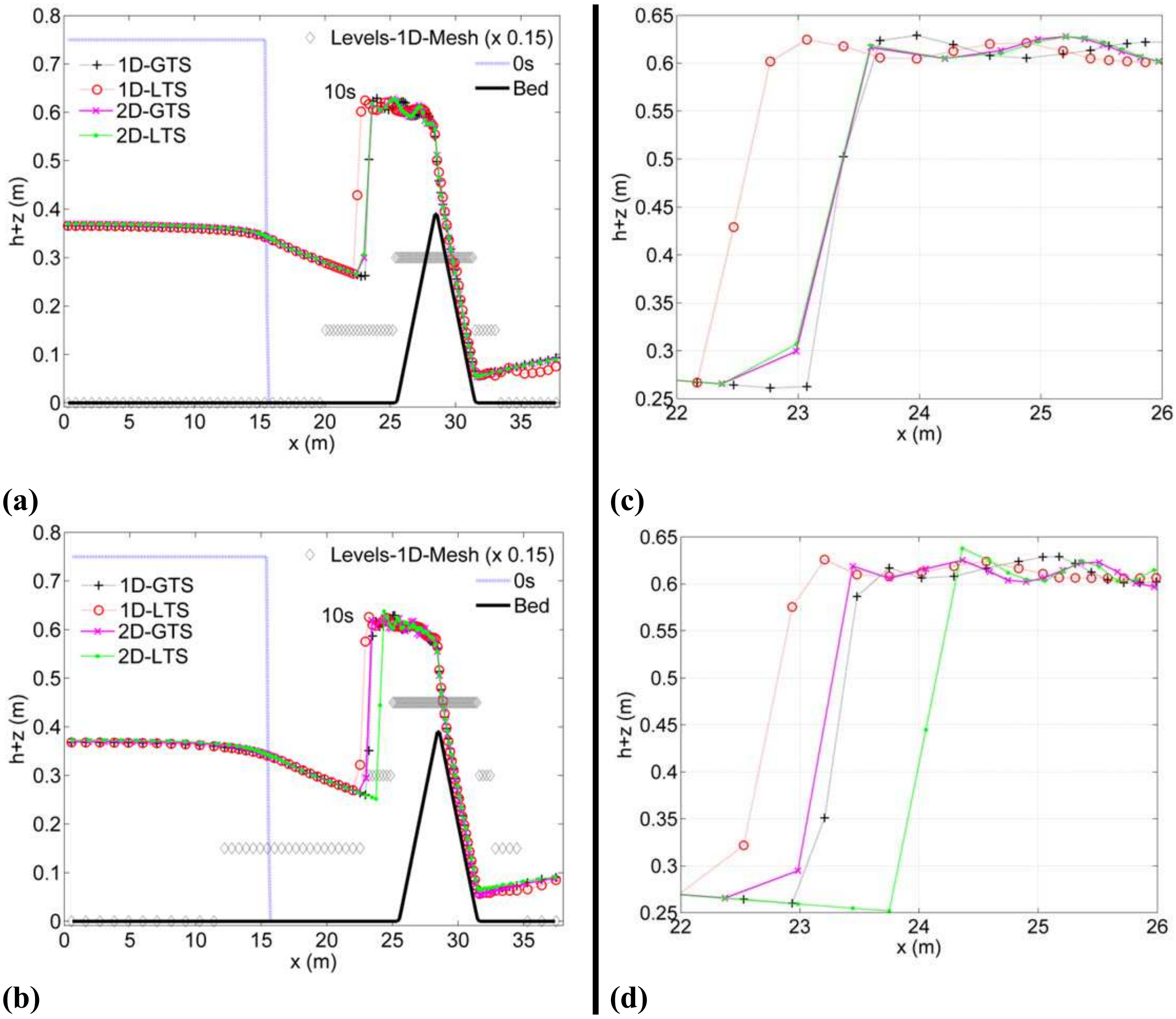
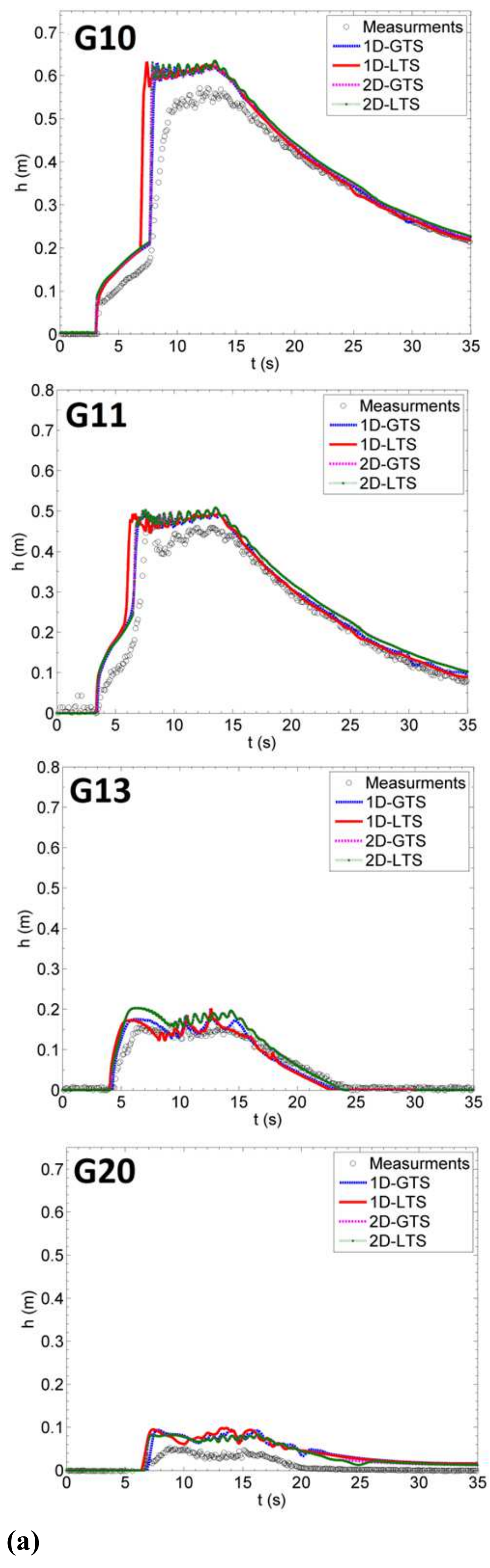


Fig. 12: Dam-break flow interacting with a triangular obstacle at t = 10s. LTS-RKDG2 calculations *vs.* GTS-RKDG2 water-surface profiles; **(a)** lev_{max} = 2 and **(b)** lev_{max} = 3, (c) zoom in around the shock wave lev_{max} = 2, and **(d)** zoom in around the shock wave lev_{max} = 3.

The upstream boundary is a solid wall while free outflow condition is permitted at the downstream boundary. Simulations are executed using the LTS- and GTS- RKDG2 variants with the mesh setups described in Table 5; *mesh-3LTSs* and *mesh-4LTSs* used for the 2D case are viewed in Fig. 11; for the 1D case, the meshes are described within Fig. 12 (i.e. the markers). The output simulation time is t = 35s. A view of the free-surface elevation longitudinal profiles predicted by the all RKDG2 versions is available in Fig. 12 at time t = 10s. Moreover, Fig. 13 contains the predicted time histories that are seen to favourably track with the measured profiles. As previews Fig. 12, the generated wave front propagates to the obstacle, climbs up and overtops the obstacle, creates a shock-wave moving to the upstream wall. A magnified view on the shock-capturing ability of the RKDG2 models (in Fig. 12c and Fig. 12d) shows a remarkable agreement between the 2D models (GTS- and LTS) and the 1D-GTS models for the simulations involving '3' refinement levels. However, this agreement appears to slightly decline when '4' levels were considered in the simulations; namely for the 2D-LTS-RKDG2 variant that predicted a delay in the capture of the shock as compared to the GTS versions (in 1D and 2D). As to the 1D-LTS-RKDG2, here, it displays a tendency to accelerate shock-capturing in all simulations. These implications thus favour the use of the 2D-LTS-RKDG2 model on *mesh-3LTSs* over any other LTS variant for this test. Taken as whole, all LTS- and GTS- RKDG2 variants successfully survived this benchmark showing slight differences throughout the whole simulations (see Fig. 13), which seem to have inconsequential effects on the stability of the LTS-RKDG2 models. The over-predictive aspect delivered by the RKDG2 predictions at G20 has no concern with the numerical algorithms; it is usually credited to the fact that the wave pattern downstream of the obstacle becomes highly complex and unstable and so the hydrostatic assumption of the shallow water equations is no longer valid. In terms of runtime saving, as shows Table 5, the use of LTS-RKDG2 scheme is on average 1.3X and 1.18X for the 1D and the 2D versions, respectively.

Table 5: Mesh configurations and runtime ratios after 35s for test-case 6.3

Simulation case	1D		2D	
Level of refinement	2	3	2	3
Baseline mesh	63	35	62×3	31×4
Domain	[0;38]	[0;38]	[0;38]×[0;12]	[0;38]×[0;32]
Runtime ratio (GTS/ LTS)	1.32X	1.36X	1.16X	1.21X



634 **Fig. 13:** Dam-break flow interacting with a triangular obstacle. Time histories produced by the RKDG2 calculations compared with measured data; **(a)** lev_{max} = 2 and **(b)** lev_{max} = 3.

6.4 2D smooth oscillatory flow in a parabolic bowl with friction

Sampson's 2D analytical test [42] is employed to study second-order mesh convergence for the RKDG2 schemes on the non-uniform mesh configuration (both LTS- and GTS- versions in 2D) and further assess their performance in handling frictional flow with wetting and drying over irregular topography. This test is featured by a constantly-moving wet/dry (circular) shoreline inside the 2D parabolic terrain $z(x,y) = h_0(x^2 + y^2)/a^2$, where h_0 and a are constants. The energy dissipation, due to friction, is assumed proportional to the magnitude of the discharge and can be integrated by altering C_f to $C_f = h\tau/\sqrt{u^2+v^2}$, where τ represents a bed-friction parameter. A 2D analytical solution can be obtainable when $\tau < p$, where $p = \sqrt{8gh_0/a^2}$ represents the peak amplitude. With this setting, the exact solution follows

$$\begin{cases} \eta(x,y,t) = h_0 - \frac{B^2}{2g} e^{-t\tau} - \frac{B}{g} e^{-t\tau/2} \left\{ \left[\frac{\tau}{2}\sin(wt) + s\cos(wt) \right] x + \left[\frac{\tau}{2}\cos(wt) + s\sin(wt) \right] y \right\} \\ u(t) = Be^{-t\tau/2}\sin(wt) \\ v(t) = -Be^{-t\tau/2}\cos(wt) \end{cases} \quad (36)$$

Where B is a velocity constant and $w = \sqrt{p^2 - \tau^2}/2$. Herein, the 2D domain is chosen to be [-5000; 5000]^2 and the constants are set to h_0 = 10m, B = 5m/s, a = 3000m and $\tau = 0.009$ s^{-1}, which is a relatively high friction factor (as $\tau = 0.009 < 0.0093 = p$). For the frictionless case (i.e., $\tau =$ 0), the flow would oscillates indefinitely with a period cycle of $T = 2\pi/w \approx$ 1345.7104s. But with the inclusion of friction effects the oscillatory flow is expected to cease into the state $\eta(x,y,\infty) = h_0$, $u(\infty) = 0$ and $v(\infty) = 0$.

The initial conditions for the flow variables are obtained from (36), evaluated at t = 0s, and the output time is t = 2Ts. Since the flow does not reach the 2D domain's boundaries, any boundary condition can be specified. To undergo the mesh convergence study, two series of simulations are run on meshes of type *mesh-3LTSs* and *mesh-4LTSs*. The baseline mesh details for the first and second series of simulations are, respectively, listed in Table 6 and Table 7. Qualitatively, however, to save space, we only show the mesh patterns associated to the coarsest baseline mesh (*i.e.,* Fig. 14) used in each series of simulations; the corresponding initial contour map of the water depth is also illustrated in Fig. 14.

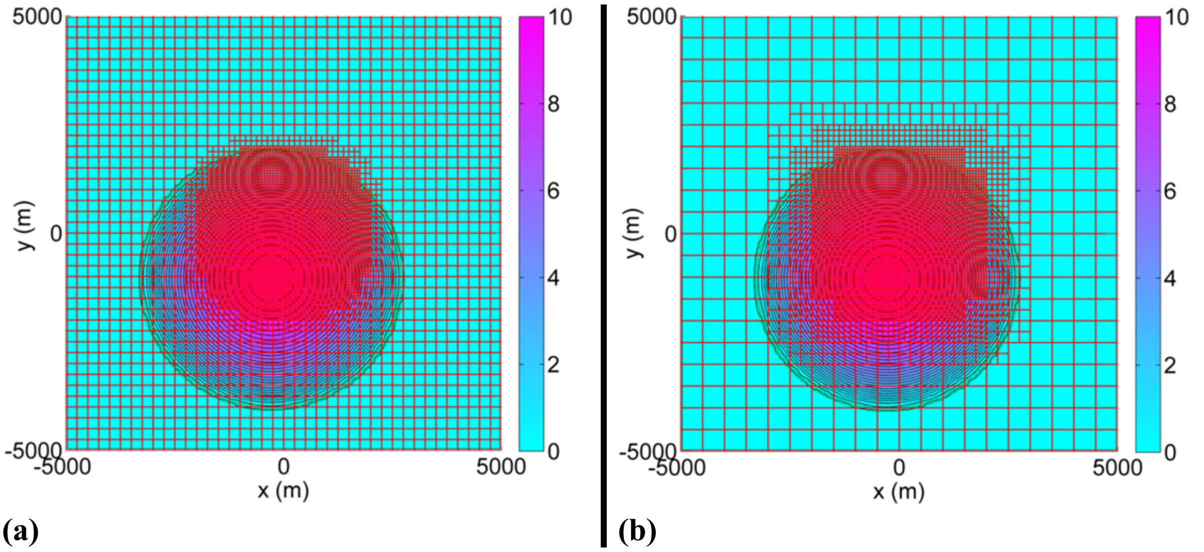


Fig. 14: Oscillatory flow in a parabolic bowl with friction. Initial water-depth condition, 2D domain and mesh configurations with a refined portion; **(a)** baseline mesh 40×40 with lev_{max} = 2 and **(b)** baseline mesh 20×20 with lev_{max} = 3.

The outputs of the 2D-LTS-RKDG2 and 2D-GTS-RKDG2 versions, at the time $T/2$s, are used to calculate the L^2-errors (and associated and L^2-orders) along the *x*-direction centreline. The quantitative results are summarized in Table 6 and Table 7, which also list the runtime ratios respective to the output time $t = 2T$s. As indicates Tables 6, both GTS- and LTS-models are noted to acquire second-order mesh-convergence on the mesh of type *mesh-3LTSs*. But for these runs, the 2D-LTS-RKDG2 variant is noted to be more expensive than the 2D-GTS-RKDG2 variant. In contrast, as point out Table 7, the 2D-LTS-RKDG2 scheme provide relative reduction in the runtime cost by a mean factor of 1.2X for the case involving a mesh of type *mesh-4LTSs*. However, on the latter setting, the RKDG2 schemes (both LTS- and GTS-) do not seem to achieve second-order convergence one the latter mesh patterns. Remarkably, these results suggest that increasing the deepness of spatial refinement levels –

although works in the favour of efficiency – pays off accuracy as such [17]; despite the complementary effects (*e.g.*, flux reinforcement in time) associated with the LTS algorithms. Thus, the question of how to comprehensively ensure conservative data (and fluxes) transfer and recovery across the heterogeneous spatial and/or temporal scales on-uniform meshes is yet to be resolved (note that, on uniform meshes, the RKDG2 delivers second-order convergence rates for this test case [16, 35]).

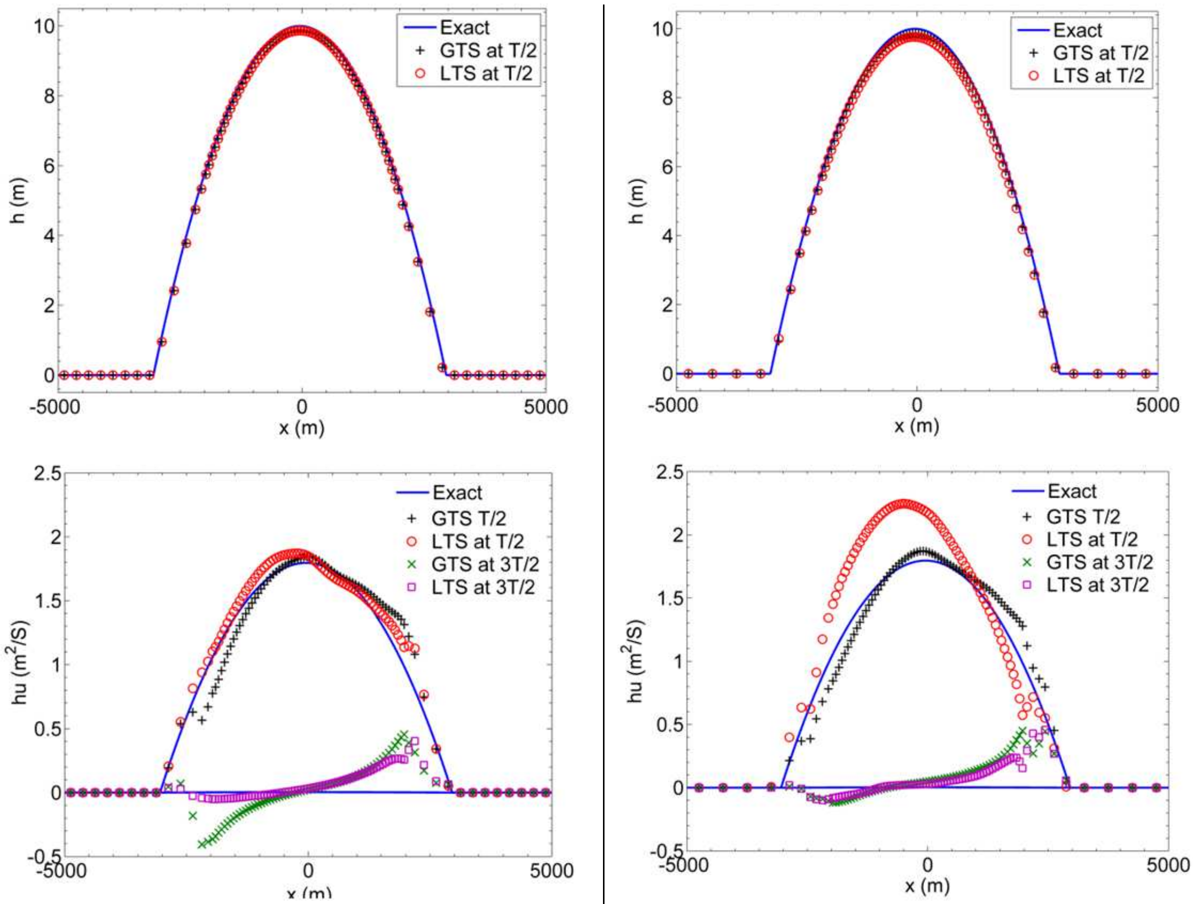
Table 6: Case of Lev_{max} = **2**. L^2-errors and -orders evaluated at ***T*/2s** and runtime ratios at **2*T*s**.

Baseline Mesh	2D-GTS-RKDG2				2D-LTS-RKDG2				Runtime ratio (GTS/ LTS)
	Error(h)	Order(h)	Error(hu)	Order(hu)	Error(h)	Order(h)	Error(hu)	Order(hu)	
40 × 40	4.50e-03	--	6.26e-04	--	3.91e-03	--	3.91e-04	--	0.18X
80 × 80	3.41e-04	1.89	1.25e-04	2.31	3.03e-04	1.87	9.77e-05	2.00	0.60X
160 × 160	2.09e-05	2.16	1.69e-05	2.88	2.03e-05	2.12	1.40e-05	2.80	0.86X

Table 7: Case of Lev_{max} = **3**. L^2-errors and -orders evaluated at ***T*/2s** and runtime ratios at **2*T*s**.

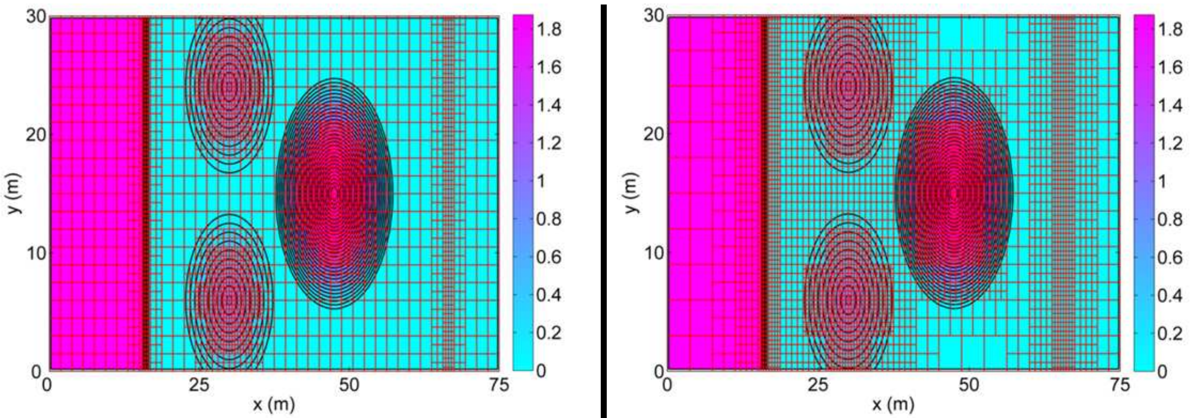
Baseline Mesh	2D-GTS-RKDG2				2D-LTS-RKDG2				Runtime ratio (GTS/ LTS)
	Error(h)	Order(h)	Error(hu)	Order(hu)	Error(h)	Order(h)	Error(hu)	Order(hu)	
20 × 20	1.19e-04	--	5.76e-04	--	1.74e-04	--	1.59e-03	--	1.3X
40 × 40	5.01e-05	1.25	2.50e-04	1.20	8.51e-05	1.03	7.67e-04	0.97	1.21X
80 × 80	2.08e-05	1.26	4.47e-05	2.50	4.01e-05	1.08	3.72e-04	1.04	1.25X

Fig. 15 compares the numerical predictions with the analytical solution along the *x*-direction centreline for the water depth variable at *T*/2s (upper panel) and the discharge variable at T/2s and 3*T*/2s (lower panel). Fig. 15 supports the aforementioned argument (revealed in Table 6 and Table 7); the predictions delivered by the all RKDG2 schemes (LTS- and GTS-) using less level of refinement (in space for the GTS and further in time for the LTS version) match much better the exact solution. Remarkable also, the 2D-LTS-RKDG2 discharge prediction is much more deviated from the 2D-GTS-RKDG2 on the mesh with the more refinement levels; thus suggestive of a cumulative effect occurring further from the temporal transfer of information (in the 2D-LTS-RKDG2) across the levels of resolution. In terms of modelling the moving wet/dry shoreline, all RKDG2 schemes successful tracked the constantlyvanishing velocity zone (see discharge plots at $3T/2$s in Fig. 15 [lower panel]) with no signs of a conflict between LTS and wetting and drying.



(a) **(b)**

Fig. 15: Oscillatory flow in a parabolic bowl with friction. LTS-RKDG2 calculations *vs.* GTS-RKDG2 calculations across the *x*-direction centreline **(a)** baseline mesh 40×40 with $lev_{max} = 2$ and **(b)** baseline mesh 20×20 with $lev_{max} = 3$.



(a) **(b)**

Fig. 16: 2D breaking wave over dry floodplain with friction. Initial free-surface elevation condition, 2D domain and mesh configuration with refined portions; **(a)** $lev_{max} = 2$ and **(b)** $lev_{max} = 3$.

6.5 2D breaking wave over dry floodplain with friction

This test may be regarded as the 2D version of the test investigated in Subsection 6.3. It is widely used as a 2D standard benchmark to assess the adequacy of computational flood models for realistic applications [17]. The 2D domain is [0; 75m]×[0; 30m] that is assumed to be enclosed by solid-walls and to initially hold a tranquil water body of 1.875m upstream of a dam located at x = 16m. Downstream of the dam, the floodplain is dry with three topographic hills (see Fig. 16) and is characterized by a roughness Manning coefficient of 0.0185. 2D-LTS-RKDG2 and 2D-GTS-RKDG2 simulations are executed on a mesh of type *mesh-3LTSs* and *mesh-4LTSs*, respectively, which are described in Table 8 and illustrated in Fig. 16. The 2D contour maps of the free-surface elevation produced by the RKDG2 models at t = 6s, 12s, and 24s are presented in Fig. 17 (*mesh-3LTSs*) and Fig. 18 (*mesh-4LTSs*). On both meshes, the LTS- and GTS-RKDG2 versions predicted nearly similar local of flow features (of shock, smooth and wet/dry character). However, the contour patterns among the LTS-RKDG2 and GTS-RKDG2 schemes correlate much better on *mesh-3LTSs* where the LTS-RKDG2 coordinate less LTSs (contrast Fig. 17 *vs.* Fig. 18). Whereas, on *mesh-4LTSs* the LTS-RKDG2 predictions are more deviated and thus again indicate of a cumulative effect associated with the depth of refinement levels.

Table 8: Mesh and runtime ratios after 24s for test-case 6.5

Simulation case	2D	
Level of refinement	2	3
Baseline mesh	40×20	20×10
Domain	[0;75]×[0;30]	[0;75]×[0;30]
Runtime ratio (GTS/ LTS)	0.5X	0.98X

In terms of runtime cost (Table 8) no runtime saving are here noted in the LTS-RKDG2 models performance, over the traditional GTS version. Possibly, such inefficiency is associated with the relatively high number of fine-cells and the presence of very high velocities. This suggests that the LTS-RKDG2 model would be able to speed-up simulation times, in 2D, when the percentage of fine cells represents a very small portion of the 2D mesh and for low flow speed.

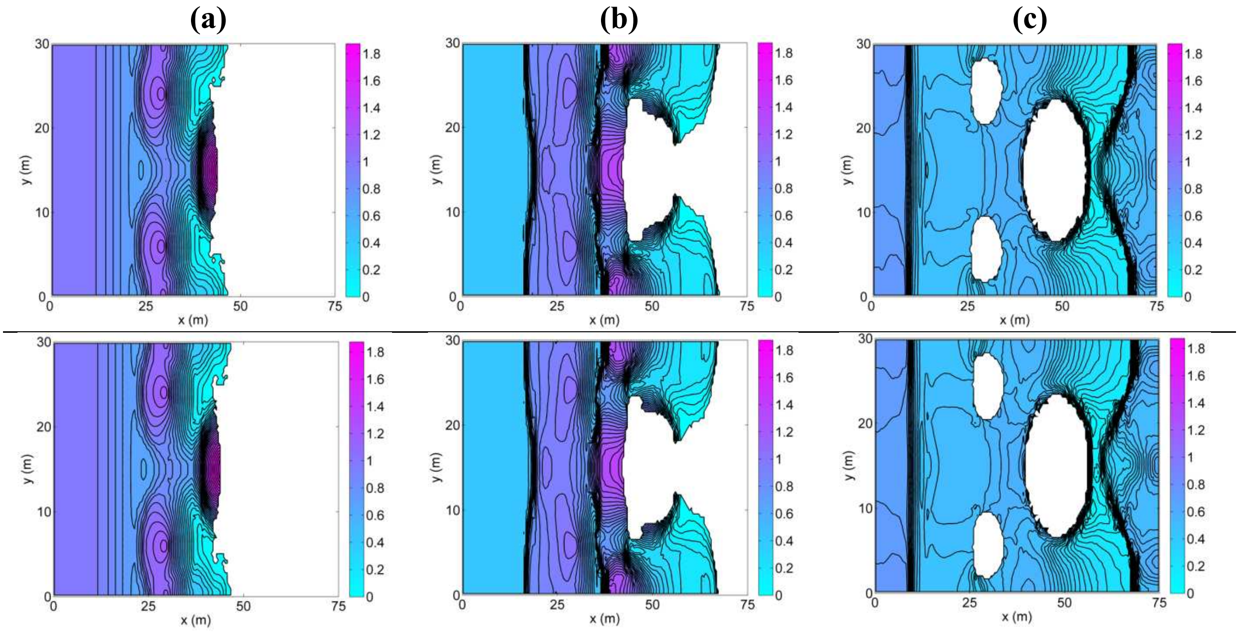


Fig. 17: 2D breaking wave over dry floodplain with friction. Contrasting the free-surface elevation contours obtained by the LTS-RKDG2 (lower panel) and the GTS-RKDG2 (upper panel) for $\boldsymbol{lev_{max}} = \mathbf{2}$; **(a)** t = 6s, **(b)** t = 12s and **(c)** t = 24s.

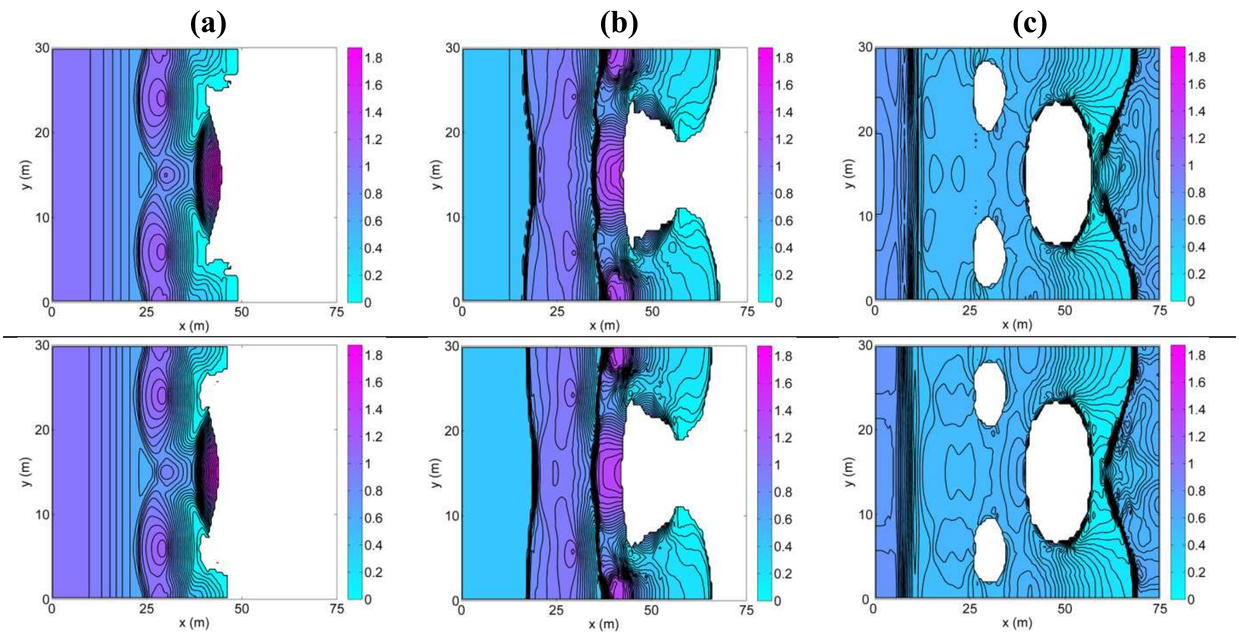


Fig. 18: 2D breaking wave over dry floodplain with friction. Contrasting the free-surface elevation contours obtained by the LTS-RKDG2 (lower panel) and the GTS-RKDG2 (upper panel) for $\boldsymbol{lev_{max}} = \mathbf{3}$; **(a)** t = 6s, **(b)** t = 12s and **(c)** t = 24s.

7. Conclusions

A LTS algorithm [24], which involves a small calculation stencil, has been integrated with a robust RKDG2 shallow water model on structured non-uniform meshes (LTS-RKDG2). Most advanced stabilizing features that enable the practical use of shallow water numerical models – previously available within the traditional GTS-RKDG2 version, i.e. for controlling slope coefficients, handling complex domain topography and wetting and drying [17] – were retained within the LTS-RKDG2 design. However further considerations were given to maintain the flux conservation (in time) across cells of different sizes, and to diminish the adverse effects of the IFTD (Implicit Friction Term Discretisation). 1D and 2D versions of the LTS-RKDG2 model were setup and ran on non-uniform meshes of type '*mesh-3LTSs*' and '*mesh-4LTSs*' that, respectively, comprised '3' and '4' levels of local spatial discretization (*e.g.,* $\{\Delta x, \Delta x/2, \Delta x/4\}$ and $\{\Delta x, \Delta x/2, \Delta x/4, \Delta x/8\}$ for the 1D meshes). On these meshes, the LTS-RKDG2 model adapted correspondingly LTSs of $\{\Delta t, \Delta t/2, \Delta t/4\}$ and $\{\Delta t, \Delta t/2, \Delta t/4, \Delta t/8\}$, whereas the GTS-RKDG2 model used the smallest GTS allowable. Selected test cases were employed to verify the LTS-RKDG2 models' implementation with respect to the associated GTS-RKDG2 schemes considering realistic aspects of hydraulic modelling.

In all tests, the LTS-RKDG2 schemes were able to generically produce very close prediction as the GTS-RKDG2 despite the presence of water jumps, irregular topographies and wetting and drying. A closer analysis of the results, however, suggest that the LTS-RKDG2 model might lose its exponential convergence property for steady state simulations, its overall second-order mesh-convergence for the case involving more depth in the spatio-temporal refinement increasingly with the dimensionality of the formulation and the deepness of refinement levels.

Table 9: Range of the relative runtime savings.

Runtime ratio (GTS /GLS)	*1D simulations*	*2D simulations*
Mesh of type "mesh-3LTSs"	1.3—2.0X	0.18—1.6X
Mesh of type "mesh-4LTSs"	1.36—2.5X	0.98—1.5X

In terms of runtime saving relative to the GTS-RKDG2 simulations, for the test cases investigated in this study (Table 9), the 1D LTS-RKDG2 formulation has speeded up efficiency by an average factor of 2; whereas, the 2D formulation relatively offered saving of around average factor of 1.6. The maximum efficiency speed up has been observed in the tests involving a relatively small proportion of fine cells (Subsection 6.1) and/or a low velocity flows (Subsection 6.2), and when more levels of spatio-temporal adaptation have been employed (*mesh-4LTSs*). For violent flows and/or cases where the mesh involves a significant portion of fine cells, LTS-RKDG2 models have been found to be much less effective. Most notably, its 2D formulation has provided very little saving for on meshes of type *mesh-4LTSs* and no saving at all for meshes of type *mesh-3LTSs*.

Based on the present findings, we essentially recommend the use of LTS-RKDG2 model on non-uniform meshes in which the refined portion constitutes a very small percentage of the global domain, namely in 2D simulations. Otherwise, the saving in runtime gained by the integration of the LTS algorithm would be eliminated by extra operational cost entailed at those cells that are smaller than the coarsest cells. Moreover, in the interest of accuracy, conservation and economy, it would be further beneficial to tailor a LTS-RKDG2 version with the least levels of LTSs. The improvement and/or extension of proposed LTS approach to higher than second-order RKDG formulation is hindered by the need of more comprehensive space-time interpolation formula and the need to cope with more inner stages within the RK mechanism.

Acknowledgement

The present work was partially developed within the framework of the Panta Rhei Research Initiative of the International Association of Hydrological Sciences. This research is supported by the UK Engineering and Physical Sciences Research Council (grants ID: EP/F030177/1 and EP/K031023/1).

References

1. Guinot, V., ed. *Godunov-type schemes: an introduction for engineers*. 2003, Elsevier: Amsterdam.
2. Toro, E.F., ed. *Shock-capturing methods for free-surface shallow flows*. 2001, John Wiley & Sons, Ltd.
3. Toro, E.F. and García-Navarro, P., *Godunov-type methods for free-surface shallow flows: A review.* Journal of Hydraulic Research, 2007. **45**(6): p. 736-751.
4. Delis, A.I. and Kampanis, N.A., *Numerical flood simulation by depth averaged free surface flow models*, in *Environmental Systems, in Encyclopedia of Life Support Systems (EOLSS)*, A. Sydow, Editor. 2009.
5. Gallegos, H.A., Schubert, J.E., and Sanders, B.F., *Two-dimensional, high-resolution modeling of urban dam-break flooding: A case study of Baldwin Hills, California.* Advances in Water Resources, 2009. **32**(8): p. 1323-1335.
6. Liang, Q., Du, G., Hall, J.W., and Borthwick, A.G.L., *Flood Inundation Modeling with an Adaptive Quadtree Grid Shallow Water Equation Solver.* Journal of Hydraulic Engineering, 2008. **134**(11): p. 1603-1610.
7. Lacasta, A., García-Navarro, P., Burguete, J., and Murillo, J., *Preprocess static subdomain decomposition in practical cases of 2D unsteady hydraulic simulation.* Computers & Fluids, 2013. **80**(0): p. 225-232.

8. Sanders, B.F., Schubert, J.E., and Detwiler, R.L., *ParBreZo: A parallel, unstructured grid, Godunov-type, shallow-water code for high-resolution flood inundation modeling at the regional scale.* Advances in Water Resources, 2010. **33**(12): p. 1456-1467.

9. Brodtkorb, A.R., Sætra, M.L., and Altinakar, M., *Efficient shallow water simulations on GPUs: Implementation, visualization, verification, and validation.* Computers & Fluids, 2012. **55**(0): p. 1-12.

10. Smith, L.S. and Liang, Q., *Towards a generalised GPU/CPU shallow-flow modelling tool.* Computers & Fluids, 2013. **88**(0): p. 334-343.

11. Eskilsson, C., El-Khamra, Y., Rideout, D., Allen, G., Chen, Q.J., and Tyagi, M., *A Parallel High-Order Discontinuous Galerkin Shallow Water Model*, in *Computational Science – ICCS 2009*, G. Allen, et al., Editors. 2009, Springer Berlin Heidelberg. p. 63-72.

12. Courant, R., Friedrichs, K., and Lewy, H., *Über die partiellen Differenzengleichungen der mathematischen Physik.* Mathematische Annalen, 1928. **100**(1): p. 32-74.

13. Zhou, T., Li, Y., and Shu, C.W., *Numerical comparison of WENO finite volume and Runge–Kutta discontinuous Galerkin methods.* Journal of Scientific Computing, 2001. **16**(2): p. 145-171.

14. Zhang, M. and Shu, C.W., *An analysis of and a comparison between the discontinuous Galerkin and the spectral finite volume methods.* Computers & Fluids, 2003. **34**(4-5): p. 581-592.

15. Kesserwani, G. and Liang, Q., *Locally Limited and Fully Conserved RKDG2 Shallow Water Solutions with Wetting and Drying.* Journal of Scientific Computing, 2012: p. 1-25.

16. Kesserwani, G. and Liang, Q., *A discontinuous Galerkin algorithm for the two-dimensional shallow water equations.* Computer Methods in Applied Mechanics and Engineering, 2010. **199**(49-52): p. 3356-3368.

17. Kesserwani, G. and Liang, Q., *Dynamically adaptive grid based discontinuous Galerkin shallow water model.* Advances in Water Resources, 2012. **37**: p. 23-39.

18. Wirasaet, D., Tanaka, S., Kubatko, E.J., Westerink, J.J., and Dawson, C., *A performance comparison of nodal discontinuous Galerkin methods on triangles and quadrilaterals.* International Journal for Numerical Methods in Fluids, 2010. **64**(10-12): p. 1336-1362.

19. Crossley, A.J. and Wright, N.G., *Time accurate local time stepping for the unsteady shallow water equations.* International Journal for Numerical Methods in Fluids, 2005. **48**(7): p. 775-799.

20. Sanders, B.F., *Integration of a shallow water model with a local time step.* Journal of Hydraulic Research, 2008. **46**(4): p. 466-475.

21. Constantinescu, E. and Sandu, A., *Multirate Timestepping Methods for Hyperbolic Conservation Laws.* Journal of Scientific Computing, 2007. **33**(3): p. 239-278.

22. Seny, B., Lambrechts, J., Comblen, R., Legat, V., and Remacle, J.F., *Multirate time stepping for accelerating explicit discontinuous Galerkin computations with application to geophysical flows.* International Journal for Numerical Methods in Fluids, 2013. **71**(1): p. 41-64.

23. Trahan, C.J. and Dawson, C., *Local time-stepping in Runge–Kutta discontinuous Galerkin finite element methods applied to the shallow-water equations.* Computer Methods in Applied Mechanics and Engineering, 2012. **217–220**(0): p. 139-152.

24. Krivodonova, L., *An efficient local time-stepping scheme for solution of nonlinear conservation laws.* Journal of Computational Physics, 2010. **229**(22): p. 8537-8551.

25. Krivodonova, L., Xin, J., Remacle, J.F., Chevaugeon, N., and Flaherty, J.E., *Shock detection and limiting with discontinuous Galerkin methods for hyperbolic conservation laws.* Applied Numerical Mathematics, 2004. **48**(3-4): p. 323-338.
26. Xing, Y. and Shu, C.W., *A new approach of high order well-balanced finite volume WENO schemes and discontinuous Galerkin methods for a class of hyperbolic systems with source terms.* Communication in Computational Physics, 2006. **1**(1): p. 101-135.
27. Liang, Q. and Marche, F., *Numerical resolution of well-balanced shallow water equations with complex source terms.* Advances in Water Resources, 2009. **32**(6): p. 873-884.
28. Audusse, E., Bouchut, F., Bristeau, M., Klein, R., and Perthame, B., *A fast and stable well-balanced scheme with hydrostatic reconstruction for shallow water flows.* SIAM Journal on Scientific Computing, 2004. **25**(6): p. 2050-2065.
29. Kesserwani, G., Liang, Q., Vazquez, J., and Mosé, R., *Well-balancing issues related to the RKDG2 scheme for the shallow water equations.* International Journal for Numerical Methods in Fluids, 2010. **62**(4): p. 428-448.
30. Brufau, P., Vázquez-Cendón, M.E., and García-Navarro, P., *A numerical model for the flooding and drying of irregular domains.* International Journal for Numerical Methods in Fluids, 2002. **39**(3): p. 247-275.
31. Bader, M., Bock, C., Schwaiger, J., and Scaba, B., *Dynamically adaptive simulations with minimal memory requirement—solving the shallow water equation using sierpinski curves.* SIAM Journal Scientific Computing, 2010. **32**(1): p. 212-228.
32. Cockburn, B. and Shu, C.-W., *The Runge-Kutta local projection P1-discontinuous Galerkin method for scalar conservation laws.* RAIRO Modél Math Anal Numér 1991. **25**(3): p. 337–361.

33. Cockburn, B. and Shu, C.-W., *Runge-Kutta discontinuous Galerkin methods for convection-dominated problems.* Journal of Scientific Computing, 2001. **16**(3): p. 173-261.

34. Shu, C.-W. and Osher, S., *Efficient implementation of essentially non-oscillatory shock-capturing schemes.* Journal of Computational Physics, 1988. **77**(2): p. 439-471.

35. Kesserwani, G. and Liang, Q., *Locally Limited and Fully Conserved RKDG2 Shallow Water Solutions with Wetting and Drying.* J. Sci. Comput., 2012. **50**(1): p. 120-144.

36. Burguete, J., Garcia-Navarro, P., Murillo, J., and Garcia-Palacin, I., *Analysis of the Friction Term in the One-Dimensional Shallow-Water Model.* Journal of Hydraulic Engineering, 2007. **133**(9): p. 1048-1063.

37. Murillo, J., García-Navarro, P., and Burguete, J., *Time step restrictions for well-balanced shallow water solutions in non-zero velocity steady states.* International Journal for Numerical Methods in Fluids, 2009. **60**(12): p. 1351-1377.

38. Meselhe, E.A., Sotiropoulos, F., and Holly Jr, F.M., *Numerical Simulation of Transcritical Flow in Open Channels.* Journal of Hydraulic Engineering, 1997. **123**(9): p. 774-783.

39. Kesserwani, G. and Liang, Q., *Influence of Total-Variation-Diminishing Slope Limiting on Local Discontinuous Galerkin Solutions of the Shallow Water Equations.* Journal of Hydraulic Engineering, 2012. **138**(2): p. 216-222.

40. Heniche, M., Secretan, Y., Boudreau, P., and Leclerc, M., *A two-dimensional finite element drying-wetting shallow water model for rivers and estuaries.* Advances in Water Resources, 2000. **23**(4): p. 359-372.

41. Hiver, J.M. *Adverse-slope and slope (bump).* in *Concerted Action on Dam Break Modelling: Objectives, Project Report, Test Cases, Meeting Proceedings*. 2000.

Université catholique de Louvain, Civ. Eng. Dept., Hydraulics Division, Louvain-la-Neuve, Belgium.

42. Sampson, J., Easton, A., and Singh, M. *Moving Boundary Shallow Water Flow in Circular Paraboloidal Basins*. in *EMAC 2003 Proceedings: 5th International Congress on Industrial and Applied Mathematics*. 2003: University of Technology, Sydney - Faculty of Science.