Large CFL explicit scheme for one-dimensional shallow water equations

Vincent Guinot, Antoine Rousseau

Large CFL explicit scheme for one-dimensional shallow water equations

V. Guinot[1,3] and A. Rousseau*[1,2]

[1]Inria LEMON, Inria, Univ. Montpellier, France
[2]IMAG, Univ. Montpellier, CNRS, Montpellier, France
[3]HydroSciences Montpellier (HSM), Univ. Montpellier, CNRS, IRD, Montpellier, France

April 18, 2023

Abstract

A large CFL algorithm is presented for the explicit, finite volume solution of hyperbolic systems of conservation laws , with a focus on the shallow water equations. The Riemann problems used in the flux computation are determined using averaging kernels that extend over several computational cells. The usual Courant-Friedrichs-Lewy stability constraint is replaced with a constraint involving the kernel support size. This makes the method unconditionally stable with respect to the size of the computational cells, allowing the computational mesh to be refined locally to an arbitrary degree without altering solution stability. The practical implementation of the method is detailed for the shallow water equations with topographical source term. Computational examples report applications of the method to the linear advection, Burgers and shallow water equations. In the case of sharp bottom discontinuities, the need for improved, well-balanced discretizations of the geometric source term is acknowledged.

1 Introduction

Explicit methods for hyperbolic systems are subjected to the so-called Courant-Friedrichs-Lewy (CFL) stability constraint $\nu = \lambda_{\max}\Delta t/\Delta x \leq 1$ [6], where $\lambda_{\max}$ is the maximum of the absolute values of the wave propagation speeds of the hyperbolic system and Δx is the width of the computational cell. This amounts to saying that the computational time step must be such that the size of the domain of dependence of the solution be smaller than the size of each computational cell

*Corresponding author: antoine.rousseau@inria.fr

in the domain. This induces a number of problems in practical applications, such as engineering studies, where computational times are seen as an important (if not the most important) criterion for mesh design, sometimes at the expense of computational accuracy. Two typical examples where the CFL constraint hinders the quality of the computational solution can be mentioned. The first is that of strongly contrasted wave propagation speeds in the hyperbolic system, e.g. when the governing equations for compressible fluid or free surface flows are coupled with those of advection-dominated transport. In such systems, the transport velocity is often an order of magnitude smaller than the pressure wave speed. The computational time step being limited by the propagation speed of the pressure waves, the CFL for the transport processes is very small, which induces a strong numerical diffusion in the transport solution. The second example is that of meshes requiring a local refinement in order to accommodate for strongly varying, local geometrical features (*e.g.* large bathymetry variations in the shallow water equations). While such cells yield a strong restriction on the computational time step, it may be important to minimize numerical diffusion (and therefore achieve CFL values as close to unity as possible) in other parts of the model rather than in this limited number of small cells. In both situations, allowing for CFL values larger than unity (albeit in restricted regions of the model) is desirable. Besides local time stepping and asymptotic formulations (see e.g. [3]) that will not be considered here, approaches using uniform time steps have been proposed in the past to overcome the CFL limitation.

A first option consists in switching from space- to time-averaging [5, 11]. This makes the scheme implicit, with the drawback that large systems must be solved. For some particular configurations (subcritical/subsonic flows), the system is triangular and becomes very easy to solve [12]. However, it is not clear how the approach could be extended to the solution of multidimensional problems over unstructured grids.

A second option is to preserve the explicit character of the scheme by allowing the domain of dependence to extend over more than one cell [20, 21, 22, 13, 17]. When linear systems of conservation laws are dealt with, the computation of the domain of dependence and the averaging are straightforward [18, 19]. For non-linear systems, however, a simple averaging may not suffice because of the multiple, non-linear interactions that may arise from the waves within the domain of dependence [22, 31] and because of the possible violations of the entropy conditions [23]. The method described in [22, 31] is time-consuming in that the computation of the domain of dependence is time-dependent and it may be necessary to account for up to three wave interactions within a given time step to achieve CFL values of 3 to 4 [31]. Moreover, even for linear systems, the method is very sensitive to the accuracy with which the domain of dependence is computed, a consideration that holds for most schemes for hyperbolic systems [32]. This means that the domain of dependence must be recomputed at every time step for each computational interface, a task that contributes to slow down the algorithm. The difficulty may be overcome to some extent by tracking the domain of dependence in a forward manner rather than backward in time [25, 26, 27], with reported CFL values up to 8 for shallow water simulations. In [38], CFL up to 25 are achieved. Second-order extensions of the method are available [24]. In [28], the large time step approach is formulated in a different way, by increasing the domain of dependence of the solution. The flux at a given interface is computed as a linear combination of the fluxes at the cell interfaces lying within the domain of dependence. The formulation is applied to Roe's and HLL flux formulae in [29, 30].

A third approach, that can be seen as a generalization of the second, uses filters to derive unconditionally stable explicit methods. For instance, [33] uses predefined convolution kernels that may be made independent from the cell width, thus simplifying the averaging procedure.

In [1], damping filters are used to enforce stability. Note that [28, 29, 30] can also be seen as resulting from a weighted average (thus a filtering) of the fluxes. The present work follows this approach, with the difference that the filtering is applied prior to the computation of the fluxes and source terms. The method is called Large CFL (LCFL) hereafter to avoid any confusions with the well-known Large Time Step (LTS) approach.

This paper is organised as follows. In Section 2, the approach is described for general hyperbolic systems. Subsection 2.1 is devoted to the broad lines of the proposed LCFL approach. The key contribution of the present work is presented in Subsection 2.2. Section 3 details the application to the one-dimensional shallow water equations, that incorporate a geometric source term in addition to the hyperbolic terms. Section 4 is dedicated to a stability analysis of the proposed method. In Section 5 we present numerical applications to the linear advection, Burgers and shallow water equations. Section 6 is devoted to a discussion and conclusions. Additional information on the performance of various convolution kernels can be found in Appendix A.

2 Numerical scheme : hyperbolic part

2.1 Algorithm overview

The purpose is to solve the conservation law

$$\partial_t \boldsymbol{u} + \partial_x \mathbf{f} = 0 \tag{1}$$

using a finite volume approach. Eq. (1) is discretized into

$$\mathbf{u}_i^{n+1} = \mathbf{u}_i^n + \frac{\Delta t}{\Delta x}\left(\mathbf{f}_{i-1/2}^{n+1/2} - \mathbf{f}_{i+1/2}^{n+1/2}\right) \tag{2}$$

where $\mathbf{u}_i^n$ and $\mathbf{f}_{i-1/2}^{n+1/2}$ are respectively the average of $\mathbf{u}$ over the ith computational cell at the nth time level and the flux at the interface $i-1/2$ between cells $i-1$ and i. $\mathbf{f}_{i-1/2}^{n+1/2}$ is computed by solving a Riemann problem with initial state

$$\mathbf{u}\left(x, t^n\right) = \begin{cases} \mathbf{u}_{i-1/2,L} & \text{if} \quad x < x_{i-1/2} \\ \mathbf{u}_{i-1/2,R} & \text{if} \quad x > x_{i-1/2} \end{cases} \tag{3}$$

The key feature of the algorithm lies in the computation of the left and right states of the Riemann problem. In the original Godunov scheme [10], the left and right states are taken equal to the average values $\mathbf{u}_{i-1}^n$ and $\mathbf{u}_i^n$ respectively. In higher-order schemes such as the MUSCL [16] or the PPM [4], the profile $\mathbf{u}\left(x, t^n\right)$ is reconstructed using piecewise linear or piecewise parabolic functions. The left and right states are obtained from suitable averages of these reconstructed profiles. In the PPM [4] and MUSCL-EVR techniques [9], the left and right states are obtained by averaging the reconstructed variables in the basis of eigenvectors of the Jacobian matrix $\mathbf{A} = \partial \mathbf{f} / \partial \mathbf{u}$. A variation using the averaged Riemann invariants is also available [34]. The EVR approach is used in what follows.

The conserved variable $\mathbf{u}$ is rewritten as

$$\mathbf{u} = \mathbf{K}\boldsymbol{\alpha} = \sum_p \alpha_p \mathbf{K}_p \tag{4}$$

where $\mathbf{K}$ is the matrix of eigenvectors of $\mathbf{A}$, $\mathbf{K}_p$ and α_p are respectively the pth eigenvector of $\mathbf{A}$ and wave strength. The algorithm is first explained for the determination of the left state of the Riemann problem.

(1) A domain of size D is defined *a priori* for the averaging. For the left state of the Riemann problem, this averaging domain extends on the left-hand side of the interface. In usual explicit schemes, the size of this domain is the cell width Δx, while the proposed LCFL algorithm allows for wider domains.

(2) The average of $\mathbf{u}(x, t^n)$ over D is used to compute the averaged matrix of eigenvectors and the wave propagation speeds:

$$\overline{\mathbf{u}_{i-1/2,L}} = \frac{1}{D} \int_{x_{i-1/2}-D}^{x_{i-1/2}} \mathbf{u}\left(x, t^n\right) \mathrm{d}x \tag{5a}$$

$$\overline{\mathbf{K}_{L,p}} = \mathbf{K}_p \left(\overline{\mathbf{u}_{i-1/2,L}}\right) \tag{5b}$$

$$\lambda_{L,p} = \lambda_p(\overline{\mathbf{u}_{i-1/2,L}}) \tag{5c}$$

(3) Each wave strength α_p is averaged over its domain of dependence (Figure 1), the width of which is denoted by $D_{L,p}$

$$D_{L,p} = \lambda_{L,p} \Delta t \tag{6a}$$

$$\overline{\alpha_{L,p}} = \begin{cases} \frac{1}{D_{L,p}} \int_{x_{i-1/2}-D_{L,p}}^{x_{i-1/2}} \alpha_p\left(x, t^n\right) \mathrm{d}x & \text{if} \quad D_{L,p} > 0 \\ \alpha_p \left(\overline{\mathbf{u}_{i-1/2,L}}\right) & \text{if} \quad D_{L,p} \leq 0 \end{cases} \tag{6b}$$

Note that the CFL stability constraint requires that all the $D_{L,p}$ be smaller than D.

(4) The left state of the Riemann problem is obtained as

$$\mathbf{u}_{i+1/2,L} = \sum_p \overline{\mathbf{K}_{,L,p}} \overline{\alpha_{L,p}} \tag{7}$$

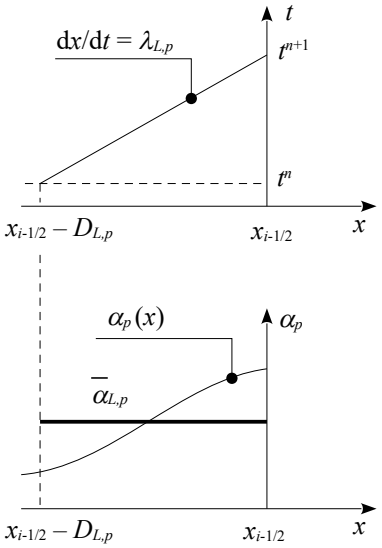

Figure 1: EVR scheme. Definition sketch for the averaging of the wave strengths on the left-hand side of the interface. Top: propagation of the pth wave strength in the (x,t) plane. Bottom: averaging of the wave strength over its domain of dependence.

The right state of the Riemann problem is defined by repeating Steps (1)-(4) above for an averaging domain D extending on the right-hand side of the interface:

$$\mathbf{u}_{i+1/2,R} = \sum_p \overline{\mathbf{K}_{,R,p}}\,\overline{\alpha_{R,p}} \tag{8}$$

with

$$\overline{\mathbf{K}_{R,p}} = \mathbf{K}_p\left(\overline{\mathbf{u}_{i-1/2,R}}\right) \tag{9a}$$

$$\overline{\mathbf{u}_{i-1/2,R}} = \frac{1}{D}\int_{x_{i-1/2}-D}^{x_{i-1/2}} \mathbf{u}\left(x,t^n\right)\,\mathrm{d}x \tag{9b}$$

$$\overline{\alpha_{R,p}} = \begin{cases} \frac{1}{D_{R,p}}\int_{x_{i-1/2}-D_{R,p}}^{x_{i-1/2}} \alpha_p\left(x,t^n\right)\,\mathrm{d}x & \text{if} \quad D_{R,p} < 0 \\ \alpha_p\left(\overline{\mathbf{u}_{i-1/2,R}}\right) & \text{if} \quad D_{R,p} \geq 0 \end{cases} \tag{9c}$$

$$D_{R,p} = \lambda_{R,p}\Delta t \tag{9d}$$

$$\lambda_{R,p} = \lambda_p\left(\overline{\mathbf{u}_{i-1/2,R}}\right) \tag{9e}$$

Remark 1. *In this case, the domain of dependence of the solution extends on the right-hand side of the interface, consequently* $D < 0$.

In the case of a subcritical/subsonic 2×2 system (i.e. with $\lambda_1 < 0$ and $\lambda_2 > 0$), the above formulae can be simplfied into [9]

$$\overline{\alpha_{L,p}} = \frac{1}{D_2}\int_{x_{i-1/2}-D_2}^{x_{i-1/2}} \alpha_p\left(x,t^n\right)\,\mathrm{d}x \quad p = 1,2 \tag{10a}$$

$$\overline{\alpha_{R,p}} = \frac{1}{D_1} \int_{x_{i-1/2}-D_1}^{x_{i-1/2}} \alpha_p\left(x, t^n\right) \, \mathrm{d}x \quad p = 1, 2 \tag{10b}$$

$$D_1 = \lambda_1 \left(\overline{\mathbf{u}_{i-1/2,R}}\right) \, \Delta t, \quad D_2 = \lambda_2 \left(\overline{\mathbf{u}_{i-1/2,L}}\right) \, \Delta t \tag{10c}$$

This is the version used in the computations hereafter.

Remark 2. *In Section 4 , the CFL constraint is shown to be*

$$\nu' = \lambda \frac{\Delta t}{D} \leq \nu'_{max} \tag{11}$$

where ν'_{max} depends only on the convolution kernel. Using the definition of the CFL number $\nu = \lambda \Delta t / \Delta x$, the above condition is equivalent to

$$\nu \leq \frac{D}{\Delta x} \nu'_{max} \tag{12}$$

Equation (11) *shows that the maximum permissible time step does not depend on the width Δx of the computational cell but on the size D of the convolution domain. Consequently, arbitrarily large CFL values may be achieved depending on the ratio $D/\Delta x$, as shown by Eq.* (12)*. This is why the term Large CFL (LCFL) is used instead of Large Time Step (LTS) for the present method.*

ARu: OK comme cela

2.2 Equivalent Riemann problem definition for large CFL

The proposed approach is based on the following considerations.

(a) Iterations should be avoided. In particular, no iterative update of the domain of dependence of the various waves is performed. This is obviously a strong limitation in the case of strongly varying wave speeds with respect to both time and space. It is expected however that in the case of slow transients and subcritical flows, such a non-iterative method should give reasonably accurate results.

(b) The computation of the averages $\overline{\alpha_{S,p}} (S = L, R)$ in Eq. (7) should yield as smooth and non-oscillatory solutions as possible, even if the size of the domain of dependence of $\alpha_{S,p}$ has not been determined with optimal accuracy. This requirement is motivated by the following considerations. While computing the size of the domain of dependence is very easy in the case of a linear system, this procedure becomes significantly more time consuming when the system is non-linear. Moreover, assuming that the present work is to be extended to two-dimensional systems, determining the exact size of the domain of dependence of a non linear system in two dimensions of space may turn out to be intractable.

In the present work, the averaging is performed using a convolution kernel

$$\overline{\alpha_{L,p}} = w_L * \alpha \tag{13a}$$

$$\overline{\alpha_{R,p}} = w_R * \alpha \tag{13b}$$

$$w_R(x) = w_L(-x) = \left\{ \begin{array}{ccc} 0 & \text{if} & x < 0 \\ f_w(x) & \text{if} & x \geq 0 \end{array} \right. \tag{13c}$$

$$f_w(x) \geq 0 \quad \forall x \geq 0 \tag{13d}$$

$$\int_0^D f_w(x) \mathrm{d}x = 1 \tag{13e}$$

where the constraints (13d) and (13e) are derived respectively from positiveness and consistency considerations. Note that consistency is to be understood in the limit of both both D and Δx tending to zero.

An obvious solution consists in giving the same weights to all computational cells

$$f_w(x) = \left\{ \begin{array}{ccc} \frac{1}{D} & \text{if} & x \in [0, D] \\ 0 & & \text{otherwise} \end{array} \right. \tag{14}$$

However this kernel yields strongly oscillatory solutions (see Appendix A). The reason is that this kernel gives the same weight to all points of the support of f_w, making the average very sensitive to the estimate of D. In the light of Consideration (b) above, this issue is eliminated by using decreasing functions f_w giving a lighter weight to large distances than to small ones. In what follows, a power law is used:

$$f_w(x) = \left\{ \begin{array}{ccc} \dfrac{b+1}{D}\left(1 - \dfrac{x}{D}\right)^b & \text{if} & x \in [0, D] \\ 0 & & \text{otherwise} \end{array} \right. \tag{15}$$

with $b = 1.5$. For the sake of computational rapidity, the convolution is not computed exactly. For Interface $i + 1/2$, the following numerical approximations are used

$$\overline{\alpha_{L,p}} \approx \frac{\sum_{k=1}^{K} \alpha_{i-k+1} \Delta x_{i-k+1} f_w(x_{i+1/2} - x_{i-k+1})}{\sum_{k=1}^{K} \Delta x_{i-k+1} f_w(x_{i+1/2} - x_{i-k+1})} \tag{16a}$$

$$\overline{\alpha_{R,p}} \approx \frac{\sum_{k=1}^{K} \alpha_{i+k} \Delta x_{i+k} f_w(x_{i+1/2} - x_{i+k})}{\sum_{k=1}^{K} \Delta x_{i+k} f_w(x_{i+1/2} - x_{i+k})} \tag{16b}$$

where K is the number of cells spanned by the kernel and Δx_i is the width of the ith computational cell. Alternative kernels are presented in Appendix A.

3 Application to the shallow water equations

3.1 Governing equations

The scheme is applied to the one-dimensional shallow water equations

$$\partial_t \mathbf{u} + \partial_x \mathbf{f} = \mathbf{s} \tag{17a}$$

$$\mathbf{u} = [h, q]^T = [h, hu]^T \tag{17b}$$

$$\mathbf{f} = [q, M]^T = \left[q, \frac{q^2}{h} + \frac{g}{2}h^2\right]^T \tag{17c}$$

$$\mathbf{s} = [0, gh(S_0 - S_f)]^T \tag{17d}$$

where g is the gravitational acceleration, h is the water depth, M is the specific force per unit width, q is the unit discharges, $S_0 = -\partial_x z_b$ and S_f are respectively the bottom and friction slopes, u is the flow velocity, and z_b is the bottom elevation. The eigenvalues of the system are

$$\lambda^{(p)} = u + (-1)^p c, \quad c = (gh)^{1/2}, \quad p = 1, 2 \tag{18}$$

Note that equation (17a) can also be rewritten using the following variable change for an easier discretization (see next subsection):

$$\partial_t \mathbf{u}' + \partial_x \mathbf{f} = \mathbf{s} \tag{19a}$$

$$\mathbf{u}' = [z, q]^T = [h + z_b, hu]^T \tag{19b}$$

The application of the EVR procedure to this system of equations gives [9]

$$\mathbf{K} = \begin{bmatrix} 1 & 1 \\ u - c & u + c \end{bmatrix} \tag{20a}$$

$$\alpha_1 = \alpha_2 = \frac{h}{2} \tag{20b}$$

Consequently, reconstructing the water depth (or free surface elevation) alone is sufficient to reconstruct $[h, q]^T$.

The salient difference between this system and the conservation form (1) presented in Section 2 is the presence of the source term $\mathbf{s}$. Many approaches for source term treatment are available [15]. The source term splitting approach is retained [2, 8]. How the source term is to be accommodated in the LCFL discretization depends on the flux formulae. Therefore, the treatment of the source term is Riemann solver-dependent so far. The discretization (2) is revised into (see Appendix B)

$$\mathbf{u}_i^{n+1} = \mathbf{u}_i^n + \frac{\Delta t}{\Delta x}\left(\mathbf{f}_{i-1/2}^{n+1/2} - \mathbf{f}_{i+1/2}^{n+1/2} + \mathbf{s}_{i-1/2,R}^{n+1/2} + \mathbf{s}_{i+1/2,L}^{n+1/2}\right) \tag{21a}$$

$$\mathbf{s}_{i-1/2,L}^{n+1/2} = \frac{\lambda_{i-1/2}^+}{\lambda_{i-1/2}^+ - \lambda_{i-1/2}^-}\mathbf{s}_{i+1/2}^{n+1/2} \tag{21b}$$

$$\mathbf{s}_{i-1/2,R}^{n+1/2} = \frac{-\lambda_{i-1/2}^-}{\lambda_{i-1/2}^+ - \lambda_{i-1/2}^-}\mathbf{s}_{i+1/2}^{n+1/2} \tag{21c}$$

$$\mathbf{s}_{i+1/2}^{n+1/2} = \begin{cases} -\frac{g}{2}\left((z_L - z_{b,L})^2 - (z_L - z_{b,R})^2\right) & \text{if} \quad z_{b,L} < z_{b,R} \\ \frac{g}{2}\left((z_R - z_{b,R})^2 - (z_R - z_{b,L})^2\right) & \text{if} \quad z_{b,L} > z_{b,R} \end{cases} \tag{21d}$$

3.2 Riemann solver

The Riemann solver is the HLL [14] / HLLC [37, 36] solver with Davis' [7] wave speed estimates. The flux formulae are

$$\mathbf{f}_{i+1/2} = \frac{\lambda_{i+1/2}^+ \mathbf{f}_{i+1/2,L} - \lambda_{i+1/2}^- \mathbf{f}_{i+1/2,R}}{\lambda_{i+1/2}^+ - \lambda_{i+1/2}^-} - \frac{\lambda_{i+1/2}^- \lambda_{i+1/2}^+}{\lambda_{i+1/2}^+ - \lambda_{i+1/2}^-}(\mathbf{u'}_{i+1/2,L} - \mathbf{u'}_{i+1/2,R}) \tag{22a}$$

$$\lambda^-_{i+1/2} = \min\left(0, \lambda^{(1)}_i, \lambda^{(1)}_{i+1}\right) \tag{22b}$$

$$\lambda^+_{i+1/2} = \max\left(0, \lambda^{(3)}_i, \lambda^{(3)}_{i+1}\right) \tag{22c}$$

In the particular case of water at rest

$$[z, q]^T(x, 0) = [z_0, 0]^T \ \ \forall x \tag{23}$$

one has

$$\lambda^+_{i+1/2} = -\lambda^-_{i+1/2} = \max(c_i, c_{i+1}) \ \ \forall i \tag{24a}$$

$$q_{i+1/2} = 0 \ \ \forall i \tag{24b}$$

$$M_{i+1/2} = \frac{g}{4}\left(h_i^2 + h_{i+1}^2\right) \ \ \forall i \tag{24c}$$

3.3 Treatment of the geometric source term

In order to preserve the static conditions (23), the convolution kernel is applied to the free surface elevation instead of the water depth:

$$z_{i+1/2,L} = w_L * z(x, t_n) \tag{25a}$$

$$z_{i+1/2,R} = w_R * z(x, t_n) \tag{25b}$$

The necessary conditions for well-balancing are examined hereafter. Consider the static conditions (23) and three consecutive cells $(i-1, i, i+1)$ with different bottom levels $(z_{b,i-1}, z_{b,i}, z_{b,i+1})$. A necessary well-balancing condition is that static conditions be preserved, that is

$$\mathbf{f}_{i-1/2} - \mathbf{f}_{i+1/2} + \mathbf{s}_{i-1/2,R} + \mathbf{s}_{i+1/2,L} = 0 \tag{26}$$

Consider first the bottom configuration $z_{b,i-1} \leq z_{b,i} \leq z_{b,i+1}$. In the particular case (23), the momentum equation becomes

$$\frac{g}{4}\left(h^2_{i-1/2,L} + h^2_{1-1/2,R}\right) - \frac{g}{4}\left(h^2_{i+1/2,L} + h^2_{i+1/2,R}\right) - \frac{g}{4}(h^2_{i-1/2,L} - h^2_{i-1/2,R}) - \frac{g}{4}(h^2_{i+1/2,L} - h^2_{i+1/2,R}) = 0 \tag{27}$$

which is equivalent to

$$(h^2_{i-1/2,R}) - (h^2_{i+1/2,L}) = 0 \tag{28}$$

Moreover, from the initial conditions

$$h_{i-1/2,R} = z_{i-1/2,R} - z_{b,i-1/2,R} = z_0 - z_{b,i-1/2,R} \tag{29a}$$

$$h_{i+1/2,L} = z_{i+1/2,L} - z_{b,i+1/2,R} = z_0 - z_{b,i+1/2,L} \tag{29b}$$

Substituting the above two relationships into (28) yields

$$z_{b,i+1/2,L} = z_{b,i-1/2,R} = z_{b,i} \tag{30}$$

In other words, the bottom level of Cell i should be used in the characterization of the Riemann problem for the flux and source term computation at both interfaces $i-1/2$ and $i+1/2$. It is easy

to check that this formula allows static conditions to be preserved for all other possible bottom configurations.

In addition, a momentum balance fix is used in the cells where the CFL exceeds a user-defined threshold (usually set around 10). The purpose of this fix is to remove spurious oscillations in the cell-averaged unit discharge profiles in the presence of steep topographical gradients combined with large CFL. Indeed, while Eq. (21d) preserves static conditions, it yields artificial oscillations in the cell-averaged unit discharge for steady state configuration involving a non-zero discharge. The reason for this is detailed in Appendix B. Such oscillations do not indicate a violation of the mass conservation principle, but only an inaccurate momentum balance due to the inaccuracy of the source term discretization. The purpose of the fix, based on a minmax procedure with the unit discharges in the neighbouring cells, is simply to adjust the local estimate of the topographic source term so as to avoid artificial oscillations.

4 Linear stability analysis

The stability analysis is carried out for the 1D linear advection equation

$$\partial_t u + \partial_x(\lambda u) = 0 \tag{31}$$

that can be extended to linear hyperbolic systems using the Riemann invariants. The purpose is to provide an analysis of the convolution kernel as applied to the first-order upwind discretization of Eq. (31):

$$u_i^{n+1} = u_i^n + \nu\left(\tilde{u}_{i-1/2,L}^n - \tilde{u}_{i+1/2,L}^n\right) \tag{32a}$$

$$\nu = \lambda\frac{\Delta t}{\Delta x} \tag{32b}$$

$$\tilde{u}_{i+1/2,L}^n = w_L * u(x,t_n) \approx \sum_{k=1}^{K} f_k u_{i-k+1}^n \Delta x \tag{32c}$$

$$\sum_{k=1}^{K} f_k \Delta x = 1, \quad \Delta x = \frac{D}{K} \tag{32d}$$

where the $f_k = f(x_k)$ are the coefficients of the convolution kernel at abscissae $k\Delta x$, and $*$ denotes the convolution product.

4.1 General framework

Following Von Neuman's stability analysis, the solution is sought in the form

$$u_i^n = u_0^0 \exp(\mu t + j\sigma i\Delta x) \quad , \quad j^2 = -1 \tag{33}$$

where $\sigma\Delta x = 2\pi/M \ \in [0,\pi]$ (M is the wave number, ranging from 2 to infinity). The numerical amplification factor is given by

$$\begin{aligned} A_K &\equiv \frac{u_i^{n+1}}{u_i^n} \\ &= 1+\nu\left(1-e^{j\sigma\Delta x}\right)\sum_{k=1}^{K} f_k e^{-jk\sigma\Delta x}\Delta x \\ &= 1+\nu' K\left(1-e^{j\sigma\Delta x}\right)\sum_{k=1}^{K} f_k e^{-jk\sigma\Delta x}\Delta x \\ &= 1+\nu' D\sum_{k=1}^{K} f_k\left(e^{-jk\sigma\Delta x}-e^{-j(k-1)\sigma\Delta x}\right) \end{aligned} \tag{34a}$$

$$\nu' \equiv \frac{\nu}{K} = \frac{\lambda\Delta t}{K\Delta x} = \frac{\lambda\Delta t}{D} \tag{34b}$$

where $j^2 = -1$. The writing with ν' is more convenient than that with ν for furter analysis because ν' is independent of K. This makes it possible to consider the limit case $\Delta x \to 0$, that yields infinite K and ν while leaving ν' unchanged.

Two amplification factors are defined.

Definition 1 (Numerical amplification factor)**.** *For any given $D > 0$, Δx, $\sigma \in (0, \pi/\Delta x]$, $\nu' > 0$ and any positive kernel function f with support $[0, D]$, we rewrite the numerical amplification factor* (34a) *as:*

$$A_K(\sigma) = 1+\nu'\big(\alpha_K(\sigma)+j\beta_K(\sigma)\big), \tag{35}$$

$$\alpha_K(\sigma) = D\sum_{k=1}^{K} f_k\big(\cos(\sigma x_k)-\cos(\sigma x_{k-1})\big), \tag{36}$$

$$\beta_K(\sigma) = -D\sum_{k=1}^{K} f_k\big(\sin(\sigma x_k)-\sin(\sigma x_{k-1})\big), \tag{37}$$

$$\tag{38}$$

where we recall that j is the unit imaginary number.

Definition 2 (Asymptotic amplification factor)**.** *For any given $D > 0$, $\sigma > 0$, $\nu' > 0$ and any positive kernel function f with support $[0, D]$, we define A_∞ as the limit expression for A_K as K tends to infinity:*

$$A_\infty(\sigma) = 1+\nu'\big(\alpha(\sigma)+j\beta(\sigma)\big), \tag{39}$$

$$\alpha(\sigma) = -D\sigma\int_0^D f(x)\sin(\sigma x)\,dx, \tag{40}$$

$$\beta(\sigma) = -D\sigma\int_0^D f(x)\cos(\sigma x)\,dx. \tag{41}$$

Figure 2 provides a graphical representation of A_K and A_∞ in the complex plane.

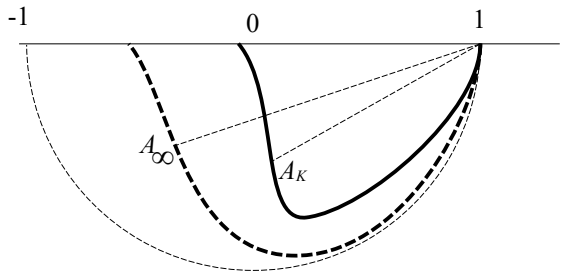

Figure 2: Representation of A_K and A_∞ in the complex plane. Bold solid line: set of the possible locations for A_k for all $\sigma\Delta x$. Bold dashed line : set of locations for A_∞ for all σ. Thin, dashed line: unit circle. A_K and A_∞ are drawn for the same σ.

The following practical results are derived in the sequel of the present section:

(S1) There exists a value for ν' such that the numerical amplification factor A_K lies within the unit circle for all $\sigma\ \Delta x \leq \pi$, thus ensuring stability of the numerical solution.

(S2) Let A_∞ be the limit location of A_K in the complex plane for $K \to \infty$. There also exists a positive ν' such that A_∞ lies within the unit circle.

Note that since f is a kernel function, f is non-negative, $\int_0^D f = 1$ and

$$I_K = \sum_{k=1}^{K} x_k f_k\ \Delta x > 0, \tag{42a}$$

$$I = \int_0^D x f(x)\, dx > 0 \tag{42b}$$

In order to analyse the stability of the upwind scheme (32a), we first prove an asymptotic result for $A_\infty(\sigma)$ (see Theorem 1), before studying the numerical amplification factor $A_K(\sigma)$ (see Theorem 2).
In all lemmas hereafter, we will use the following equalities:

$$|A_\infty(\sigma)|^2 = 1 + \nu'\Big(2\alpha(\sigma) + \nu'\big(\alpha^2(\sigma) + \beta^2(\sigma)\big)\Big) \tag{43a}$$

$$|A_K(\sigma)|^2 = 1 + \nu'\Big(2\alpha_K(\sigma) + \nu'\big(\alpha_K^2(\sigma) + \beta_K^2(\sigma)\big)\Big) \tag{43b}$$

4.2 Asymptotic Stability

We now prove the following three lemmas.

Lemma 1. *Let $\nu_1' = I/(5D)$, there exists $\sigma_1^* > 0$ such that:*

$$\forall(\nu', \sigma) \in (0, \nu_1') \times (0, \sigma_1^*), |A_\infty(\sigma)| \leq 1$$

Proof. Elementary computations yield

$$\alpha(0) = 0, \quad \alpha'(0) = 0, \quad \alpha''(0) = -2ID \tag{44a}$$

$$\beta(0) = 0, \quad \beta'(0) = -D \tag{44b}$$

Consequently, there exists $\sigma^* > 0$ such that, for every $0 < \sigma < \sigma^*$:

$$-3ID\sigma^2/2 < \alpha(\sigma) < -ID\sigma^2/2, \tag{45a}$$

$$-2D\sigma < \beta(\sigma) < -D\sigma/2. \tag{45b}$$

Define $\sigma_1^* = \min(\sigma^*, 2/(3I))$ and choose $\nu' > 0$ such that $\nu' < \nu_1' = I/(5D)$. For $\sigma < \sigma_1^*$ and $\nu' < I/(5D)$ one has

$$\begin{array}{rcl} 2\alpha + \nu'(\alpha^2 + \beta^2) & < & -\sigma^2 \left(ID - \nu' D^2 (\frac{9I^2}{4}\sigma^2 + 4) \right) \\ & < & -\sigma^2 D(I - 5\nu' D) \\ & < & 0 \end{array} \tag{46}$$

Using Equation (43a) concludes the proof. □

Lemma 2. *There exists ν_2'>0 such that:*

$$\forall(\nu', \sigma) \in (0, \nu_2') \times [\sigma_1^*, 2\pi/D], |A_\infty(\sigma)| \leq 1$$

Proof. Let $\Phi \colon [\sigma_1^*; \dfrac{2\pi}{D}] \longrightarrow \mathbf{R}$.

$$\sigma \longmapsto \frac{-2\alpha(\sigma)}{\alpha^2(\sigma) + \beta^2(\sigma)}$$

The function Φ is positive and continuous on a compact set, so it is bounded. In particular there exists $\Phi_{\min} > 0$ such that:

$$\forall \sigma \in [\sigma_1^*; \frac{2\pi}{D}], \Phi(\sigma) \geq \Phi_{\min} \tag{47}$$

Let us now set $\nu_2' = \Phi_{\min} > 0$, then for every $\sigma \in [\sigma_1^*; \frac{2\pi}{D}]$, equations (47) and (43a) lead to $|A_\infty(\sigma)| \leq 1$. □

Lemma 3. *There exists ν_3'>0 such that:*

$$\forall(\nu', \sigma) \in (0, \nu_3') \times [2\pi/D, +\infty), \quad |A_\infty(\sigma)| \leq 1$$

Proof. The result comes directly from (43a), together with trigonometry considerations and relying on the fact that f is a positive, continuous, non constant, decreasing function. □

Finally, we are now able to state the following asymptotic stability theorem:

Theorem 1. *For every $D > 0$ and every decreasing kernel function f over $[0, D]$,*

$$\exists \nu^* > 0 : \forall(\nu', \sigma) \in (0, \nu^*) \times [0, +\infty), |A_\infty(\sigma)| \leq 1$$

Proof. If f is constant, then $A_\infty(\sigma)$ can be computed explicitly and $\nu^* = 1$ is sufficient.
If f is not constant, the proof relies on Lemmas 1, 2 and 3, choosing

$$\nu^* = \min(\nu'_1, \nu'_2, \nu'_3) > 0$$

and recalling that $A_\infty(0) = 1$ for every $\nu' > 0$. □

4.3 Discrete stability

The scheme behaviour is investigated for a vanishing $\Delta x = D/K$ and a fixed time step. We recall that A_K can be written as:

$$A_K(\sigma) = 1 + \nu'\big(\alpha_K(\sigma) + j\beta_K(\sigma)\big)$$

We have the following lemmas for $A_K(\sigma)$:

Lemma 4. *Let $\nu'_4 = I_K/(5D)$, there exists $\sigma_1^* > 0$ such that:*

$$\forall(\nu'_K, \sigma) \in (0, \nu'_4) \times (0, \sigma_1^*), \quad |A_K(\sigma)| \leq 1$$

Lemma 5. *There exists $\nu'_5 > 0$ such that:*

$$\forall(\nu'_K, \sigma) \in (0, \nu'_5) \times, [\sigma^*, \pi/\Delta x], \quad |A_K(\sigma)| \leq 1$$

The proofs for lemmas 4-5 are similar to the ones of Lemmas 1-2 and will not be repeated here. We can now state the following stability theorem:

Theorem 2. *For every $D > 0$, every $K \in \mathrm{N}^*$ and every decreasing kernel function f over $[0, D]$,*

$$\exists\ \nu_K^* > 0 : \forall(\nu, \sigma) \in (0, \nu_K^*) \times [0, \pi/\Delta x], |A_K(\sigma)| \leq 1$$

Proof. If f is constant, then $A_K(\sigma)$ can be computed explicitly and $\nu^* = 1$ is sufficient.
If f is not constant, the proof relies on Lemmas 4 and 5, choosing

$$\nu_K^* = \min(\nu'_4, \nu'_5) > 0$$

and recalling that $A_K(0) = 1$ for every $\nu'_K > 0$. □

4.4 Practical computation of the stability constraint

In practice, ν_K^* is needed to compute the maximum permissible time step $\Delta t_{\max}$ as

$$\Delta t_{\max} = \min_i \left(\nu_K^* \frac{\Delta x_i}{\lambda_i} \right) \tag{48}$$

Let $\nu_K^{'\max}$ be the maximum ν' for which the numerical amplification factor $A_K = 1 + \nu'(\alpha_K + j\beta_K)$ remains within the unit circle. Let A, B, B' and C be respectively the points $z = 1$, $z = 1 + \alpha_K + j\beta_K$, $z = 1 + \nu'(\alpha_K + j\beta_K) = A_K$ and the intersection of (AB) with the unit circle in the complex plane (Figure 3). The numerical solution is stable provided that B' remains within the unit circle for

all σ. In other words, AB' $\leq$ AC is a necessary and sufficient condition for linear stability. Since $\mathrm{AB}' = \nu' \mathrm{AB}$, stability is equivalent to

$$\nu' \leq \frac{\mathrm{AC}}{\mathrm{AB}} = \frac{-2\alpha_K(\sigma)}{\alpha_K^2(\sigma) + \beta_K^2(\sigma)} \quad \forall\, \sigma \tag{49}$$

Consequently

$$\nu_K^{'\max} = \min_{\sigma} \frac{-2\alpha_K(\sigma)}{\alpha_K^2(\sigma) + \beta_K^2(\sigma)} \tag{50}$$

From a practical point of view, Equation (50) is applied for a large number (typically 100) of σ within $[0, \pi/\Delta x]$ and the minimum value is retained.

Remark 3. *It is noted that the constraint* (50) *depends on* K *and is not mesh-independent. However, numerical experiments seem to indicate that, for convolution kernels given by decreasing functions over* $[0, D]$ *such as those presented in Appendix A,*

$$\rho_K = \left| \frac{A_K - 1}{A_\infty - 1} \right| \leq 1, \quad \forall K \in \mathrm{N}^* \tag{51}$$

which implies that $\nu_K^* > \nu^*$ *for every* $K \in \mathrm{N}^*$. *The consequence would be a mesh-independent stability condition, obtained from the limit case* $K \to \infty$. *Whether Property* (51) *actually holds is left for further research.*

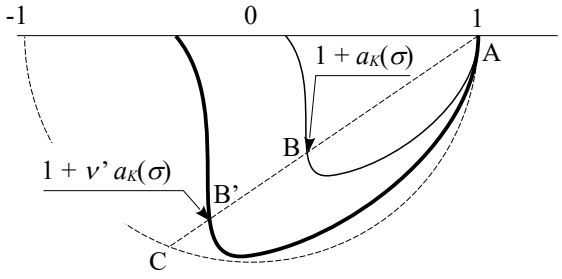

Figure 3: Computation of the maximum permissible ν'. Definition sketch in the complex plane, with $a_K(\sigma) = \alpha_K(\sigma) + j\beta_K(\sigma)$.

5 Computational examples

5.1 Linear advection

The method is first tested for the linear advection equation

$$\partial_t u + \partial_x(cu) = 0 \tag{52}$$

where c is the advection velocity and u is a scalar. The initial condition is

$$u(x,0) = \begin{cases} u_0 & \text{for } x \in [x_1, x_2] \\ 0 & \text{otherwise} \end{cases} \tag{53}$$

The computational grid is irregular. All cells have a width Δx_0, except for cell 101, with a width Δx_1 significantly smaller than the others. The numerical values of the parameters are given in Table 1.

parameter	meaning	value
c	advection velocity	1 m s^{-1}
u_0	step height	0.5 m
x_1	LH abscissa of the step	10 m
x_2	RH abscissa of the step	60 m
Δx_0	default cell width	1 m
Δx_1	width of Cell 101	10^{-2} m

Table 1: Linear advection equation. Test parameters.

The equation is solved using two methods, with two different time steps (see Table 2). The first is the classical explicit Godunov method, that is subjected to the unit CFL stability constraint (here, $\Delta t = 10^{-2}$ s). The second is the proposed LCFL method, with a time step such that the CFL in the coarse cells in unity (therefore, the CFL in the narrow cell is 100). The convolution kernel is given by (15), with $D = 2$ m and $b = 1.5$. The CPU times for the two methods are shown in Table 2.

Method	Δt	Max. CFL	CPU time
Godunov	0.01 s	1	13.25 s
LCFL	1.2 s	100	0.14 s

Table 2: Linear advection equation. CPU times for the classical, explicit Godunov method and the LCFL algorithm. Kernel function: Eq. (15), with $(D, b) = (2\text{ m}, 1.5)$.

The computational results are shown in Figure 4. The Godunov solution is subjected to extreme numerical diffusion owing to the very small time step enforced by the narrow cell. This results in a CFL value of 0.01 in the coarse cells, hence the strongly damped profile. In contrast, the LCFL solution is subjected to a much milder damping. Besides, the CPU time is reduced by two orders of magnitude compared to the Godunov solution.

5.2 Burgers equation

As in Subsection 5.1, a Burgers-type equation

$$\partial_t u + \partial_x \left(k u^2 \right) = 0 \tag{54}$$

where k is a constant, is solved over an irregular grid using the classical Godunov and the proposed LCFL methods. The mesh charateristics and initial conditions are the same as those of the previous test (see Table 1), the constant k in Eq. (54) is set to unity. The convolution kernel is the same as

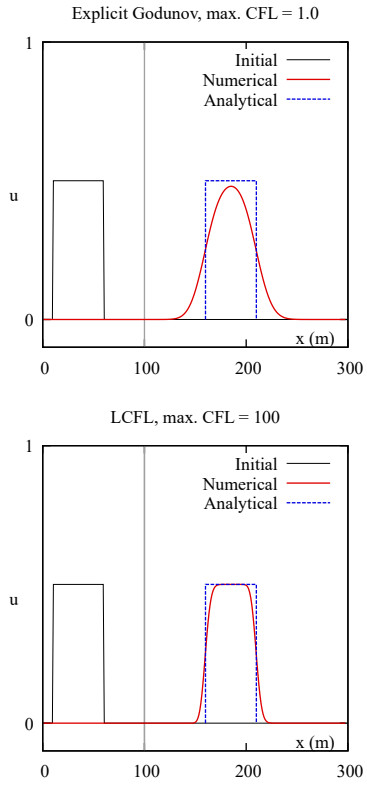

Figure 4: Linear advection equation. Top : Godunov solution. Bottom: LCFL solution. The narrow cell ($x = 100$ m) is indicated by the vertical grey line in the graphs.

.

that of the previous test. Given the initial conditions, the wave propagation speed $\lambda = 2ku$ ranges between 0 and 1 m s^{-1}. The computational time step, maximum CFL value over the computation and the CPU times for this test are given in Table 3.

Method	Δt	Max. CFL	CPU time
Godunov	0.01 s	1	21.55 s
LCFL	1.0 s	100	0.188 s

Table 3: Burgers equation. CPU times for the classical, explicit Godunov method and the LCFL algorithm. Kernel function: Eq. (15), with $(D, b) = (2 \text{ m}, 1.5)$.

Figure 5 shows the profiles obtained using the two numerical methods. The self-sharpening character of the shock contributes to a reduced numerical diffusion in the Godunov solution compared to the linear advection case. The LCFL solution is slightly sharper than the Godunov solution, but the main advantage of the latter over the former is the CPU time reduced by two

orders of magnitude owing to the much larger permissible time step.

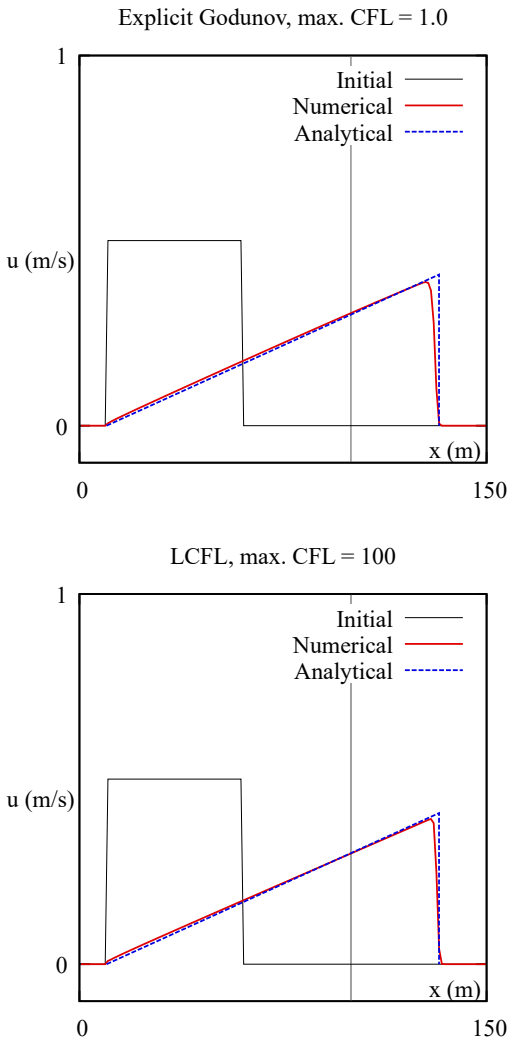

Figure 5: Burgers equation. The narrow cell ($x = 100$ m) is indicated by the vertical grey lines in the graphs.

5.3 Shallow water equations: dambreak problem over flat bottom

This test consists in solving the following Riemann problem

$$[h, q]^T(x, 0) = \begin{cases} [h_L, 0]^T & \text{if } x < 0 \\ [h_R, 0]^T & \text{if } x > 0 \end{cases} \tag{55}$$

The analytical solution is detailed in [35]. The parameters used in the present application are provided in Table 4. The problem is solved numerically using three approaches. Firstly, the

Parameter	meaning	numerical value
g	gravitational acceleration	9.81 m s^{-2}
h_L	water depth on left-hand side of dam	10 m
h_R	water depth on right-hand side of dam	5 m
Δx_0	default cell width	1 m
Δx_1	width of cells 101 and 202	10^{-2} m

Table 4: Dambreak problem parameters.

Godunov scheme is used over a regular grid with $\Delta x = 1$ m and computational time step Δt corresponding to the classical unit CFL stability constraint. Secondly, the Godunov scheme is used over an irregular grid with $\Delta x = \Delta x_0 = 1$ m everywhere, except at $x = \pm 50$ m, where $\Delta x = \Delta x_1 = 10^{-2}$ m. There again, Δt is such that the maximum CFL is unity, yielding a time step 100 times as small as in the first case. Thirdly, the LCFL algorithm is used, with three possible values for D ($D = 2$ m, $D =$5 m and $D = 10$ m) and a fixed $\nu' = 0.85$. The numerical parameters and the CPU times are given in Table 5.

Method	Δt	max. CFL	CPU time
Godunov, regular grid $\Delta x = 1$ m	8.8×10^{-2} s	1.0	0.438 s
Godunov, irregular grid $\Delta x_0 = 1$ m, $\Delta x_1 = 0.01$ m	8.8×10^{-4} s	1.0	43.0 s
LCFL, irregular grid, $D = 2$ m	0.15 s	170	0.28 s
LCFL, irregular grid, $D = 5$ m	0.3 s	340	0.18 s
LCFL, irregular grid, $D = 10$ m	0.6 s	680	0.11 s

Table 5: Dambreak problem. Numerical parameters and CPU times.

Figure 6 shows the water depths obtained from the three approaches at $t = 10$ s. The top graph shows the solution given by the Godunov scheme over the regular grid with a maximum CFL equal to unity. The middle plot shows the water depth profile obtained using the Godunov scheme on the irregular grid. With a CPU time almost 100 times larger, the solution is subjected to substantial numerical diffusion owing to the small CFL (smaller than or equal to 10^{-2}) in the cells of width $\Delta x = 1$ m. The bottom plot shows the solution obtained using the LCFL algorithm for a kernel support size $D = 2$ m. The LCFL algorithm allows for CFL values of 1.7 in the cells with $\Delta x = 1$ m and 170 in the two cells with $\Delta x = 10^{-2}$ m. The consequence is a milder numerical diffusion and reduced CPU times. The LCFL scheme is even slightly faster than the original Godunov scheme on the regular grid.

Figure 7 illustrates the sensitivity of the LCFL solution to the kernel support size D. As shown in Table 5, larger Δt values can be used as D increases because ν' is fixed. Accordingly, the CPU time decreases. As expected and can be seen in Figure 7, the numerical diffusion increases with D. Numerical diffusion is minimized when D is close to the size of the wider cells in the domain. Increasing D increases the number of cells of which the left and right states of the Riemann problem are averaged, thus leading to solution smoothing. There is thus a trade-off to be found between CPU time and solution accuracy.

5.4 Shallow water equations: dambreak problem over sinusoidal bottom

The purpose of the present subsection is to illustrate the performance of the method in the presence of a variable bottom elevation. More specifically, the well-balancing issues detailed in Subsection 3.3 and in Appendix B are addressed. This test uses the same initial conditions as in Subsection 5.3, but the bottom follows a continuous, sine wave law:

$$z_b(x) = a \cos\left(2\pi \frac{x}{L}\right) \tag{56}$$

with $(a, L) = (1 \text{ m}, 10 \text{ m})$. Four simulations are carried out. The first is the so-called refeence simulation. It uses Godunov's scheme with a regular cell width $\Delta x = 5$ cm. Since no analytical solution exists for the IVP over a sinusoidal bottom, the purpose of this simulation is to provide a reference solution with which the remaining solutions are to be compared. In the second simulation, the first-order Godunov scheme is used with a regular cell width $\Delta x = 1$ m. In the third, the Godunov scheme is used with $\Delta x = \Delta x_0$ everywhere, except in Cells 101 and 202, where Δx is set to Δx_1. The fourth simulation is the same as the third, except that the LCFL approach is used instead of the first-order Godunov scheme. The computational time steps, maximum CFL values and CPU times for these simulations are provided in Table 6

Method	Δt	max. CFL	CPU time
Godunov, regular grid $\Delta x = 5$ cm	4.5×10^{-3}	1.0	315 s
Godunov, regular grid $\Delta x = 1$ m	8.7×10^{-2} s	1.0	0.66 s
Godunov, irregular grid $\Delta x_0 = 1$ m, $\Delta x_1 = 0.01$ m	1.06×10^{-3} s	1.0	56.0 s
LCFL, irregular grid $\Delta x_0 = 1$ m, $\Delta x_1 = 0.01$ m	0.103 s	109.4	0.546 s

Table 6: Dambreak problem over sinusoidal bottom. Numerical parameters and CPU times.

Figure 8 shows the free surface elevation and unit discharge profiles resulting from the four simulations. Figure 9 is a zoomed view over the central part of the domain.

5.5 Dambreak over discontinuous bottom

In the present subsection, the performance of the method is assessed for a transient propagating over a discontinuous bottom

$$z_b(x) = \begin{cases} z_0 \text{ if } x \in [x_1, x_2] \\ z_1 \text{ otherwise} \end{cases} \tag{57}$$

The numerical values of the parameters are given in Table 7. This test is more challenging than that reported in the previous subsection. In Subsection 5.4, the difference between the bottom elevation of the cells decreases with Δx, while it is fixed in the case of a discontinuous bottom. Since the latter situation is as likely to occur as the former in practical applications, assessing the behaviour of the LCFL method for this configuration is important. Six simulations are carried out.

The first simulation uses a uniform cell width $\Delta x = 10^{-2}$ m. The first-order Godunov scheme is used to solve the dambreak problem numerically. The purpose is to provide a numerical solution that can be considered as numerically converged to the exact one. In the second simulation, the

Godunov scheme is used with $\Delta x = 1$ m. The third and fourth simulations use the same uniform cell width, except for the two cells at the bottom discontinuities. The cells on the lower side of the bottom discontinuity are assigned a width $\Delta x = \Delta x_1$ as given in Table 7. In the fifth and sixth simulations, the same cell width distributions are used, but the numerical scheme is the proposed LCFL approach.

Figures 10 and 11 show the profiles obtained for $\Delta x_1 = 10^{-2}$ m and $\Delta x_1 = 10^{-1}$ m respectively. For $\Delta x_1 = 10^{-2}$ m, the maximum CFL in the narrow cells is 120, while it is equal to 12 in the case $\Delta x_1 = 10^{-1}$ m. Figure 10 clearly shows that for $\Delta x_1 = 10^{-2}$ m the amplitude and propagation speed of the shock are erroneous. A correct amplitude and propagation speed are recovered for $\Delta x_1 = 10^{-1}$ m (Figure 11), by using a threshold $\nu_{\text{thres}} = 2$ in the source term limiter (see Appendix B). This however is achieved at the expense of strong peaks in the water depth and unit discharge profiles in the narrow cells in the immediate neighbourhood of the bottom discontinuity. Numerical experiments (not displayed here for the sake of conciseness) show that these oscillations are eliminated as soon as the narrow cells are not located next to the bottom discontinuity.

parameter	meaning	value
g	gravitational acceleration	9.81 m s^{-2}
x_1	abscissa of left step	-50 m
x_2	abscissa of right step	$+50$ m
z_0	lower bottom elevation	0 m
z_1	upper bottom elevation	3 m
z_L	initial free surface elevation on LHS of dam	10 m
z_R	initial free surface elevation on RHS of dam	5 m
Δx_0	default cell width	1 m
Δx_1	width of cells 101 and 202	0.1 m or 10^{-2} m

Table 7: Dambreak problem over discontinuous bottom. Test parameters.

Method	Δt	max. CFL	CPU time
Godunov, regular grid $\Delta x = 0.01$ m	1.01×10^{-3} s	1.0	56.5 s
Godunov, regular grid $\Delta x = 1$ m	8.8×10^{-2} s	1.0	1.22 s
Godunov, irregular grid $\Delta x_0 = 1$ m, $\Delta x_1 = 0.01$ m	1.01×10^{-3} s	1.0	113 s
Godunov, irregular grid $\Delta x_0 = 1$ m, $\Delta x_1 = 0.1$ m	1.01×10^{-3} s	1.0	10.2 s
LCFL, irregular grid $\Delta x_0 = 1$ m, $\Delta x_1 = 0.01$ m	0.105 s	120	1.11 s
LCFL, irregular grid $\Delta x_0 = 1$ m, $\Delta x_1 = 0.1$ m	0.105 s	12	1.12 s

Table 8: Dambreak problem over discontinuous bottom. Numerical parameters and CPU times.

6 Discussion - Conclusions

A large CFL, explicit finite volume algorithm for the solution of one-dimensional hyperbolic systems has been presented, with a focus on the shallow water equations. The proposed algorithm is non-iterative. It is based on a filtering technique that allows the stencils of the numerical schemes to be extended to an arbitrary number of cells. Sample applications include the linear advection

equation (Subsection 5.1), the inviscid Burgers equation (Section 5.2) and the one-dimensional shallow water equations (Subsections 5.3 - 5.5). The following conclusions may be drawn.

(a) The method can handle arbitrarily large CFL numbers. This does not mean however that the computational time step is unrestricted. It is subjected to a stability contraint similar to the CFL condition, where the width of the computational cell Δx is replaced with the size D of the domain of dependence of the solution. Sample applications show that the method performs well for local CFL values larger than 10^2.

(b) As shown in Appendix A, the smoothness of the numerical solution is influenced by the form of the convolution kernel used in the filtering process. Decreasing power law kernel functions are found to provide the best trade-off between kernel simplicity and permissible computational time step length.

(c) The method being non-iterative and based on a linearization for the solution of the Riemann problem, may produce oscillatory solutions in regions where the propagation speed of the waves changes rapidly and/or cancels within a single or a few cells (see Section A.3).

(d) Applications to the shallow water equations in the presence of topography-induced source terms show that the method allows static solutions to be preserved when used in combination with classical well-balancing techniques. The method performs equally well when flows over smooth (continous) topographies are simulated.

(e) However, in the presence of large CFLs and shocks moving over strongly discontinuous topography, the well-balancing error in the discretization of the source term are exacerbated in large CFL regions. This does not result in solution instability. Nevertheless, local, unphysical oscillations, restricted to the cells with the largest CFLs, are observed in the numerical solution. Besides, for large CFL values, weak solutions over discontinuous topography may be locally wrong (inaccurate shock amplitude and propagation speed).

(f) The optimal size D results from a trade-off between solution accuracy and computational rapidity, as illustrated by the sensitivity analysis in Subsection 5.3. The convolution kernel acts as a smoothing procedure, the strength of which increases with the ratio $D/\Delta x$. If D is significantly larger than the average cell width in the computational domain, smoothing occurs over large distances within the domain, thus yielding a strongly degraded solution. Conversely, if D is significantly smaller than the average Δx, the CFL number ν is smaller than unity over wide distances, thus yielding numerical diffusion over large parts of the domain. Consequently, the optimal size D is deemed to be of the same order of magnitude as the average size of the comutational cells.

(g) The maximum permissible pseudo-CFL number $\nu' = \lambda \Delta t / D$ for stability depends on the kernel function f_w, the length D of its support and the number $K = D/\Delta x$ of computational cells within D. For practical applications, it would be interesting to derive a stability constraint that does not depend on K in order to speed up the computation of the permissible computational time step.

It is believed that the issue mentioned in (e) can be eliminated (or at least minimized to a large extent) by devising higher-accuracy source term balancing techniques. It is not known yet if such higher-order balancing may be borrowed directly from existing techniques or if they should be developed specifically for the proposed large CFL algorithm.

Paths for research include, but are not limited to the following aspects: (i) adapting the method to the shallow water equations in the presence of dry beds, (ii) generalizing the method to the solution of multidimensional hyperbolic systems, such as the 2D shallow water equations.

Appendix A. Performance of various convolution kernels

Necessary conditions are consistency and positivity

$$\sum_{k=1}^{K} f_k = 1 \tag{58a}$$

$$f_k \geq 0 \quad \forall k \tag{58b}$$

In the present implementation of the kernel, several options verifying the above conditions have been tested :

$$f^{(1)} = \frac{1}{D} \tag{59a}$$

$$f^{(2)}(x) = \frac{e^{-a(x/D)^b}}{\int_0^D e^{-a(x/D)^b_k} \mathrm{d}x} \ , \quad a > 0, b > 0 \tag{59b}$$

$$f^{(3)} = \frac{(1 - x/D)^b}{\int_0^D (1 - x/D)^b \mathrm{d}x} \ , \quad a > 0 \tag{59c}$$

The performance of the various kernels is llustrated by solving the linear advection and Burgers equations

$$\partial_t u + \partial_x u = 0 \tag{60a}$$

$$\partial_t u + \partial_x \left(\frac{u^2}{2} \right) = 0 \tag{60b}$$

for the following initial and boundary conditions

$$u_i^0 = \begin{cases} 0 & \text{if } \frac{K}{\Delta x} \leq i \leq \frac{K}{\Delta x} + 500 \\ \frac{1}{2} & \text{otherwise} \end{cases} \quad \forall i \tag{61a}$$

$$u_{1/2}^n = 0 \, \forall \, n \tag{61b}$$

The parameters of the convolution kernels (see Figure 12) are given in Table 9. For a given

Kernel	Parameters	$\nu_{\max}$	$\nu'_{\max}$
$f^{(1)}$	$N = 60$	60	1.0
$f^{(2)}$	$(N, a, b) = (60, 1, 2)$	50.27	0.838
$f^{(2)}$	$(N, a, b) = (60, 3.5, 2)$	34.98	0.583
$f^{(2)}$	$(N, a, b) = (60, 3, 2)$	37.27	0.621
$f^{(3)}$	$(N, b) = (60, 0.5)$	47.62	0.794
$f^{(3)}$	$(N, b) = (60, 1.5)$	33.86	0.564

Table 9: Sample simulations. Kernel parameters and maximum permissible CFL values.

kernel support D, the maximum permissible CFL value $\nu_{\max}$ is strongly dependent on the type and parameters of the function. As shown by the table, the quicker the function decreases, the smaller $\nu_{\max}$. The largest $\nu_{\max}$ value is obtained for the constant function $f^{(3)}$. With $\nu_{\max}$ values smaller than 10, the inverse power function $f^{(2)}$ is of limited interest compared to the other three.

Figures 13 - 15 show the computational results obtained for the linear advection equation (60a) for various function parameters and CFL values. All four plots show the same trends.

(a) Decreasing the CFL value contributes to increase numerical diffusion and yields smoother numerical profiles. This is illustrated by the plots in Figs. 13 and 15 (compare top and bottom).

(b) Kernels giving lighter weights to large x tend to yield smoother profiles than functions giving heavier weights. This is illustrated by Figure 13, where the kernel $f^{(1)}$ is used. Of the three kernels tested, this is the kernel that gives the heaviest weights to large distances. This is also the kernel that yields the most irregular numerical profiles. The effect can also be observed in Figure 14 (compare middle and bottom).

(c) However, the smoothing effect of a quickly decreasing kernel is balanced with the corresponding decrease in the maximum CFL value. Consequently, for a given ν, decreasing the weight of large distances induces a smaller $\nu_{\max}$, thus a larger $\nu/\nu_{\max}$. This explains the counter-intuitive behaviour observed in Figure 15 (compare top and bottom). For the top graph, one has $\nu/\nu_{\max} = 0.67$, while $\nu/\nu_{\max} = 0.95$ for the bottom graph.

Figures 16 - 18 show the computational results obtained for the Burgers equation (60b). The added value of this test over the linear advection test is twofold. Firstly, the CFL is not uniform over the domain, which provides additional insights into the performance of the method. Secondly, the behaviour of the method in the presence of shocks can be assessed. The CFL value and kernel parameters used in this test are identical to those used for the linear advection test. The following conclusions may be drawn.

(d) All three solutions exhibit spikes at the tail of the rarefaction wave. This is due to a very large error in the size of the domain of dependence. At the tail of the rarefaction wave, the solution u is zero, which means that the domain of dependence of the exact solution has a zero size in all cells located to the left of the tail. However, this is not accounted for by the convolution approach, that is based on a fixed averaging size D. As a consequence, the fluxes across the first few interfaces in the tail are strongly underestimated, which results in a numerical solution almost identical to the initial one. This drawback is inevitable given the non-iterative nature of the approach.

(e) As in the linear advection test, Kernel $f^{(1)}$ yields the most scattered results. The function $f^{(3)}$ with $b = 1.5$ is the one giving the best performance (see Fig. 18).

In the light of these results, the function $f^{(3)}$ is retained for the applications presented in Section 5.

Appendix B. Source term balancing issues

The present appendix is devoted to a detailed motivation of the source term oscillation fix used in the algorithm. It is reminded from Subsection 3.1 that the mass and momentum equations

$$\partial_t h + \partial_x q = 0 \tag{62a}$$

$$\partial_t q + \partial_x M = -gh\partial_x z_b \quad , \quad M = \frac{q^2}{h} + \frac{g}{2}h^2 \tag{62b}$$

are discretized using finite volumes with fluxes computed by the HLL Riemann solver and source term splitting handled by an upwinding procedure [2, 8]

$$h_i^{n+1} = h_i^n + \frac{\Delta t}{\Delta x_i}\left(q_{i-1/2} - q_{i+1/2}\right) \tag{63a}$$

$$q_i^{n+1} = q_i^n + \frac{\Delta t}{\Delta x_i}\left(M_{i-1/2} - M_{i+1/2} + s_{i-1/2,R} + s_{i+1/2,L}\right) \tag{63b}$$

$$q_{i-1/2} = \frac{\lambda^+ q_L - \lambda^- q_R}{\lambda^+ - \lambda^-} - \frac{\lambda^-\lambda^+}{\lambda^+ - \lambda^-}(z_L - z_R) \tag{63c}$$

$$M_{i-1/2} = \frac{\lambda^+ M_L - \lambda^- M_R}{\lambda^+ - \lambda^-} - \frac{\lambda^-\lambda^+}{\lambda^+ - \lambda^-}(q_L - q_R) \tag{63d}$$

$$s_{i-1/2,L} = -\frac{\lambda^-}{\lambda^+ - \lambda^-}s_{i-1/2} \tag{63e}$$

$$s_{i-1/2,R} = \frac{\lambda^+}{\lambda^+ - \lambda^-}s_{i-1/2} \tag{63f}$$

where L and R indicate the left and right states for Interface $i-1/2$ and $s_{i-1/2}$ (see Equation (21d)) is the discretized version of the bottom source term. As shown in Subsection 3.3, the above discretization preserves static conditions, that is

$$(z_i, q_i) = (z_0, 0)\ \forall\ i \Longrightarrow \begin{cases} q_{i-1/2} = 0 & \forall\ i \\ M_L - M_R + s_{i-1/2} = 0 & \forall\ i \\ q_i^{n+1} = q_i^n = 0 & \forall\ i \\ z_i^{n+1} = z_i^n = z_0 & \forall\ i \end{cases} \tag{64}$$

Consider now a steady state situation with a non-zero unit discharge Q. Without loss of generality, assume $z_L < z_R$ in Eq. (21d). From continuity considerations, the interface unit discharges are all equal to Q. Consequently, one has from Eqs. (63c, 63d, 21d)

$$\frac{\lambda^+ q_L - \lambda^- q_R}{\lambda^+ - \lambda^-} - \frac{\lambda^-\lambda^+}{\lambda^+ - \lambda^-}(z_L - z_R) = Q\ \forall\ i \tag{65a}$$

$$\frac{Q^2}{z_L - z_{b,L}} + \frac{g}{2}(z_L - z_{b,L})^2 - \frac{Q^2}{z_R - z_{b,R}} - \frac{g}{2}(z_R - z_{b,R})^2 + \frac{g}{2}\left((z_L - z_{b,R})^2 - (z_L - z_{b,L})^2\right) = 0\ \forall\ i \tag{65b}$$

In contrast with the static case, the configuration $z_L = z_R$ does not fulfill Eq. (65b) because $\frac{Q^2}{z_L - z_{b,L}} \neq \frac{Q^2}{z_L - z_{b,L}}$ when $Q \neq 0$ and $z_{b,L} \neq z_{b,R}$. Consequently, a non-zero flow steady state

solution implies $z_L \neq z_R$, which means from Eq. (65a) that $(q_L, q_R) \neq (Q, Q)$. Steady state conditions thus require that the cell-averaged unit discharge q not be constant with respect to x. This is often misinterpreted as a loss of continuity, while it only implies that the discretization of the momentum equation lacks accuracy. The consequence of this is the presence of artificial oscillations in steady state cell-averaged q_i profiles, although the interface discharges $q_{i-1/2}$ are constant with respect to x. Such oscillations are exacerbated in the case of the large CFL algorithm because the momentum discretization error is multiplied by the ratio $\Delta t/\Delta x$ that may be 1 to 2 orders of magnitude larger than in usual numerical schemes.

Bearing in mind that the oscillations in the cell-averaged q-profiles stem only from a low-order discretization of the momentum equation and not from a loss of continuity, a straightforward momentum correction is applied : the value for q is limited by a simple min-max procedure only in the cells with the highest CFL values:

$$q_i^{n+1} \longmapsto \max(q_{\min}, \min(q_{\max}, q_i^{n+1})) \text{ if } \nu_i^{n+1} > \nu_{\text{thres}} \tag{66a}$$

$$q_{\max} = \max(q_{i-1}^{n+1}, q_{i+1}^{n+1}) \tag{66b}$$

$$q_{\min} = \min(q_{i-1}^{n+1}, q_{i+1}^{n+1}) \tag{66c}$$

where ν_{thres} is a user-defined CFL value above which the correction is applied. In the present applications, $\nu_{\text{thres}} = 10$ was selected.

References

[1] W. Bao and D. Jaksch. An explicit unconditionally stable numerical method for solving damped nonlinear schrödinger equations with a focusing nonlinearity. *SIAM Journal on Numerical Analysis*, 41(4):1406–1426, January 2003.

[2] Alfredo Bermudez and Ma Elena Vazquez. Upwind methods for hyperbolic conservation laws with source terms. *Computers & Fluids*, 23(8):1049–1071, November 1994.

[3] C. Chalons, M. Girardin, and S. Kokh. Large time step and asymptotic preserving numerical schemes for the gas dynamics equations with source terms. *SIAM Journal on Scientific Computing*, 35(6):A2874–A2902, January 2013.

[4] P. Colella and P.R. Woodward. The piecewise parabolic method (PPM) for gas-dynamical simulations. *Journal of Computational Physics*, 54(1):174–201, April 1984.

[5] J.P. Collins, P. Colella, and H.M. Glaz. An implicit-explicit eulerian godunov scheme for compressible flow. *Journal of Computational Physics*, 116(2):195–211, February 1995.

[6] R. Courant, K. Friedrichs, and H. Lewy. On the partial difference equations of mathematical physics. *IBM Journal of Research and Development*, 11(2):215–234, March 1967.

[7] S.F. Davis. Simplified second-order godunov-type methods. *SIAM Journal on Scientific and Statistical Computing*, 9(3):445–473, May 1988.

[8] L. Fraccarollo, H. Capart, and Y. Zech. A godunov method for the computation of erosional shallow water transients. *International Journal for Numerical Methods in Fluids*, 41(9):951–976, 2003.

[9] S. Soares Frazão and V. Guinot. An eigenvector-based linear reconstruction scheme for the shallow-water equations on two-dimensional unstructured meshes. *International Journal for Numerical Methods in Fluids*, 53(1):23–55, 2007.

[10] S.K. Godunov. A difference method for numerical calculation of discontinuous solutions of the equations of hydrodynamics. *Matematicheskii Sbornik*, 47(3):271–306, 1959.

[11] V. Guinot. The time-line interpolation method for large-time-step godunov-type schemes. *Journal of Computational Physics*, 177(2):394–417, April 2002.

[12] V. Guinot. An unconditionally stable, explicit godunov scheme for systems of conservation laws. *International Journal for Numerical Methods in Fluids*, 38(6):567–588, 2002.

[13] A. Harten. On a large time-step high resolution scheme. *Mathematics of Computation*, 46(174):379–399, 1986.

[14] A. Harten, P.D. Lax, and B. van Leer. On upstream differencing and godunov-type schemes for hyperbolic conservation laws. *SIAM Review*, 25(1):35–61, January 1983.

[15] Georges Kesserwani. Topography discretization techniques for godunov-type shallow water numerical models: a comparative study. *Journal of Hydraulic Research*, 51(4):351–367, June 2013.

[16] B. Van Leer. Towards the ultimate conservative difference scheme. IV. a new approach to numerical convection. *Journal of Computational Physics*, 23(3):276–299, March 1977.

[17] B.P. Leonard. Note on the von neumann stability of explicit one-dimensional advection schemes. *Computer Methods in Applied Mechanics and Engineering*, 118(1-2):29–46, September 1994.

[18] B.P. Leonard, A.P. Lock, and M.K. Macvean. The nirvana scheme applied to one-dimensional advection. *International Journal of Numerical Methods for Heat & Fluid Flow*, 5(4):341–377, April 1995.

[19] B.P. Leonard, A.P. Lock, and M.K. MacVean. Conservative explicit unrestricted-time-step multidimensional constancy-preserving advection schemes. *Monthly Weather Review*, 124(11):2588–2606, November 1996.

[20] R.J. Leveque. Large time step shock-capturing techniques for scalar conservation laws. *SIAM Journal on Numerical Analysis*, 19(6):1091–1109, December 1982.

[21] R.J. Leveque. Convergence of a large time step generalization of godunov's method for conservation laws. *Communications on Pure and Applied Mathematics*, 37(4):463–477, July 1984.

[22] R.J. LeVeque. A large time step generalization of godunov's method for systems of conservation laws. *SIAM Journal on Numerical Analysis*, 22(6):1051–1073, December 1985.

[23] S. Lindqvist, P. Aursand, T. Flåtten, and A.A. Solberg. Large time step TVD schemes for hyperbolic conservation laws. *SIAM Journal on Numerical Analysis*, 54(5):2775–2798, January 2016.

[24] F. Liu and H. Dong. Second-order large time step wave adding scheme for hyperbolic conservation laws. *Journal of Computational Physics*, 408:109279, May 2020.

[25] M. Morales-Hernandez, P. García-Navarro, and J. Murillo. A large time step 1d upwind explicit scheme (cfl $>$ 1): Application to shallow water equations. *Journal of Computational Physics*, 231(19):6532–6557, August 2012.

[26] M. Morales-Hernández, A. Lacasta, J. Murillo, and P. García-Navarro. A large time step explicit scheme (cfl $>$ 1) on unstructured grids for 2d conservation laws: Application to the homogeneous shallow water equations. *Applied Mathematical Modelling*, 47:294–317, July 2017.

[27] J. Murillo, P. García-Navarro, P. Brufau, and J. Burguete. Extension of an explicit finite volume method to large time steps (cfl $>$ 1): application to shallow water flows. *International Journal for Numerical Methods in Fluids*, 50(1):63–102, 2006.

[28] M. Prebeg. *Large time step methods for hyperbolic conservation laws*. PhD thesis, Norwegian University of Science and Technology, 2017.

[29] M. Prebeg, T. Flåtten, and B. Müller. Large time step roe scheme for a common 1d two-fluid model. *Applied Mathematical Modelling*, 44:124–142, April 2017.

[30] M. Prebeg, T. Flåtten, and B. Müller. Large time step HLL and HLLC schemes. *ESAIM: Mathematical Modelling and Numerical Analysis*, 52(4):1239–1260, July 2018.

[31] Z. Qian and C.H. Lee. A class of large time step godunov schemes for hyperbolic conservation laws and applications. *Journal of Computational Physics*, 230(19):7418–7440, August 2011.

[32] P. Roe. Designing CFD methods for bandwidth—a physical approach. *Computers & Fluids*, 214:104774, January 2021.

[33] E.E. Rosinger. Optimal accuracy of unconditionally stable explicit numerical methods for nonlinear evolution PDE's. *Applied Mathematics Letters*, 7(5):97–100, September 1994.

[34] L. Savic and F.M. Holly. Dambreak flood waves computed by modified godunov method. *Journal of Hydraulic Research*, 31(2):187–204, March 1993.

[35] J.J. Stoker. *Water waves. The mathematical theory with applications*. Interscience Publishers Inc., New York, 1957.

[36] E.F. Toro. The HLLC riemann solver. *Shock Waves*, 29(8):1065–1082, June 2019.

[37] E.F. Toro, M. Spruce, and W. Speares. Restoration of the contact surface in the HLL-riemann solver. *Shock Waves*, 4(1):25–34, July 1994.

[38] R. Xu, D. Zhong, B. Wu, X. Fu, and R. Miao. A large time step godunov scheme for free-surface shallow water equations. *Chinese Science Bulletin*, 59(21):2534–2540, May 2014.

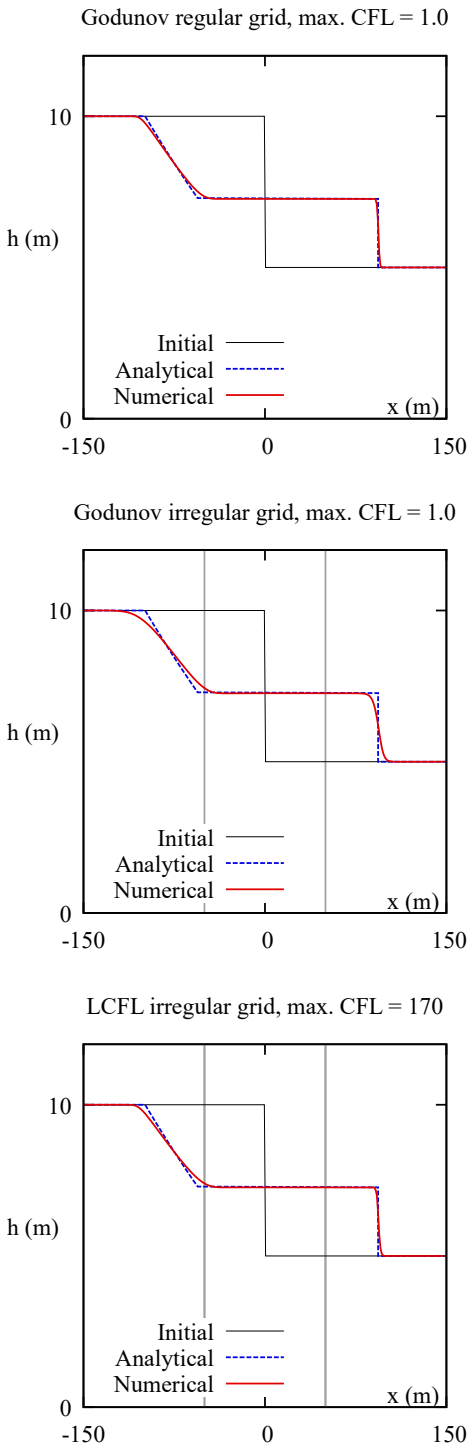

Figure 6: Dambreak problem. Top: solution computed by the Godunov scheme over a regular grid. Middle: solution computed by the Godunov scheme over an irregular grid with two narrow cells at $x = \pm 50$ m. Bottom: solution computed by the LCFL method over an irregular grid with two narrow cells at $x = \pm 50$ m. The locations of the narrow cells are indicated by the vertical grey lines in the middle and bottom graphs.

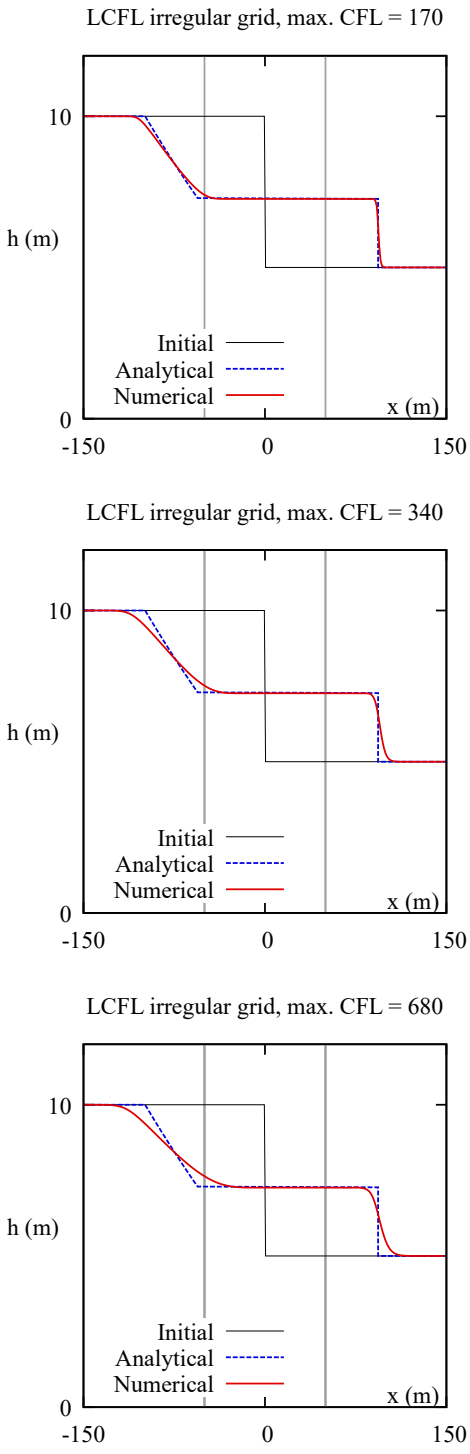

Figure 7: Dambreak problem. Solution computed by the LCFL method over an irregular grid with two narrow cells at $x = \pm 50$ m. Top: $D = 2$ m, $\Delta t = 0.15s$. Middle: $D = 5$ m, $\Delta t = 0.3s$. Bottom: $D = 10$ m, $\Delta t = 0.6s$.

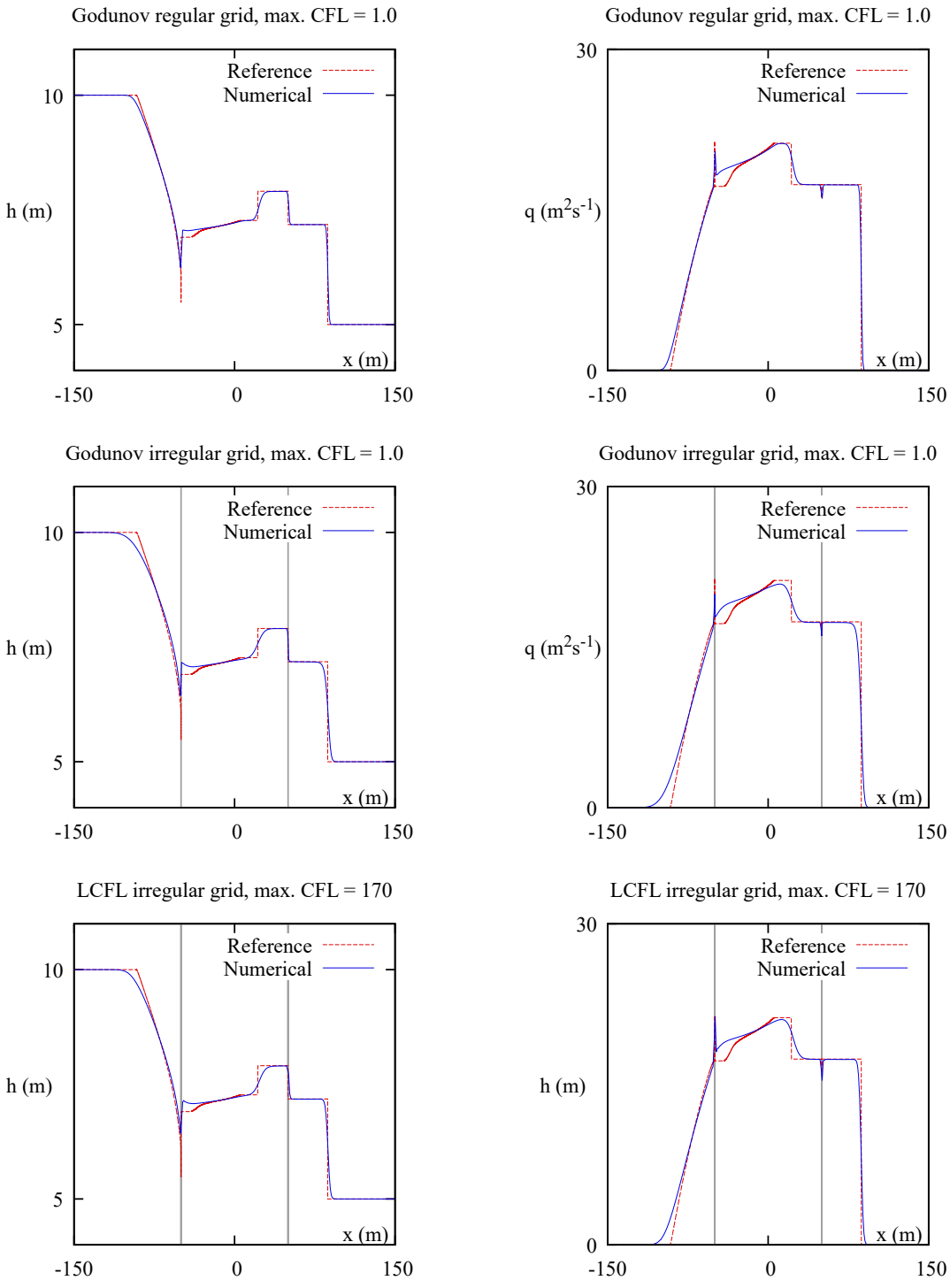

Figure 8: Dambreak problem over sinusoidal bottom. Top: solution computed by the Godunov scheme over a regular grid. Middle: solution computed by the Godunov scheme over an irregular grid with two narrow cells at $x = \pm 50$ m. Bottom: solution computed by the LCFL method over an irregular grid with two narrow cells at $x = \pm 50$ m. The locations of the narrow cells are indicated by the vertical grey lines in the middle and bottom graphs.

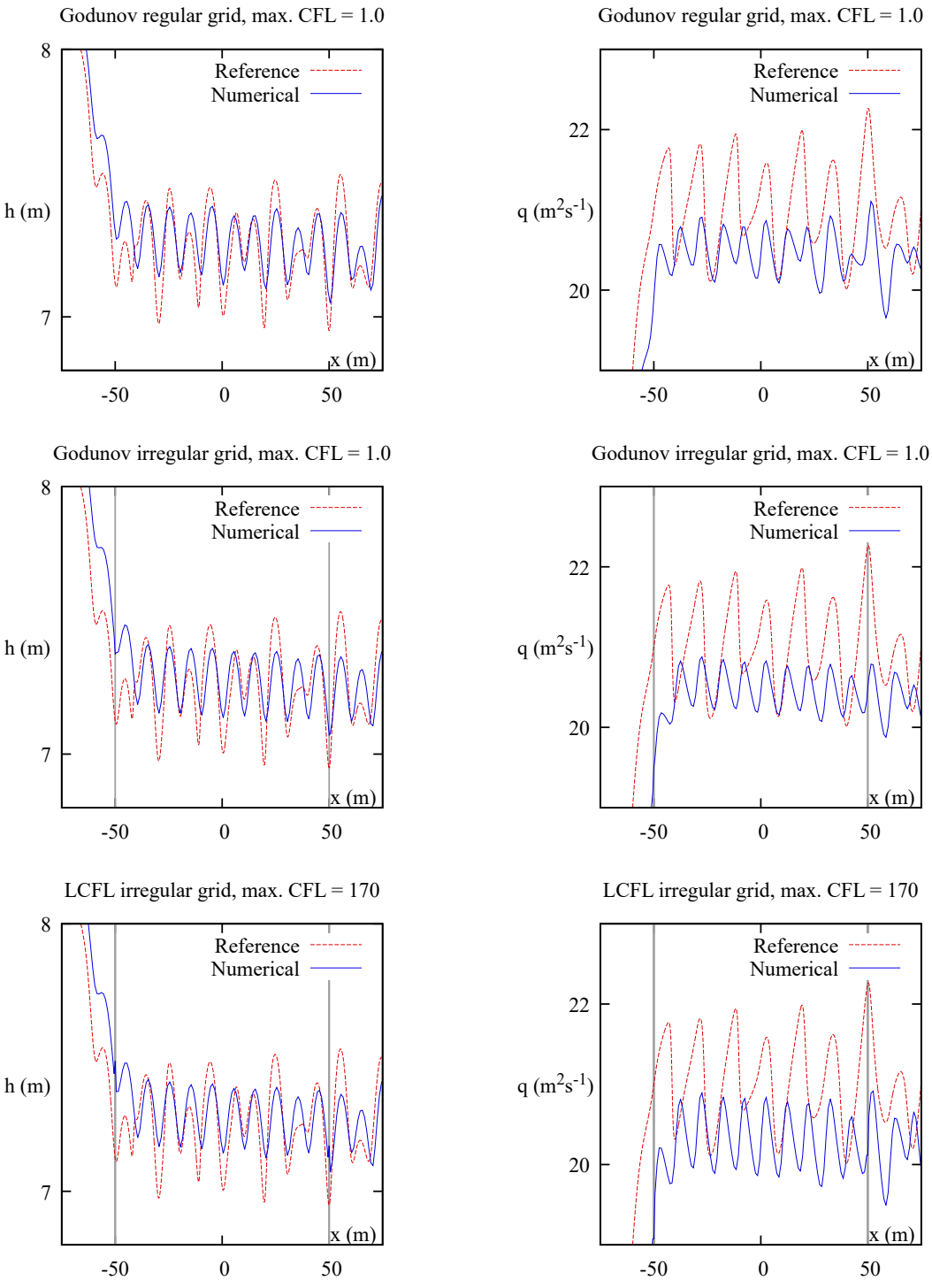

Figure 9: Dambreak problem over sinusoidal bottom. Zoomed graphs over the central parts. Top: solution computed by the Godunov scheme over a regular grid. Middle: solution computed by the Godunov scheme over an irregular grid with two narrow cells at $x = \pm 50$ m. Bottom: solution computed by the LCFL method over an irregular grid with two narrow cells at $x = \pm 50$ m. The locations of the narrow cells are indicated by the vertical grey lines in the middle and bottom graphs.

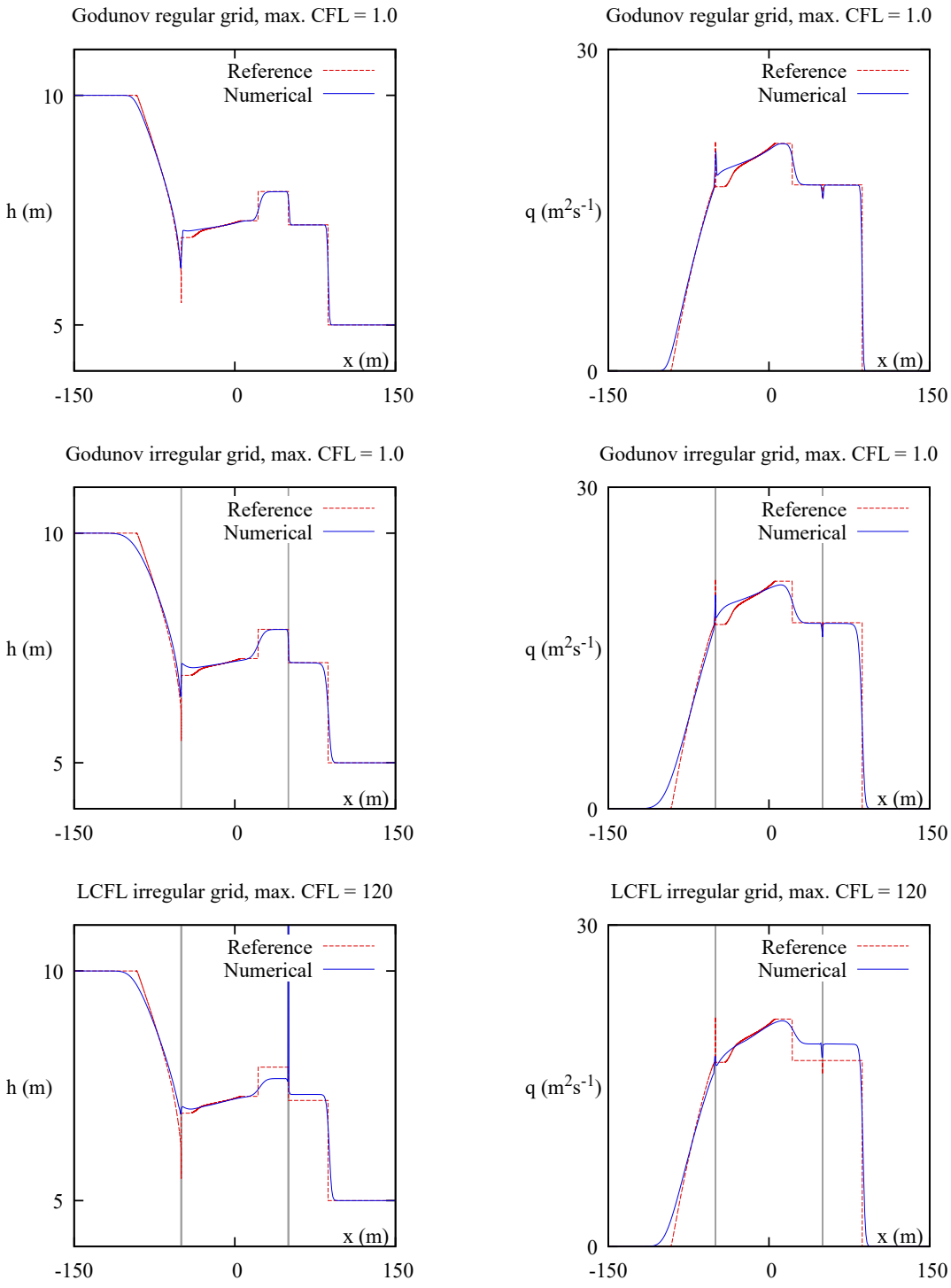

Figure 10: Dambreak problem over discontinuous bottom. Top: solution computed by the Godunov scheme over a regular grid. Middle: solution computed by the Godunov scheme over an irregular grid with two narrow cells $\Delta x_1 = 10^{-2}$ m at $x = \pm 50$ m. Bottom: solution computed by the LCFL method over an irregular grid with two narrow cells at $x = \pm 50$ m. The locations of the narrow cells are indicated by the vertical grey lines in the middle and bottom graphs.

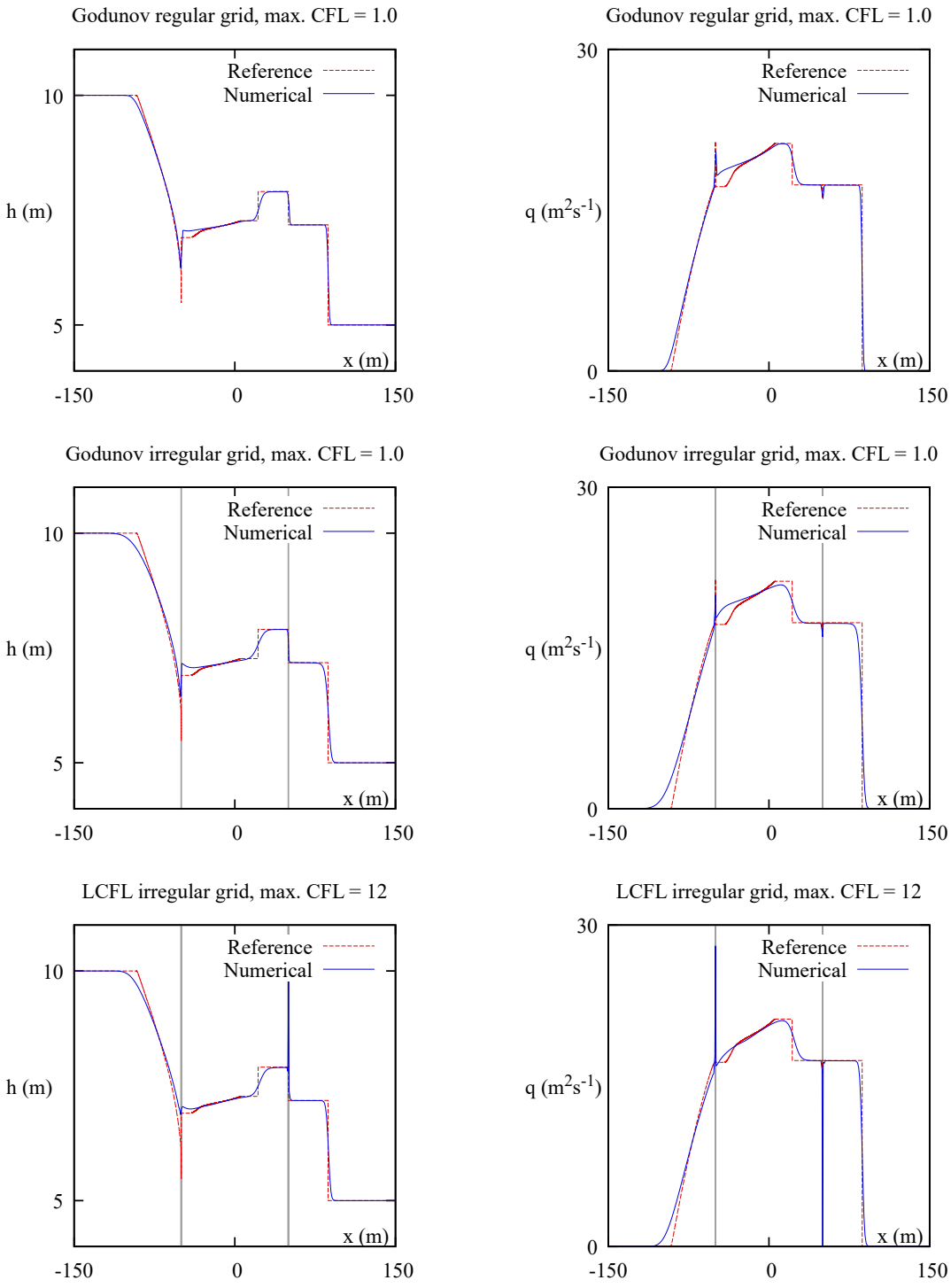

Figure 11: Dambreak problem over discontinuous bottom. Top: solution computed by the Godunov scheme over a regular grid. Middle: solution computed by the Godunov scheme over an irregular grid with two narrow cells $\Delta x_1 = 0.1$ m at $x = \pm 50$ m. Bottom: solution computed by the LCFL method over an irregular grid with two narrow cells at $x = \pm 50$ m. The locations of the narrow cells are indicated by the vertical grey lines in the middle and bottom graphs.

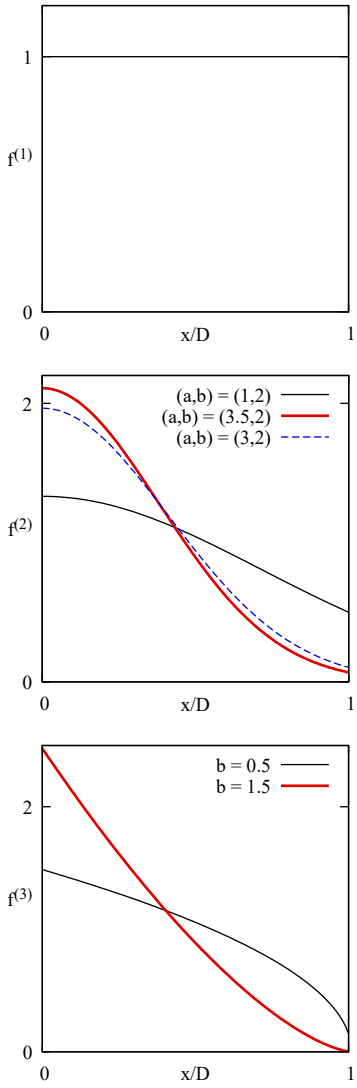

Figure 12: Sample simulations. Kernel functions for the parameters in Table 9.

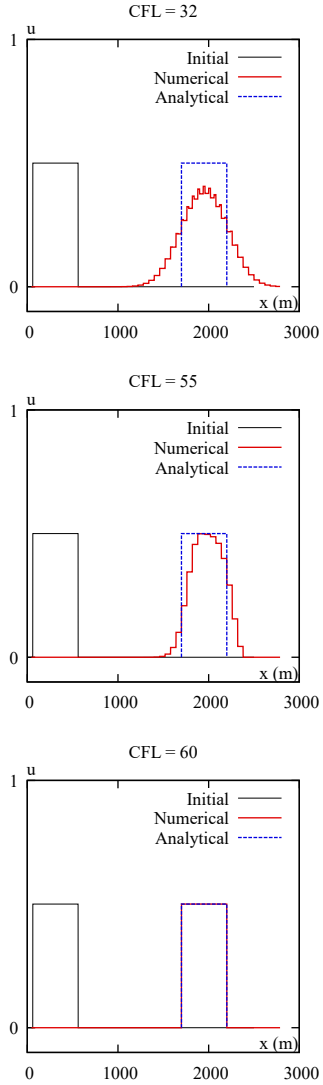

Figure 13: Linear advection equation. Solution for kernel function $f^{(1)}$. Max. permissible CFL value: 60.

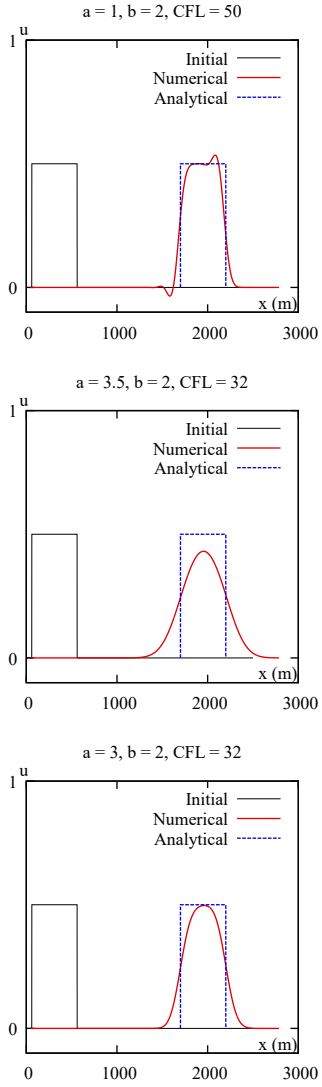

Figure 14: Linear advection equation. Solution for kernel function $f^{(2)}$. Max. permissible CFL values: 50.27 (top), 34.98 (middle), 37.27 (bottom).

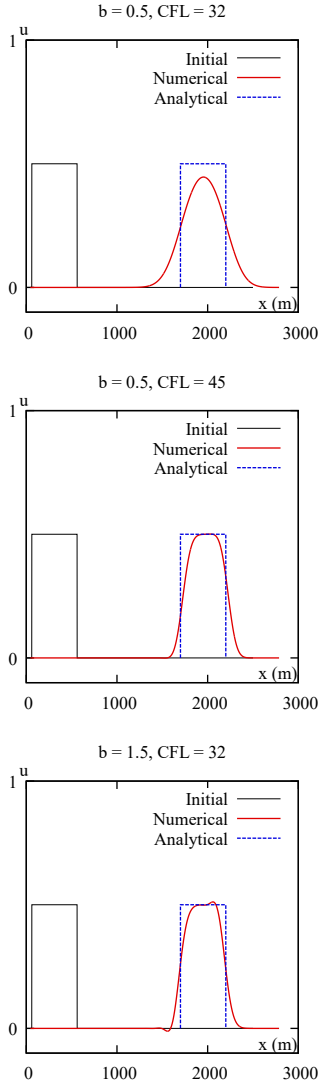

Figure 15: Linear advection equation. Solution for kernel function $f^{(3)}$. Max. permissible CFL values: 47.62 (top, middle), 33.86 (bottom).

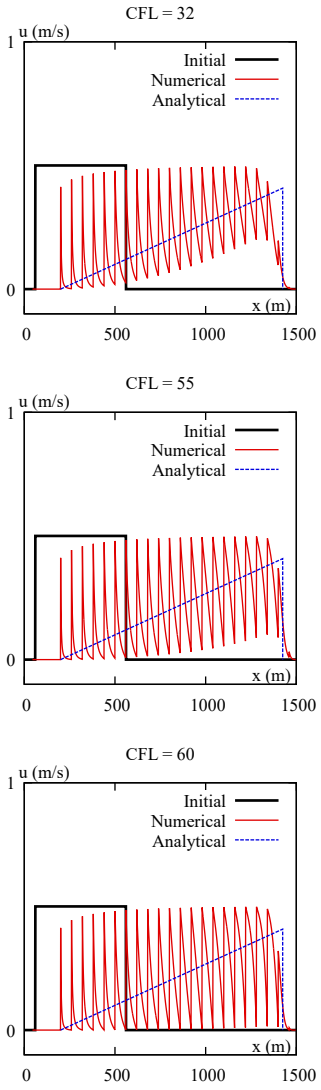

Figure 16: Burgers equation. Solution for kernel function $f^{(1)}$.

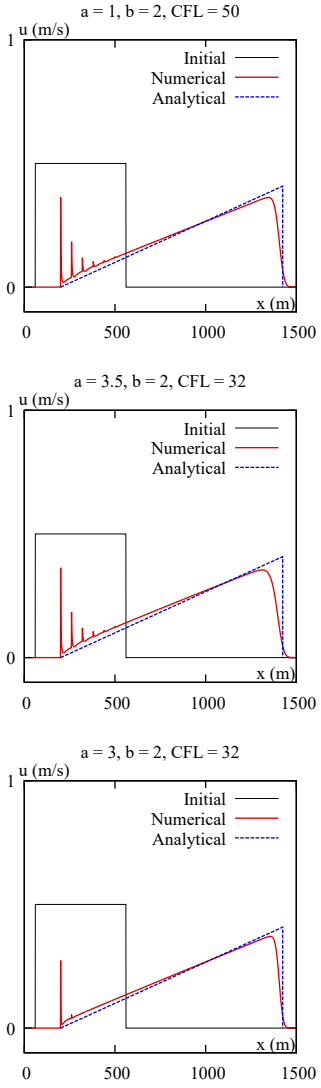

Figure 17: Burgers equation. Solution for kernel function $f^{(2)}$.

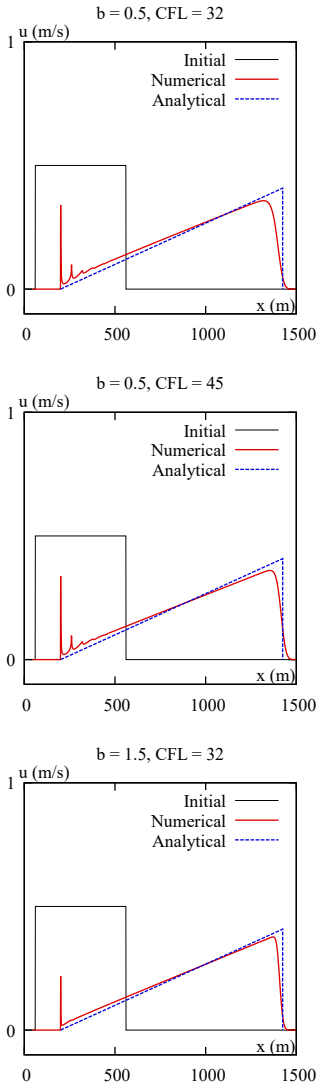

Figure 18: Burgers equation. Solution for kernel function $f^{(3)}$.