An Arbitrary High Order and Positivity Preserving Method for the Shallow Water Equations

M. Ciallella[(1)], L. Micalizzi[(2)], P. Öffner[(3)] and D. Torlo[(4)*]

(1): Team CARDAMOM, INRIA, Univ. Bordeaux, CNRS, Bordeaux INP, IMB, UMR 5251, France

(2): Institute of Mathematics, University of Zurich, Switzerland,

(3): Institute of Mathematics, Johannes Gutenberg-University Mainz, Germany,

(4): SISSA mathLab, SISSA, via Bonomea 265, 34136, Trieste.

All authors contributed equally to this project.

August 8, 2022

Abstract

In this paper, we develop and present an arbitrary high order well-balanced finite volume WENO method combined with the modified Patankar Deferred Correction (mPDeC) time integration method for the shallow water equations. Due to the positivity-preserving property of mPDeC, the resulting scheme is unconditionally positivity preserving for the water height. To apply the mPDeC approach, we have to interpret the spatial semi-discretization in terms of production-destruction systems. Only small modifications inside the classical WENO implementation are necessary and we explain how it can be done. In numerical simulations, focusing on a fifth order method, we demonstrate the good performance of the new method and verify the theoretical properties.

Keywords: positivity preserving, well-balanced, WENO, modified Patankar, shallow water, deferred correction.

1 Introduction

In the last years, the development of structure preserving high-order methods for hyperbolic conservation/balance laws have been an active field of research [5, 6, 14, 35, 24, 58, 63]. In the context of the shallow water equations, one is mainly interested in maintaining positive levels of water height, in conserving the equilibrium/stationary states and in entropy conservation/dissipation methods. There exists various ways to obtain these desired results, e.g. the applications of limiters for the positivity is only one example, cf. [8, 50, 38, 42, 49, 15, 65] and references therein.

In this paper, we also deal with these issues and we present a new high-order, well-balanced, positivity preserving method for the shallow water equation starting from a classical WENO scheme. In order to obtain well-balanced (WB) solutions, we subtract the residual of the a priori known stationary solution from our numerical scheme as shown in [8].

Then, to ensure the positivity of the water height, the modified Patankar Deferred Correction (mPDeC) method is used for the time-integration of this variable. Even if (modified) Patankar (mP) methods have been already used inside a numerical method for fluid simulations [30, 31, 40], those mP schemes have been based on extensions of classical RK methods and they are of maximum order three[1]. By applying the

*Corresponding authors: `mirco.ciallella@inria.fr` (M. Ciallella), `lorenzo.micalizzi@math.uzh.ch` (L. Micalizzi), `poeffner@uni-mainz.de` (P. Öffner), `davide.torlo@sissa.it` (D. Torlo)

[1]Mostly, they have been only used for the source terms, multicomponent terms or in a post-processing process.

modified Patankar trick inside the Deferred Correction (DeC) framework, the authors of [46] were able to construct a conservative, arbitrarily high-order and positivity preserving method for production-destruction systems (PDS) of ordinary differential equations.
To obtain a positive WENO spatial reconstruction, a positive limiter must be used [66, 48]. In this work the mPDeC is applied for the first time to a PDE problem, with a finite volume WENO spatial discretization. This lead to a high order accurate method enjoying all the previously cited properties (WB, positivity preservation).
Finally, in order to apply mPDeC on the semi-discretized problem, the finite volume method must be rewritten into a PDS and we explain in details how this can be done. It should be stressed out that only small modifications inside the classical finite volume implementation are necessary as it can be seen in the reproducibility repository [18]. To our opinion the modifications can be adapted to most WENO codes in a straightforward manner and the approach is a good alternative to already existing methods.
The paper is structured as follows: In Section 2, we introduce the considered model, the classical shallow water equations, and we repeat the basic properties focusing on steady state solutions and the positivity of the water height. Next, in Section 3 we describe the used classical finite volume WENO approach from [56, 57] and its well-balanced modification from [8]. We focus on the fifth order WENO method. However, all the ingredients to go to arbitrary high order can be found in the related repository [18]. In Section 4, the time-integration is considered focusing on Deferred Correction (DeC), the modified Patankar approach and its combination to the modified Patankar DeC (mPDeC) method developed in [46]. In Section 5, we describe how mPDeC can be combined with the WENO approach. It is important to interpret the semi-discretization in terms of production-destruction systems. Details of the implementation are given with additional algorithms. Then, in Section 6, we verify the theoretical properties of the scheme with numerical simulations with WENO5 focusing on the high-order accuracy, the well-balanced and the positivity preserving properties. In addition, we demonstrate also the excellent performance for more challenging test cases. Finally, in Section 7 we summarize the obtained results and perspectives for future works.
In Appendixes A and B we describe in details the WENO reconstruction and apply it for WENO5 with 4-points Gaussian quadrature rule, which, up to our knowledge, is not available in literature.

2 Shallow Water Equations

2.1 Model

The shallow water equations (SWE) model the behaviour of shallow free surface flows under the action of gravity. They are used to simulate the flows in rivers and coastal areas, and can be applied to predict tides, storm surge levels and coastline changes from hurricanes and ocean currents. They are also used in atmospheric flows, debris flows, and certain hydraulic structures like open channels and sedimentation tanks. SWEs take the form of non-homogeneous hyperbolic conservation laws with source terms modeling the effects of bathymetry and viscous friction. In this paper, we will consider the effect of the bathymetry as the only source term. If the bottom topography is assumed to be constant with respect to time, the SWEs can be recast in balance law form as:

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \mathcal{F}(\mathbf{u}) = \mathcal{S}(\mathbf{u}, x, y) \quad \text{on} \quad \mathbf{\Omega}_T = \mathbf{\Omega} \times [0, T] \subset \mathbb{R}^2 \times \mathbb{R}^+ \tag{1}$$

with conserved variables, flux and source terms given by

$$\mathbf{u} = \begin{bmatrix} h \\ hu \\ hv \end{bmatrix}, \quad \mathcal{F}(\mathbf{u}) = \begin{bmatrix} \mathbf{F} & \mathbf{G} \end{bmatrix} = \begin{bmatrix} hu & hv \\ hu^2 + g\frac{h^2}{2} & huv \\ huv & hv^2 + g\frac{h^2}{2} \end{bmatrix}, \quad \mathcal{S}(\mathbf{u}, x, y) = -gh \begin{bmatrix} 0 \\ \frac{\partial b}{\partial x}(x, y) \\ \frac{\partial b}{\partial y}(x, y) \end{bmatrix} \tag{2}$$

where h represents the relative water height, $\underline{u} = (u, v)$ are the flow speed components, g is the gravity acceleration and $b(x, y)$ is the local bathymetry. The source term helps modeling the effects induced on the flow caused by the bathymetry changes in space. Finally, it is also convenient to introduce the free surface water level $\eta := h + b$. All the aforementioned variables can be better interpreted by looking at Figure 1.

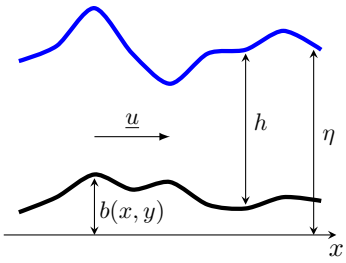


Figure 1: Shallow Water Equations: definition of the variables.

2.2 Properties of the model

As it is described *inter alia* in [10, 9, 37, 51, 65], the construction and development of effective and accurate numerical methods for the shallow water equations have received much interest in the last decades and it is still ongoing. In particular, one is interested in schemes that preserve physical quantities or structures from the continuous level. In this paper, we are targeting two types of difficulties which are often encountered when working with SWE: the preservation of steady state solutions and water height positivity[2].

2.2.1 Steady state solutions

The SWE system (2) is known to admit some steady state solutions whose form depends on the equilibrium between the source terms $\mathcal{S}$ and the remaining terms of the equations. The numerical simulations should be able to capture these behaviors even on coarse grids. Without additional techniques, many methods fail at balancing the source terms and the flux, resulting in small pertubations of the steady state. The perturbations could be then amplified by the method causing a bad approximation of the exact behaviour. This situation is sometimes called *numerical storm* in such context. To prevent it, one is interested in schemes that are capable of exactly balancing the flux and the source terms to obtain the desired steady-state solution. Numerical schemes enjoying this property are called **well-balanced schemes**.
The still water surface is often the first equilibrium taken in consideration. It is given by

$$u = v = 0; \qquad \eta(x, y, t) = h(x, y, t) + b(x, y) \equiv \eta_0 \in \mathbb{R}_0^+, \quad \forall\, (x, y) \in \mathbf{\Omega},\, t \in [0, T]. \tag{3}$$

It represents a steady-state solution, and is referred to as **lake at rest**. However, that is only a special case of the **moving water equilibrium**. Provided we verify the compatibility condition for the bathymetry which is $(-v, u) \cdot \nabla b = 0$, the steady state solutions are characterized by the invariants

$$h(x, y, t)\mathbf{u}(x, y, t) = Const. \quad \text{ and } E(x, y, t) = \frac{1}{2}||\mathbf{u}(x, y, t)||^2 + g(h(x, y, t) + b(x, y, t)) = Const. \tag{4}$$

Here, E is the specific total energy (moving water equilibrium variable), cf. [51] for more details.
Obviously, the lake at rest (3) is a special case of (4) when the velocity reduces to zero. Because of these reasons, it is desirable to solve (2) with well-balanced schemes and also our described method has this property as we shall describe in Section 3.2.

Remark 2.1 (C-Property)**.** *As described in [13, 50, 51], instead of speaking of well-balanced schemes, one could alternatively say that a scheme enjoys the* ***C-property*** *if it preserves exactly the steady state* (3)*. However one still speaks of C-property when referring to other steady states* (4)*. When the conservation of the steady state is not exact but is obtained within error rates below the formal accuracy of the scheme, one often speaks of* ***generalized C-property*** *[50].*

[2] We do not focus on entropy/energy preservation, see [9, 49] and reference therein for this topic.

2.2.2 Positivity of the solution

The other major difficulty which pops up in many simulations of the SWE is the appearance of dry regions in many real applications as for example dam break problems, flood waves and run-up phenomena at a coast with tsunamis. Here, the water height ($h = 0$) will be zero. As a result, all eigenvalues of the Jacobian of the flux coincide, cf. [51] and the SWE model will be not strictly hyperbolic anymore. If, by numerical oscillations h becomes negative, the problem is also not well-posed and the calculations will simply break down. It is thus essential for a good scheme to preserve the positivity of h at any time and any point. Especially, in situations when we have dry and wet areas, the scheme has to be constructed such that it can handle these numerical challenges. In this paper, we shall target the issue of positivity of the solutions by combining a positivity preserving time-integration method together with the WENO approach. More about this will follow in Section 4.3.

Remark 2.2. *Apart from the wet-dry areas in which the value of h approaches 0, we may have some regions of the space domain in which the bottom topography b overcomes the water level resulting in areas which are completely dry. In this context, the given definition (3) of the lake at rest steady state is unsatisfactory because in the mentioned regions we have $\eta_0 < b$ which, if we strictly stick to (3), would give $h < 0$ which does not correspond to the physics of the phenomen that must be modeled. Therefore in such cases we need to modify the definition of the lake at rest steady state introducing the so-called* ***dry lake at rest***

$$u = v = 0; \qquad h(x,y,t) = \begin{cases} \eta_0 - b(x,y), & \text{if } b \le \eta_0, \\ 0, & \text{else}, \end{cases} \qquad \forall\,(x,y) \in \mathbf{\Omega},\ t \in [0,T]. \tag{5}$$

3 Space discretization: Finite Volume method

The system of PDEs considered herein will be solved by means of the Method Of Lines (MOL). Thus, space and time are going to be treated separately. This section has the goal of presenting a standard high order finite volume framework. In particular, we will consider a Cartesian setting but the method can be easily applied to a general unstructured framework in a straightforward way.
The computational domain $\mathbf{\Omega}$ is covered with $N_x \times N_y$ non-overlapping control volumes

$$\Omega_{i,j} = [x_{i-1/2}, x_{i+1/2}] \times [y_{j-1/2}, y_{j+1/2}] \tag{6}$$

with the dimensions given by $\Delta x = x_{i+1/2} - x_{i-1/2}$ and $\Delta y = y_{j+1/2} - y_{j-1/2}$.
Considering the system of hyperbolic balance laws described by (1) and (2), for the control volume $\Omega_{i,j}$ we can define the cell average at time t:

$$\mathbf{U}_{i,j}(t) := \frac{1}{\Delta x \Delta y} \int_{x_{i-1/2}}^{x_{i+1/2}} \int_{y_{j-1/2}}^{y_{j+1/2}} \mathbf{u}(x,y,t)\, \mathrm{d}x\mathrm{d}y. \tag{7}$$

Integrating (1) over $\Omega_{i,j}$ provides the semi-discrete evolution formula with respect to $\mathbf{U}_{i,j}$

$$\frac{\mathrm{d}\mathbf{U}_{i,j}(t)}{\mathrm{d}t} + \frac{1}{\Delta x}(\mathbf{F}_{i+1/2,j}(t) - \mathbf{F}_{i-1/2,j}(t)) + \frac{1}{\Delta y}(\mathbf{G}_{i,j+1/2}(t) - \mathbf{G}_{i,j-1/2}(t)) = \mathbf{S}_{i,j}(t) \tag{8}$$

where $\mathbf{S}_{i,j}$ is the cell average of the source terms over cell $\Omega_{i,j}$ at time t

$$\mathbf{S}_{i,j}(t) := \frac{1}{\Delta x \Delta y} \int_{x_{i-1/2}}^{x_{i+1/2}} \int_{y_{j-1/2}}^{y_{j+1/2}} \mathcal{S}(x,y,t)\, \mathrm{d}x\mathrm{d}y \tag{9}$$

and $\mathbf{F}_{i+1/2,j}$ and $\mathbf{G}_{i,j+1/2}$ are the cell-averages of the physical fluxes over cell boundaries at time t:

$$\mathbf{F}_{i+1/2,j}(t) = \frac{1}{\Delta y} \int_{y_{j-1/2}}^{y_{j+1/2}} \mathbf{F}(\mathbf{u}(x_{i+1/2},y,t))\, \mathrm{d}y, \tag{10}$$

$$\mathbf{G}_{i,j+1/2}(t) = \frac{1}{\Delta x} \int_{x_{i-1/2}}^{x_{i+1/2}} \mathbf{G}(\mathbf{u}(x,y_{j+1/2},t))\, \mathrm{d}x. \tag{11}$$

Equation (8), along with all the previous definitions, is so far exact.
From now until the end of this section, by an abuse of notation, we will not explicitly write the dependence on t in all variables. $\mathbf{U}_{i,j}$, $\mathbf{F}_{i+1/2,j}$, $\mathbf{G}_{i,j+1/2}$ and $\mathbf{S}_{i,j}$ can be then approximated to the desired order of accuracy using appropriate quadrature formulae and reconstruction techniques. From now on, we shall focus only on $\mathbf{F}_{i+1/2,j}$ ($\mathbf{G}_{i,j+1/2}$ is obtained in a similar manner). Many low order and high order reconstruction techniques have been developed in the last decades. Few of them can be found in [26, 62, 19, 57, 25, 7, 20, 44]. Once the reconstruction has been performed, at each face we have two sets of values of $\mathbf{U}$, corresponding to $x^L_{i+1/2}$ and $x^R_{i+1/2}$, which will be referred to as the left and right extrapolated values:

$$\mathbf{u}^L_{i+1/2,\theta} = \mathbf{u}(x^L_{i+1/2}, y_\theta)\ , \quad \mathbf{u}^R_{i+1/2,\theta} = \mathbf{u}(x^R_{i+1/2}, y_\theta). \tag{12}$$

In our case, the weighted essentially non-oscillatory (WENO) [56, 57] reconstruction has been considered to avoid severe oscillations at discontinuities. By applying a consistent quadrature rule, and dropping the time dependence, the flux in the x-direction reads,

$$\mathbf{F}_{i+1/2,j} = \frac{1}{\Delta y}\sum_{\theta=1}^{N_\theta} w_\theta \mathbf{F}(\mathbf{u}(x_{i+1/2}, y_\theta)) = \frac{1}{\Delta y}\sum_{\theta=1}^{N_\theta} w_\theta \hat{\mathbf{F}}(\mathbf{u}^L_{i+1/2,\theta}, \mathbf{u}^R_{i+1/2,\theta}). \tag{13}$$

where the subscript $\theta = 1, \ldots, N_\theta$ corresponds to different Gaussian points $y_\theta \in [y_{j-1/2}, y_{j+1/2}]$ and weights $w_\theta \in [0, 1]$. The last step in the evaluation of the fluxes replaces $\mathbf{F}(\mathbf{u}(x_{i+1/2}, y_\theta))$ in (13) with a monotone and consistent numerical flux $\hat{\mathbf{F}}(\mathbf{u}^L, \mathbf{u}^R)$. The consistency is proved only if $\hat{\mathbf{F}}(\mathbf{u}, \mathbf{u}) = \mathbf{F}(\mathbf{u})$.
In our scheme we employ a Rusanov-type Riemann solver:

$$\hat{\mathbf{F}}(\mathbf{u}^L, \mathbf{u}^R) = \frac{1}{2}\left(\mathbf{F}(\mathbf{u}^R) + \mathbf{F}(\mathbf{u}^L)\right) - \frac{1}{2} s_{max}\left(\mathbf{u}^R - \mathbf{u}^L\right), \tag{14}$$

where s_{max} is the maximum eigenvalue of the normal flux-Jacobian of the system (1).
In order to express the dependence of some quantities on several cell averages $\mathbf{U}_{i,j}$, it is useful to collect them all in the vector $\mathbf{U}$.

3.1 Weighted Essentially Non-Oscillatory (WENO) method

As it is described in Shu's seminal paper [56] the classical WENO (as well as ENO) approach contains three major steps:

1. Use the WENO reconstruction procedure which will be described in the following to obtain the values at the Gaussian points. This step involves two one-dimensional reconstructions in the two directions.
2. Compute the numerical flux at quadrature points and integrate them at each cell interface.
3. Form a classical semidiscrete FV method and apply a time-integration method to update the cell averaged values.

3.1.1 Scalar reconstruction

The goal of the WENO method is to compute point-wise value of variable of interest $u(x, y)$ at Gaussian quadrature points $(x_{i+1/2}, y_\theta)$, in order to have a conservative and high order accurate procedure. In general, two ways can be followed to obtain the same result: genuine multidimensional reconstruction [55] and dimension-by-dimension reconstruction [59, 12]. The latter is a procedure made up by successive one-dimensional reconstruction sweeps and it is much simpler and less computationally expensive than the genuine multidimensional one. For this reason, we shall only focus on this one.
The high order reconstructed variables we are looking for will be referred to as $u^L_{i+1/2,\theta}$ and $u^R_{i+1/2,\theta}$. For the left values, we need to reconstruct the variable inside the cell $\Omega_{i,j}$, while, for the right values, similar arguments apply on the cell $\Omega_{i+1,j}$. We aim at reconstructing the variables with an accuracy of order p (p

odd). So, we define a stencil of p cells, $\{\Omega_{l_x,l_y}, \quad l_x = i-r+1,\dots,i+r-1,\ l_y = j-r+1,\dots,j+r-1\}$, where $2r-1=p$. For instance, WENO5 has accuracy $p=5$, with $r=3$, and uses a 5-cells stencil from $i-2$ to $i+2$.
In the first step of the two-dimensional reconstruction, a one-dimensional WENO reconstruction along the x-direction is performed obtaining the averages at cell interface $x_{i+1/2}$ with respect to the y-direction for $l_y = j-r+1,\dots,j+r-1$

$$v^R_{l_y} = \frac{1}{\Delta y}\int_{y_{l_y-1/2}}^{y_{l_y+1/2}} u(x^R_{i+1/2},y)\,\mathrm{d}y, \qquad v^L_{l_y} = \frac{1}{\Delta y}\int_{y_{l_y-1/2}}^{y_{l_y+1/2}} u(x^L_{i+1/2},y)\,\mathrm{d}y. \tag{15}$$

In the second sweep we perform another one-dimensional reconstruction along the y-direction in the Gaussian integration points on the y-axis ($x = x_{i+1/2}, y = y_\theta$), with $y_\theta \in [y_{j-1/2}, y_{j+1/2}]$. The reconstructed values can be, more generally, defined for each WENO sweep as the one-dimensional averages $q_i = \frac{1}{\Delta\xi}\int_{\xi_{i-1/2}}^{\xi_{i+1/2}} q(\xi)\,\mathrm{d}\xi$ of a function $q(\xi)$ where $\Delta\xi = \xi_{i+1/2} - \xi_{i-1/2}$ is the cell size.
For each one-dimensional step of the procedure, there are r candidate stencils for reconstruction. For each of these stencils, made up by r cells, there is a correspoding polynomial of degree $(r-1)$ referred as $p_m(\xi)$ $m = 0,\dots,r-1$. The goal of the WENO reconstruction is that of using all information coming from the r stencils employed for the reconstruction. For this reason, the WENO approach defines the reconstructed value as a convex combination of the r values of all polynomials in each quadrature point, weighted with positive nonlinear weights. The weights are chosen in order to achieve $(2r-1)$th order of accuracy when the solution is smooth and prefer the smoother stencils when discontinuities occur in the field. For a given (quadrature) point $\tilde{\xi}$ the design of weights consists of three steps. Firstly, the optimal linear weights d_m are sought so that the combination of all polynomials with these weights produces the polynomial of degree $(2r-2)$ corresponding to the large stencil. Then, the nonlinear weights ω_m can be defined as $\omega_m = \frac{\alpha_m}{\sum_{k=0}^{r-1}\alpha_k}$ with $\alpha_m = \frac{d_m}{(\beta_m+\epsilon)^2}$, where ϵ is a small constant introduced to avoid division by zero (we use $\epsilon = 10^{-6}$ in the simulations) and β_m are the smoothness indicators

$$\beta_m = \sum_{k=1}^{r-1}\int_{\xi_{i-1/2}}^{\xi_{i+1/2}} \left(\frac{\mathrm{d}^k}{\mathrm{d}x^k}p_m(\xi)\right)^2 \Delta\xi^{2k-1}\mathrm{d}\xi\ , \quad m = 0,\dots,r-1. \tag{16}$$

If some of d_m are negative then a special procedure must be used to tackle the reconstruction problem [55]. The final WENO reconstructed quantity is given by $q(\tilde{\xi}) = \sum_{k=0}^{r-1} p_k(\tilde{\xi})\omega_k$.
The numerical experiments presented herein have been performed through a piece-wise parabolic WENO5 reconstruction ($r=3$), which formally corresponds to fifth order accurate approximation for smooth solutions. However, in order to actually retain the fifth-order accuracy the quadrature formulae must be consistent with the WENO reconstruction. As Titarev and Toro stated in [59], the best results in terms of accuracy and computational cost for $r=3$ are obtained if the two-point Gaussian quadrature rule is used. However, this leads eventually to a formal fourth order of accuracy. For this reason, we implemented the four-point Gaussian quadrature rule with positive optimal weights. Up to our knowledge, the two-point Gaussian quadrature has already been thoroughly discussed in many references cited above. However, the case with 4-point Gaussian quadrature has not been fully described in literature, hence, we are going to introduce all the coefficients needed to use such formula in Appendix B and a Matlab script to compute the weights and coefficients for all orders is provided in [18].

Remark 3.1 (Positivity limiter)**.** *We aim at a positive solution and during the reconstruction procedure, it might happen that $h(x^L_{i+1/2})$ or $h(x^R_{i+1/2})$ become negative. In order to ensure that positive cell averages lead to positive reconstructions at the cell interfaces, we use the positivity limiter introduced by Perthame and Shu [48] and developed for two dimensional problems in [66]. The limiter is used in the simulation section with a parameter $\varepsilon = 10^{-6}$ as minimum water height, if not otherwise specified. We refer to [65, 66] for details on the implementation. This limiter when used in combination with the forward Euler (FE) method restricts the CFL conditions to $CFL^{FE} := w_1^{Lobatto}$ the weight of the Gauss–Lobatto quadrature rule of the*

corresponding space accuracy. For instance, with WENO5, $CFL^{FE} = 1/12$. The restriction slightly improves for high order SSPRK methods, for example we have $CFL^{SSPRK(5,4)} \approx 1.508 \cdot CFL^{FE}$. Unfortunately, explicit SSPRK methods are at most fourth order accurate, so for fifth order schemes (as DeC5), there is no warranty that the solution stays nonnegative under any CFL condition. Instead, we highlight that the new presented approach is unconditionally positive and thus not subjected to any CFL restriction.

3.2 Well-Balanced modification of the standard Finite Volume method

In order to achieve Well-Balancing with respect to the (eventually dry) lake at rest steady state, in this work we coupled the WENO formulation with a simple modification firstly introduced in [8]. The modification consists in recasting the original problem into an equivalent one in terms of the deviation of the seeked solution $\mathbf{U}$ from the reference solution $\tilde{\mathbf{U}}$ which must be preserved. In the particular case in which a steady solution ($\frac{\partial \tilde{\mathbf{U}}}{\partial t} = 0$) must be preserved, the modification leads to the new problem

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{U}_{i,j} + \frac{1}{\Delta x}(\mathbf{F}_{i+1/2,j}(\mathbf{U}) - \mathbf{F}_{i-1/2,j}(\mathbf{U})) - \frac{1}{\Delta x}(\mathbf{F}_{i+1/2,j}(\tilde{\mathbf{U}}) - \mathbf{F}_{i-1/2,j}(\tilde{\mathbf{U}})) + \\ \frac{1}{\Delta y}(\mathbf{G}_{i,j+1/2}(\mathbf{U}) - \mathbf{G}_{i,j-1/2}(\mathbf{U})) - \frac{1}{\Delta y}(\mathbf{G}_{i,j+1/2}(\tilde{\mathbf{U}}) - \mathbf{G}_{i,j-1/2}(\tilde{\mathbf{U}})) = \\ \mathbf{S}_{i,j}(\mathbf{U}) - \mathbf{S}_{i,j}(\tilde{\mathbf{U}}), \end{aligned} \tag{17}$$

which can be interpreted as a classical finite volume formulation with modified fluxes and source:

$$\begin{aligned} \overline{\mathbf{F}}_{i+1/2,j}(\mathbf{U}) &= \mathbf{F}_{i+1/2,j}(\mathbf{U}) - \mathbf{F}_{i+1/2,j}(\tilde{\mathbf{U}}), \\ \overline{\mathbf{G}}_{i,j+1/2}(\mathbf{U}) &= \mathbf{G}_{i,j+1/2}(\mathbf{U}) - \mathbf{G}_{i,j+1/2}(\tilde{\mathbf{U}}), \\ \overline{\mathbf{S}}_{i,j}(\mathbf{U}) &= \mathbf{S}_{i,j}(\mathbf{U}) - \mathbf{S}_{i,j}(\tilde{\mathbf{U}}). \end{aligned} \tag{18}$$

This approach is very easy to code and, further, the structures related to the steady reference solution can be computed in advance once and then used for every timestep without affecting the computational time. It must be underlined that, with this technique, all cell average computations, WENO reconstruction and source terms of the reference solution are performed following the same procedures and quadrature rules carried out for solving the balance law. Hence, all the terms always match when at the equilibrium.

4 Time discretization

After the description of the space discretization, we will introduce the time discretization in the this section. Due to the MOL approach, it is enough to focus here on the simple ODE case. The time discretization is one of the major point of this paper since we want to apply for the first time the arbitrary high-order, conservative and positivity preserving modified Patankar Deferred Correction method (mPDeC) together with the described WENO approach to the shallow water equations resulting in a high-order, conservative, unconditionally positivity preserving, non-oscillatory and well-balanced scheme. In agreement with [39], we refer to the numerical methods which are provably positive with respect to the water height without any CFL restriction, i.e., for all time steps $\Delta t > 0$, as **unconditionally** positivity preserving, in the context of the numerical solution of the shallow water equations.
Before describing the combined algorithm, we introduce the Deferred Correction (DeC) method [21] as described in [1, 6, 60] and we repeat its modification using the Patankar trick from [46, 45, 61].

4.1 Deferred Correction method

The general DeC approach has been introduced in [21] and has been further developed and applied in [17, 36, 41] in different contexts, whereas a simplified version was presented in [1]. In [1], a compact operator notation was also introduced and we shall follow this framework herein. However, even if the notation changed, the

main idea of DeC is always the same. DeC is based on the Picard-Lindelöf Theorem in the continuous setting and the classical proof makes use of Picard iterations to minimize the error and to obtain convergence. DeC is constructed to mimic these Picard iterations at the discrete level and decreases the approximation error in several iterative steps. To explain the method, we consider the following time-dependent initial value problem

$$y'(t) = f(y(t)), \qquad y(t_0) = y_0, \tag{19}$$

where $y : \mathbb{R} \to \mathbb{R}^S$ and $f : \mathbb{R}^S \to \mathbb{R}^S$ resulting from our MOL approach.
For our description, two operators are introduced: $\mathcal{L}^1$ and $\mathcal{L}^2$. Here, the $\mathcal{L}^1$ operator represents a low-order easy-to-solve numerical scheme, e.g. the explicit Euler method, and $\mathcal{L}^2$ is a high order operator that can present difficulties in its practical solution, e.g. an implicit RK scheme. The DeC method can be written as a combination of these two operators. Given a time interval $[t^n, t^{n+1}]$, we subdivide it into M subintervals

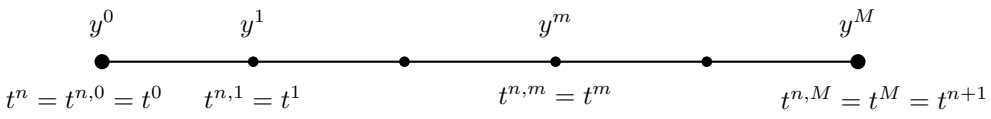


Figure 2: Time interval divided into subintervals

$\{[t^{n,m-1}, t^{n,m}]\}_{m=1}^M$, where $t^{n,0} = t^n$ and $t^{n,M} = t^{n+1}$. Therefore, we mimic for every subinterval $[t^0, t^m]$ the Picard–Lindelöf Theorem for both operators $\mathcal{L}^1$ and $\mathcal{L}^2$. With an abuse of notation, we drop the dependency on the timestep n for subtimesteps $t^{n,m}$ and substates $y^{n,m}$ as denoted in Figure 2.
Then, the $\mathcal{L}^2$ operator is given by

$$\mathcal{L}^2(y^0, \dots, y^M) := \begin{cases} y^M - y^0 - \Delta t \sum_{r=0}^M \theta_r^M f(y^r) \\ \vdots \\ y^1 - y^0 - \Delta t \sum_{r=0}^M \theta_r^1 f(y^r) \end{cases}. \tag{20}$$

where θ_r^m are the weights of a high order quadrature rule in $\{t^m\}_{m=0}^M$.
The $\mathcal{L}^2$ operator represents a high order numerical scheme if set equal to zero, i. e., $\mathcal{L}^2(y^0, \dots, y^M) = 0$. The order depends on the distribution of the subtimesteps, for instance, with M equispaced subtimesteps one obtains $(M+1)$th order, while with M Gauss–Lobatto quadrature subtimesteps one has $(2M)$th order. Unfortunately, the resulting scheme is implicit and, further, the terms f may be nonlinear. The $\mathcal{L}^1$ operator is given by the forward Euler discretization for each state y^m in the time interval, i. e.,

$$\mathcal{L}^1(y^0, \dots, y^M) := \begin{cases} y^M - y^0 - \beta^M \Delta t f(y^0) \\ \vdots \\ y^1 - y^0 - \beta^1 \Delta t f(y^0) \end{cases} \tag{21}$$

with coefficients $\beta^m := \frac{t^m - t^0}{t^M - t^0}$.
To simplify the notation and to describe DeC, we introduce the matrix of states for the variable y at all subtimesteps.

$$\mathbf{y} := (y^0, \dots, y^M) \in \mathbb{R}^{(M+1)\times S}, \text{ such that} \tag{22}$$

$$\mathcal{L}^1(\mathbf{y}) := \mathcal{L}^1(y^0, \dots, y^M) \text{ and } \mathcal{L}^2(\mathbf{y}) := \mathcal{L}^2(y^0, \dots, y^M). \tag{23}$$

The DeC algorithm uses a combination of the $\mathcal{L}^1$ and $\mathcal{L}^2$ operators to provide an iterative procedure. The aim is to recursively approximate $\mathbf{y}^*$, the numerical solution of the $\mathcal{L}^2(\mathbf{y}^*) = 0$ scheme, similarly to the Picard iterations in the continuous setting. The successive states of the iteration process will be denoted by the superscript (k), where k is the iteration index, e.g. $\mathbf{y}^{(k)} \in \mathbb{R}^{(M+1)\times S}$. The total number of iterations

(also called correction steps in the following) is denoted by K. To describe the procedure, we have to refer to both the m-th subtimestep and the k-th iteration of the DeC algorithm. We will indicate the variable by $y^{m,(k)} \in \mathbb{R}^S$. Finally, the DeC method can be written as

DeC Algorithm

$$\begin{aligned} &y^{0,(k)} := y(t^n), \quad k = 0, \dots, K, \\ &y^{m,(0)} := y(t^n), \quad m = 1, \dots, M, \\ &\mathcal{L}^1(\mathbf{y}^{(k)}) = \mathcal{L}^1(\mathbf{y}^{(k-1)}) - \mathcal{L}^2(\mathbf{y}^{(k-1)}) \text{ with } k = 1, \dots, K, \end{aligned} \tag{24}$$

where K is the number of iterations that we want to compute.
Using the procedure (24), we need, in particular, as many iterations as the desired order of accuracy p, i. e., $K = p$. This means that we choose the number of subtimesteps in a way that the order of the $\mathcal{L}^2$ operator is itself equal to p. In practice, for each correction and each subtimestep, $\mathcal{L}^{1,m}(\underline{\boldsymbol{y}}^{(k)}) = \mathcal{L}^{1,m}(\underline{\boldsymbol{y}}^{(k-1)}) - \mathcal{L}^{2,m}(\underline{\boldsymbol{y}}^{(k)})$ reduces to solve

$$y_\alpha^{m,(k)} - y_\alpha^0 - \Delta t \sum_{r=0}^{M} \theta_r^m f_\alpha(y^{r,(k-1)}) = 0, \qquad \forall \alpha = 1, \dots, I. \tag{25}$$

For more information and properties of the DeC approach, we refer to [3, 29] and references therein. In the following, we explain how to adapt the DeC approach to obtain a conservative and positivity preserving time integration scheme.

4.2 Patankar method for production-destruction systems

Many problems (19) in nature can be written as a production destruction system (PDS) for the unknown $y \in \mathbb{R}^S$

$$f_\alpha(y) = \sum_{\beta=1}^{S} (p_{\alpha,\beta}(y) - d_{\alpha,\beta}(y)), \tag{26}$$

where $p_{\alpha,\beta}, d_{\alpha,\beta} \geq 0$ are the production and destruction terms, respectively. The production and destruction terms are conveniently written as matrices. Applications for PDS are for example the biological and/or chemical reactions such as algal bloom [11]. Also parts (or all) of the semi discretization of hyperbolic conservation/balance laws can be interpreted in such PDS system as described in [30, 31, 39] and also later in this work. The calculated solutions are often describing physical quantities that enjoy some properties, for instance concentrations of chemicals or water height in the context of SWE should be nonnegative. The following definition may be introduced for ODE systems:

Definition 4.1. *An ODE (19) is called* positive, *if positive initial data $y_0 > 0$ result in positive solutions $y(t) > 0, \forall t$. Here, inequalities for vectors are interpreted componentwise, i.e., $y(t) > 0$ means $\forall \alpha \colon y_\alpha(t) > 0$. A PDS (26) is* conservative, *if $p_{\alpha,\beta}(y) = d_{\beta,\alpha}(y)$, $\forall\, \alpha,\, \beta,\, y$.*

These properties should be preserved by the numerical scheme as well. Thus, we introduce the following discrete counterpart.

Definition 4.2. *A numerical method computing $y^{n+1} \approx y(t_{n+1})$ given $y^n \approx y(t_n)$ is called* conservative, *if $\sum_\alpha y_\alpha^{n+1} = \sum_\alpha y_\alpha^n$. It is called* unconditionally positive, *if $y^n > 0$ implies $y^{n+1} > 0$ for any time step $\Delta t > 0$.*

From literature [11], it is well-known that the implicit Euler method is conservative and *unconditionally* positive preserving whereas the explicit Euler method is only conservative (it might be positive under time

step restrictions). To avoid solving a fully nonlinear system of equations, the so–called Patankar modifications have been applied to the explicit Euler method. To build an unconditionally positive numerical scheme, Patankar had the idea [47] of firstly weighting the destruction term in the original explicit Euler method with a coefficient as follows

$$y_\alpha^{n+1} = y_\alpha^n + \Delta t \left(\sum_{\beta=1}^{S} p_{\alpha,\beta}(y^n) - \sum_{\beta=1}^{S} d_{\alpha,\beta}(y^n) \frac{y_\alpha^{n+1}}{y_\alpha^n} \right), \quad \alpha = 1, \dots, S. \tag{27}$$

Indeed, the resulting scheme (27) is unconditionally positive and the implicit terms can be collected on the left hand side, but the conservation relation is violated. Burchard et al. had the idea [11] not only to weight the destruction term but also the production term:

$$y_\alpha^{n+1} = y_\alpha^n + \Delta t \left(\sum_{\beta=1}^{S} p_{\alpha,\beta}(y^n) \frac{y_\beta^{n+1}}{y_\beta^n} - \sum_{\beta=1}^{S} d_{\alpha,\beta}(y^n) \frac{y_\alpha^{n+1}}{y_\alpha^n} \right), \quad \alpha = 1, \dots, S. \tag{28}$$

They called their constructed scheme (28) **modified Patankar scheme** and proved that it is is unconditionally positive and conservative. The resulting scheme is linearly implicit, meaning that collecting all the implicit terms on the left hand side, we obtain a linear system at each time iteration. Based on this technique, extensions to second and third order modified Patankar Runge–Kutta (MPRK) methods have been made by several researchers in such context, cf. [30, 31, 33, 34]. Also the semi implicit RK methods proposed in [16] can be interpreted as Patankar methods as they weight only the destruction terms [61]. Finally, in [46] an arbitrarily high-order, conservative and positivity preserving scheme based on the DeC framework has been constructed. Herein we describe the main idea. For the details of the properties and proofs, we refer again to [46].

4.3 Modified Patankar Deferred Correction method

The modified Patankar Deferred Correction (mPDeC) is based on the DeC algorithm (25) and it consists in a modification of the $\mathcal{L}^2$ operator through the modified Patankar trick. This amounts to weight the production-destruction terms with respect to the intermediate approximations.
Using the fact that initial states $y_\alpha^{0,(k)}$ are identical for any correction k, the mPDeC correction steps can be rewritten [46] for $k = 1, \dots, K$, $m = 1, \dots, M$ and $\forall \alpha = 1, \dots, S$ as

$$y_\alpha^{m,(k)} - y_\alpha^0 - \sum_{r=0}^{M} \theta_r^m \Delta t \sum_{\beta=1}^{S} \left(p_{\alpha,\beta}(y^{r,(k-1)}) \frac{y^{m,(k)}_{\gamma(\beta,\alpha,\theta_r^m)}}{y^{m,(k-1)}_{\gamma(\beta,\alpha,\theta_r^m)}} - d_{\alpha,\beta}(y^{r,(k-1)}) \frac{y^{m,(k)}_{\gamma(\alpha,\beta,\theta_r^m)}}{y^{m,(k-1)}_{\gamma(\alpha,\beta,\theta_r^m)}} \right) = 0, \tag{29}$$

where θ_r^m are the DeC quadrature weights in time and

$$\gamma(\alpha, \beta, \theta) := \begin{cases} \alpha & \text{if } \theta \geq 0, \\ \beta & \text{if } \theta < 0. \end{cases}$$

Finally, the new numerical solution is $y^{n+1} = y^{M,(K)}$. As in the classical DeC framework, the choice of the distribution, the number of subtimesteps M and the number of iterations K determines the order of accuracy of the scheme. To reach order p, classically $M = p - 1$ equispaced subintervals and $K = p$ corrections should be used. As proven in [46], the scheme is conservative, positivity preserving and can reach arbitrary high order.
A description of the assembly of the mass matrix of the system (29) is described in Algorithm 1 and one timestep of the mPDeC algorithm is sketched in Algorithm (24) where the evolution formula is given by Algorithm 2.

Remark 4.3 (Subtimestep distribution). *In our numerical simulations, we apply Gauss-Lobatto nodes in every timestep. They have the advantage of requiring less subtimesteps to reach pth order of accuracy. In the following we will use $M = 3$ Gauss-Lobatto subtimesteps, which guarantee 6th order of accuracy for the operator $\mathcal{L}^2$ and $K = 5$ iterations aiming at a 5th order scheme to match the spatial discretization accuracy of WENO5.*

Remark 4.4 (Solution of the linear system). *At each subtimestep m and iteration (k) we need to solve the linear system given by (29). The mass matrix obtained has the following form:*

$$\mathbb{M}(y^{m,(k-1)})_{\alpha,\beta} = \begin{cases} 1 + \Delta t \sum_{r=0}^{M} \sum_{\beta=1}^{S} \frac{\theta_r^m}{y_\alpha^{m,(k-1)}} \left(d_{\alpha,\beta}(y^{r,(k-1)}) \mathbb{1}_{\{\theta_r^m > 0\}} - p_{\alpha,\beta}(y^{r,(k-1)}) \mathbb{1}_{\{\theta_r^m < 0\}} \right), & \text{for } \alpha = \beta, \\ -\Delta t \sum_{r=0}^{M} \frac{\theta_r^m}{y_\beta^{m,(k-1)}} \left(p_{\alpha,\beta}(y^{r,(k-1)}) \mathbb{1}_{\{\theta_r^m > 0\}} - d_{\alpha,\beta}(y^{r,(k-1)}) \mathbb{1}_{\{\theta_r^m < 0\}} \right), & \text{for } \alpha \neq \beta, \end{cases} \tag{30}$$

where $\mathbb{1}$ is the indicator function. The mass matrix assembly algorithm is described in Algorithm 1. The linear system will then read

$$\mathbb{M}(y^{m,(k-1)}) y^{m,(k)} = y(t^n). \tag{31}$$

Remark 4.5 (Division on almost wet areas). *When the water height is low, we might encounter troubles in computing the divisions in (30) as the denominator might be very small. They hypothesis behind the production and destruction system that says that as $h_\alpha \to 0$ also $d_\alpha \to 0$, can be difficult to be obtain at a numerical level. Hence, to be sure that those divisions do not lead to extremely high values when they should go to 0, we slightly modify the way we implement the division as suggested in [40]. Given any numerator n and denominator d of (30), we approximate the division by*

$$\frac{n}{d} \approx \begin{cases} 0 & d < \varepsilon, \\ \frac{2d \cdot n}{d^2 + \max\{d^2, \varepsilon\}} & d \geq \varepsilon, \end{cases} \tag{32}$$

with ε a small tolerance value. Along the computations, if not specified, we will use $\varepsilon := 10^{-6}$. This formulation allow to smoothly pass from $\frac{n}{d}$ to 0 as $d \to 0$. Moreover, when $d^2 \geq \varepsilon$ the division will be exact.

In the following, we apply the mPDeC time marching algorithm (29) to the WENO finite volume semidiscretization to solve the shallow water equations. Below we describe the actual implementation procedure.

5 Implementation of well-balanced mPDeC-WENO Scheme

In the following part, we will describe how the semi-discretization, using the WENO approach, can be written and interpreted as a PDS in order to apply the mPDeC scheme. We will see that only small modifications in the WENO code are necessary. Furthermore, this interpretation is not only restricted to the WENO procedure but can also applied to other high-order FV/FD discretizations. Our WENO approach is working only as a generic example. Finally, the complete algorithm will be presented.

5.1 WENO and production-destruction systems

First of all we must underline the fact that in order to preserve the positivity of the water height h, the mPDeC scheme is going to be applied only to the first equation of system (1) as h is the only variable that must stay nonnegative. Thus, a simple DeC approach is going to be used to evolve the momentum equations. So from now on, we shall only talk about the modifications introduced for the first equation to turn it into a production-destruction system. Given the foundations of finite volume schemes, each control volume has fluxes entering and exiting its boundary and, for each boundary face, the flux going from element $\alpha = [i, j]$ to element $\beta = [l, r]$ is going to be equal in module and opposite in sign to the flux from element β to element

Algorithm 1 Mass

Require: Production-destruction functions $p_{\alpha,\beta}(\cdot)$, $d_{\alpha,\beta}(\cdot)$, Δt, previous correction variables $\underline{\boldsymbol{y}}^{(k-1)}$, current subtimestep m.

$\mathbb{M} \leftarrow \mathbb{I}$
for $\alpha = 1$ **to** S **do**
  for $\beta = 1$ **to** S **do**
    for $r = 0$ **to** M **do**
      if $\theta_r^m \geq 0$ **then**
        $\mathbb{M}_{\alpha,\beta} \leftarrow \mathbb{M}_{\alpha,\beta} - \Delta t \theta_r^m \frac{p_{\alpha,\beta}(\boldsymbol{y}^{r,(k-1)})}{y_\beta^{m,(k-1)}}$
        $\mathbb{M}_{\alpha,\alpha} \leftarrow \mathbb{M}_{\alpha,\alpha} + \Delta t \theta_r^m \frac{d_{\alpha,\beta}(\boldsymbol{y}^{r,(k-1)})}{y_\alpha^{m,(k-1)}}$
      else
        $\mathbb{M}_{\alpha,\beta} \leftarrow \mathbb{M}_{\alpha,\beta} + \Delta t \theta_r^m \frac{d_{\alpha,\beta}(\boldsymbol{y}^{r,(k-1)})}{y_\beta^{m,(k-1)}}$
        $\mathbb{M}_{\alpha,\alpha} \leftarrow \mathbb{M}_{\alpha,\alpha} - \Delta t \theta_r^m \frac{p_{\alpha,\beta}(\boldsymbol{y}^{r,(k-1)})}{y_\alpha^{m,(k-1)}}$
      end if
    end for
  end for
end for
return $\mathbb{M}$

Algorithm 2 mPDeC Update formula

Require: $\underline{\boldsymbol{y}}^{(k-1)}$, Δt, production-destruction functions $p_{\alpha,\beta}(\cdot)$, $d_{\alpha,\beta}(\cdot)$, m.

Compute the mass matrix $\mathbb{M}(\boldsymbol{y}^{m,(k-1)}) \leftarrow$Mass$(p_{\alpha,\beta}(\cdot), d_{\alpha,\beta}(\cdot), \Delta t, \underline{\boldsymbol{y}}^{(k-1)}, m)$ using Algorithm 1
Compute $\boldsymbol{y}^{m,(k)}$ solving the linear system $\mathbb{M}(\boldsymbol{y}^{m,(k-1)})\boldsymbol{y}^{m,(k)} = \boldsymbol{y}^n$ given by (29) with Jacobi Algorithm 3
return $\boldsymbol{y}^{m,(k)}$

Algorithm 3 Jacobi iterative method

Require: $\mathbb{D}$ diagonal of the matrix, $\mathbb{L}$ off-diagonal terms of the matrix, $\mathtt{r}$ right hand side of the system, $\mathtt{tol}$ tolerance.

$\mathtt{err} \leftarrow 2 \cdot \mathtt{tol}, \quad k \leftarrow 0, \quad \mathtt{x}^k \leftarrow \mathtt{r}$
while $\mathtt{err} > \mathtt{tol}$ **do**
  $k \leftarrow k + 1$
  $\mathtt{x}^{k+1} \leftarrow \mathbb{D}^{-1}(\mathtt{r} - \mathbb{L}\mathtt{x}^k)$
  $\mathtt{err} \leftarrow ||\mathtt{x}^k - \mathtt{x}^{k-1}||$
end while
return $\mathtt{x}^{k+1}$

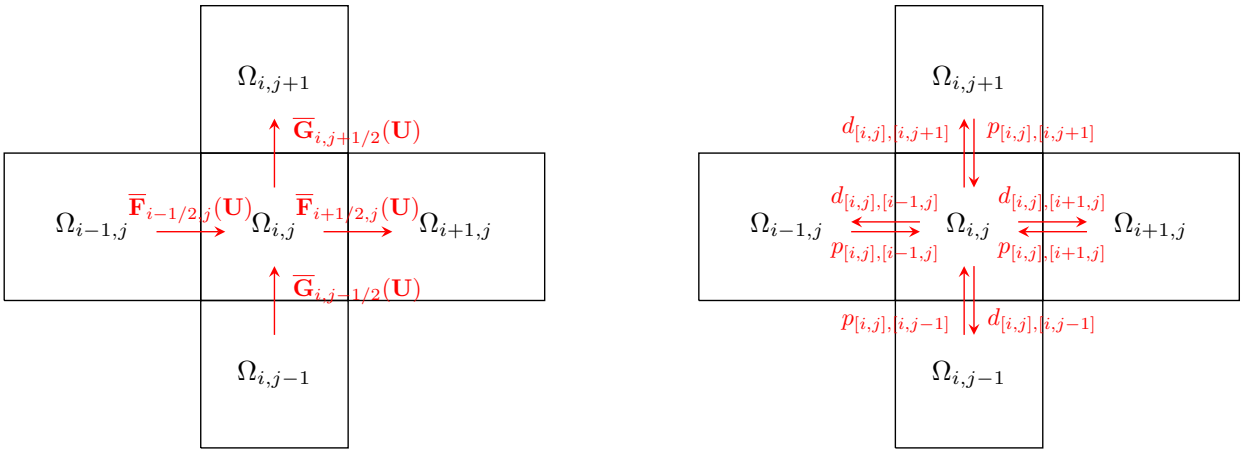


Figure 3: Cell $\Omega_{i,j}$, its four neighbors and its production and destruction terms.

α. This is the key feature for turning a finite volume schemes into a PDS. Note that the source is zero on h component.
Therefore, we can define the production and destruction terms for a general $\alpha = [i,j]$ and the neighboring $\beta = [l,r]$ as

$$\begin{aligned}
p_{[i,j],[i-1,j]}(\mathbf{U}) &= +\frac{1}{\Delta x}\overline{\mathbf{F}}_{i-1/2,j}(\mathbf{U})^+, \quad d_{[i,j],[i-1,j]}(\mathbf{U}) = -\frac{1}{\Delta x}\overline{\mathbf{F}}_{i-1/2,j}(\mathbf{U})^-, \\
p_{[i,j],[i+1,j]}(\mathbf{U}) &= -\frac{1}{\Delta x}\overline{\mathbf{F}}_{i+1/2,j}(\mathbf{U})^-, \quad d_{[i,j],[i+1,j]}(\mathbf{U}) = +\frac{1}{\Delta x}\overline{\mathbf{F}}_{i+1/2,j}(\mathbf{U})^+, \\
p_{[i,j],[i,j-1]}(\mathbf{U}) &= +\frac{1}{\Delta y}\overline{\mathbf{G}}_{i,j-1/2}(\mathbf{U})^+, \quad d_{[i,j],[i,j-1]}(\mathbf{U}) = -\frac{1}{\Delta y}\overline{\mathbf{G}}_{i,j-1/2}(\mathbf{U})^-, \\
p_{[i,j],[i,j+1]}(\mathbf{U}) &= -\frac{1}{\Delta y}\overline{\mathbf{G}}_{i,j+1/2}(\mathbf{U})^-, \quad d_{[i,j],[i,j+1]}(\mathbf{U}) = +\frac{1}{\Delta y}\overline{\mathbf{G}}_{i,j+1/2}(\mathbf{U})^+,
\end{aligned} \tag{33}$$

where with the superscript $^+$ and $^-$ we denote the positive and the negative part respectively. All the other $p_{\alpha,\beta}$ and $d_{\alpha,\beta}$ not defined here are set to 0. Clearly, this define a conservative and positive PDS, as the properties in Definition 4.1 are verified. The visualization of the production and destruction terms in Figure 3 may help the reader. We clearly observe that the matrices $(p_{\alpha,\beta})$ and $(d_{\alpha,\beta})$ are $S \times S$ sparse matrices, with, at most, 4 nonzero entries per row and $S = N_x \cdot N_y$.
Once the production and destruction matrices have been assembled, the next step consist in running the mPDeC algorithm in Eq. (29). Note that the matrix built in (30) is sparse as well with at most 5 nonzero entries for each row (4 nonzero entries of the production/destruction terms and the diagonal term). Hence, in the numerical computations we will use the classical Jacobi iterative method, see Algorithm 3, to obtain the solution of system (29) at each iteration. Indeed, it is provable [46, 11] that the Jacobi iteration algorithm converges when applied on the matrix defined in (30).
In all calculations, we set the tolerance to machine precision and the algorithm converge towards the solution in few iterations. Experimentally, we have seen that usually 10-20 iterations suffice, in the worst cases 40 iterations are needed and the number of iterations do not depend much on the mesh size. Overall, considering the assembly the mass matrix, the inversion of the system with Jacobi iterations, the whole mPDeC procedure increases the computational costs of around 18% with respect to the original DeC algorithm using the same CFL. The code has not been construct with the goal of optimizing all the procedures, so it might well be that this extra computational cost can be decreased with better implementations. The greatest advantage is anyway that no CFL restrictions are required in order to guarantee the positivity of the solution. So, a CFL= 1 suffices to guarantee the stability of the method. In the numerical test section, we will also detail the number of Jacobi iterations and the overhead computational cost that the method brings into the system

with respect to the explicit method.

Remark 5.1 (Efficiency and properties of Jacobi iterative method)**.** *Though being a very simple algorithm, Jacobi iterative method is particularly effective for this application. Indeed, the mass matrix in* $\mathbb{R}^{S\times S}$ *is very sparse, i.e., only* $5S$ *non-zero elements, hence, at each iterations, only* $5S$ *multiplications are computed. Moreover, experimentally, we have seen that the overhead computational cost of the Jacobi solver is around 10% of the whole computational cost and that the iterations needed are usually below 20, see section 6. Moreover, Jacobi guarantees the positivity of the solution at every iteration of the procedure.*
Clearly, there are other many iterative methods to solve linear system [54], e.g. Krylov preconditioning methods. All of these methods have larger complexity per iteration, but may converge faster to the solution of the system. Nevertheless, for most of these methods it is not possible to guarantee the positivity of the solution of the iterative solver along the process. That is why, we will use the Jacobi iterative method.

5.2 Full Algorithm

After describing how the WENO (FV) procedure can be interpreted as a PDS, we give a more precise description of the full algorithm which is used to calculate the numerical solution.
In our version of this approach, we have to adapt the steps taking into account the re-interpretation of the WENO approach as a production destruction system and the well balancing approach and this is described in Algorithm 4.
Finally, we can combine all the ingredients described above in a full algorithm as in Algorithm 5. There, we simply use the mPDeC to evolve in time and the production and destruction functions are given by the WENO description from above.

Algorithm 4 WENO FV with PDS structure

Require: $\mathbf{U}_{i,j}$, well balanced fluxes
1: Reconstruct on the quadrature points on cell interfaces the variable $\mathbf{U}$ in a high order fashion
2: Compute the numerical fluxes at quadrature points on cell interfaces $\hat{\mathbf{F}}(\mathbf{u}^L, \mathbf{u}^R)$
3: Subtract the correction for well balanced problems and obtain $\overline{\mathbf{F}}$ and $\overline{\mathbf{G}}$
4: Integrate over the cell interface to obtain the numerical fluxes $\mathbf{F}_{i+1/2,j}, \mathbf{F}_{i-1/2,j}$ and $\mathbf{G}_{i,j-1/2}, \mathbf{G}_{i,j+1/2}$
5: Compute $p_{\alpha,\beta}(\mathbf{U})$, $d_{\alpha,\beta}(\mathbf{U})$ as in (33)
6: **return** $p_{\alpha,\beta}(\mathbf{U})$, $d_{\alpha,\beta}(\mathbf{U})$

Algorithm 5 Full algorithm

Require: $\mathbf{U}^0_{i,j}$, T
1: $t = 0$
2: **while** $t < T$ **do**
3: Compute Δt by CFL restrictions
4: $\mathbf{U}^{n+1}$=DeC($\mathbf{U}^n, \Delta t$, WENOPDS) with DeC Algorithm (24) where update formula (29) is used for h and (25) is used for hu and hv and the WENO PSD function are given by Algorithm 4
5: $t = t + \Delta t$
6: **end while**

Conclusion 5.2. *The described method is high order accurate, positivity preserving, conservative, non-oscillatory and well-balanced for the shallow water equation.*

Remark 5.3 (Difference with respect to classical WENO)**.** *We want to highlight the differences between a classical WENO and the proposed algorithm are minimal. Indeed, once the spatial discretization is performed with a simple WENO step we need to apply two easy modifications. The first one consists in subtracting the flux related to the steady state variables. The second one consists in the definition of the production and*

destruction terms. Then the mPDeC can be applied as a simple time integration scheme. The code and these modifications are available at the reproducibility repository [18].

Remark 5.4 (Advantages of mPDeC)**.** *We shall remark that the presented method does not require any CFL constraint to obtain positive solutions for h, while the classical positivity limiter for WENO5 requires a CFL number of $1/12 \approx 0.083$. Clearly, a CFL number for a classical explicit method must be anyway used (between 1 and 1.5), but this allow to run the simulation with much less time steps than a classical explicit WENO scheme, with the extra computational cost of the Jacobi iterative method, which is negligible with respect the cost of decreasing the CFL number of factor of 12. Moreover, the procedure allows to have a provably positive method with arbitrarily high order of accuracy, while with positive limiters applied to WENO schemes only SSPRK methods guarantee the positivity of the solutions and they exist only up to order 4 [27].*

Remark 5.5 (Stability and accuracy of combined time-integration methods)**.** *We combined two time-integration methods, the mPDeC and the DeC approach. How can we assess the stability and accuracy properties the combined method. For the accuracy of the combined scheme, we can exclude a loss of accuracy as long as the two methods are consistent up to the same order of accuracy, even if applied separately to each equation. For instance, one could perform a simple Taylor expansion to prove the error behavior for each component and also in the combination step-after-step. In practice, as done in [46], it is sufficient to develop the Taylor expansion on the modified Patankar weighting coefficients to obtain a high order approximation of the DeC. This is also verified in our numerical tests. Stability is more problematic since it is not clear how to analyze it even for modified Patankar schemes in the ODE case, where only few preliminary works are available [32, 61]. This is also due to the nonlinear character of the scheme itself. Therefore, a stability investigation for PDEs is far beyond the goals of this paper. However, in general explicit methods have a more restrictive stability region than implicit methods. Therefore, we will simple assume that our stability region is determined by the underlying explicit method (DeC).*

6 Numerical Simulations

The goal of this section is to present the results obtained with the fifth order positivity-preserving mPDeC scheme, compared to that given by the classical fifth order DeC time integration method. The first test case consists in assessing the convergence properties of the spatial and temporal discretization on an unsteady vortex-type solution [53]. Afterwards, we focus on testing the well-balanced implementation for the lake at rest solution by showing its impact on perturbation analysis. Finally, three challenging simulations are performed to prove its capabilities to cope with wet-dry fronts. It is important to underline that in all simulations, especially those involving shocks, when there is no comparison with DeC5 (dam break problems), the mPDeC5 method obtains consistent results with respect to classical methods. For all the simulations carried out herein periodic boundary conditions have been considered together with the local Lax-Friedrichs (Rusanov) numerical flux.

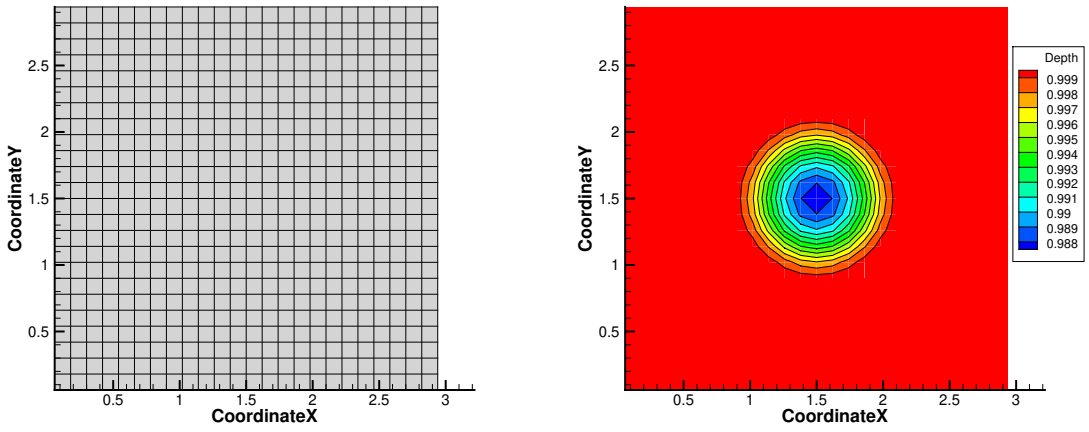
6.1 Unsteady vortex

In order to verify the order of accuracy we consider a moving smooth vortex. The computational domain is the square $[0,3] \times [0,3]$. The initial condition is given by some perturbations δ applied on a homogeneous background field $(h_0, u_0, v_0) = (1, 2, 3)$. Hence, the perturbation for the depth variable h is

$$h(r) = h_0 - \delta h(r) = h_0 - \gamma \begin{cases} e^{-\frac{1}{\arctan^3(1-r^2)}}, & \text{if } r < 1, \\ 0, & \text{else,} \end{cases} \qquad \text{with } r = \sqrt{(x-1.5)^2 + (y-1.5)^2} \tag{34}$$

and the vortex amplitude is $\gamma = 0.1$. The velocity field is affected by the following perturbations

$$\begin{pmatrix} \delta u \\ \delta v \end{pmatrix} = \sqrt{2\, g\, \partial_r h} \begin{pmatrix} (y-1.5) \\ -(x-1.5) \end{pmatrix}, \tag{35}$$



(a) Computational mesh

(b) Exact solution

Figure 4: Unsteady vortex: test case setting.

where $\partial_r h = \partial_r h(r)$ is a function of the radial distance from the center of the vortex

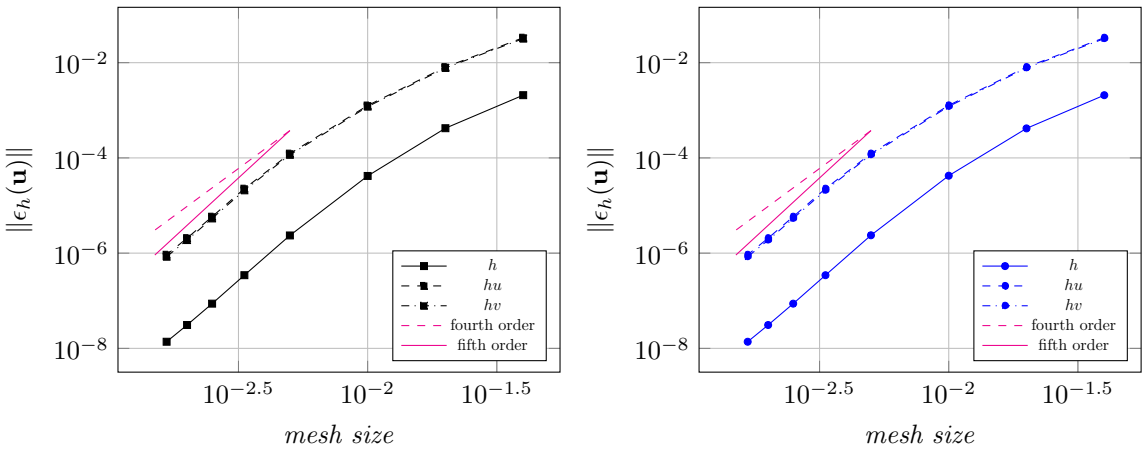
$$\partial_r h(r) = \frac{3\,\gamma\, e^{-\frac{1}{\arctan^3(1-r^2)}}}{\arctan^4(r^2-1)((r^2-1)^2+1)}. \tag{36}$$

It is important to highlight the fact that this solution is $\mathcal{C}^\infty$, which is a fundamental property for testing arbitrarily high order schemes. Many vortex-type solutions can be found available online but most of them can only be used to test lower order schemes. The exact solution of this problem is given by

$$h(x,y,t) = h(x-u_0t, y-v_0t, 0), \quad u(x,y,t) = u(x-u_0t, y-v_0t, 0), \quad v(x,y,t) = v(x-u_0t, y-v_0t, 0). \tag{37}$$

For the unsteady vortex, two convergence tests are run for WENO5 coupled with both the time integration schemes DeC5 and mPDeC to corroborate the fact that, for smooth flows, the results should be almost identical. We used CFL=0.7. The convergence tests are run on cartesian meshes of size 25×25, 50×50, 100×100, 200×200, 300×300, 400×400, 500×500 and 600×600. The computational mesh of the coarsest grid and initial condition for this test case are shown in Figure 4. For these convergence tests we used a tolerance $\varepsilon = 10^{-30}$ both for the positivity limiter and the mPDeC divisions, since the errors that we obtain are of the order of 10^{-8}. The error $||\epsilon_h(\mathbf{u})||$ is the $\mathbb{L}^1$ norm of the difference between the exact solution and the approximated one. Figure 5 points out the predicted fifth order behavior for both time integration schemes.

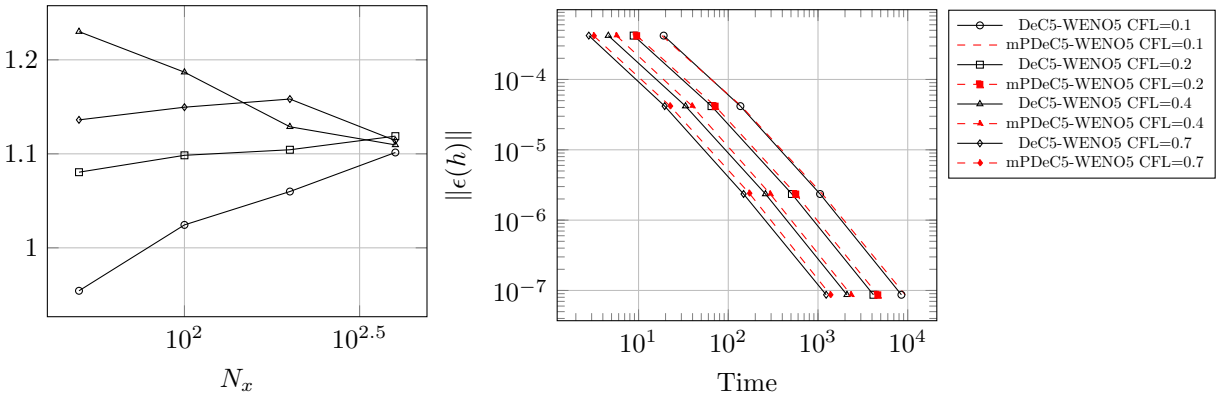
In Figure 6 we study the computational costs of the two methods (mPDeC5 and DeC5) with respect to different CFL numbers and mesh refinements. In Figure 6(a) we plot the ratio of computational time of mPDeC5 over the computational time of DeC5 needed to finish the simulation for the same CFL and mesh. We see that for fine meshes, when the computational times are more reliable, all ratios are close to 1.1. This means that the overhead that mPDeC5 requires, with respect to an explicit method, is of about 10%. In Figure 6(b) we plot errors with respect to computational times and we see a very small difference between mPDeC5 and DeC5, while there is a huge difference in computational costs when changing the CFL. The mPDeC5 is guaranteed to run at any $\text{CFL}^{\text{FE}} < 1$, while the positivity for other SSPRK methods is guaranteed only for $\text{CFL}^{\text{FE}} < 1/12$ with WENO5. Hence, there is huge advantage with mPDeC. Finally, in Figure 6(c) we can observe on average how many iterations are needed to solve the linear system and a confidence interval defined by the average $\pm\ \frac{1}{2}$ standard deviation. It is clear that Jacobi iterations are



(a) DeC5−WENO5

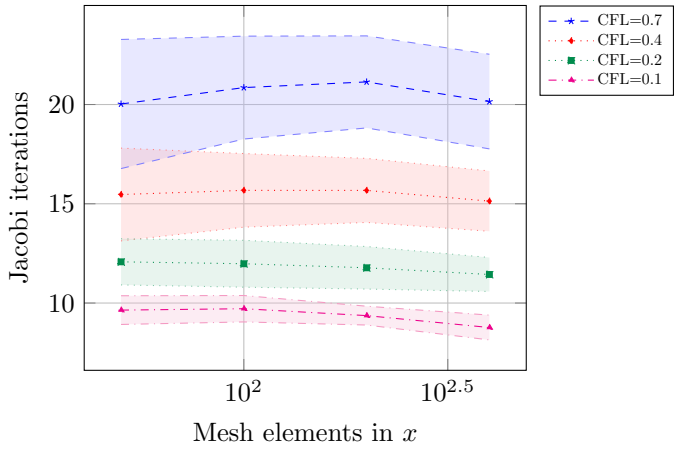
(b) mPDeC5−WENO5

Figure 5: Unsteady vortex: convergence tests.



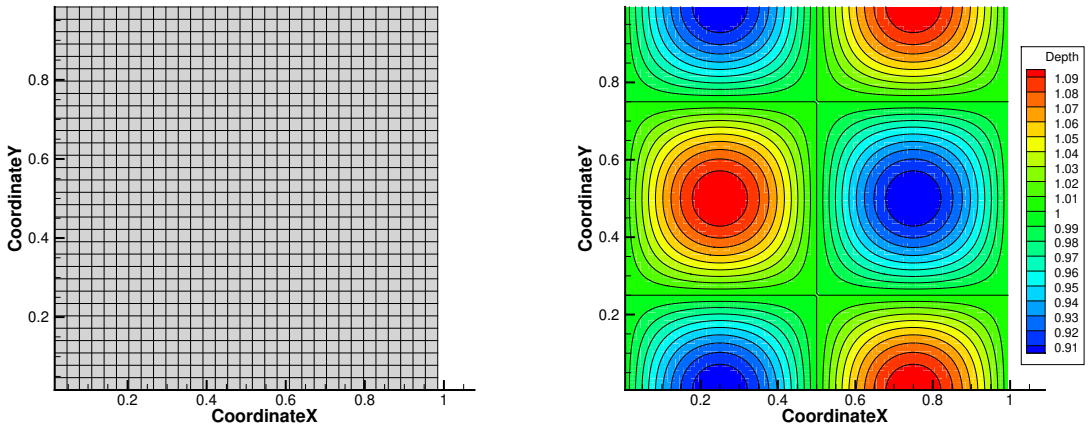
(a) Ratio of computational time of mPDeC over DeC

(b) Computational time and error



(c) Average of Jacobi iterations and confidence interval

Figure 6: Unsteady vortex test: computational time and Jacobi iterations



(a) Computational mesh

(b) Exact solution of h

Figure 7: Lake at rest: test case setting.

mainly driven by the CFL number, which explicitly appear as a coefficient of the mass matrix minus the identity. Indeed, all factors $\Delta t \frac{d}{h}$ or $\Delta t \frac{p}{h}$ are linearly dependent on the CFL as production and destruction terms are proportional to $\frac{1}{\Delta x}$.

6.2 Lake at rest

As already introduced in the theoretical part, this test case is needed to prove the presented scheme is well-balanced. The computational domain is the square $[0,1] \times [0,1]$ and the steady solution of this problem and bathymetry are briefly summarized below

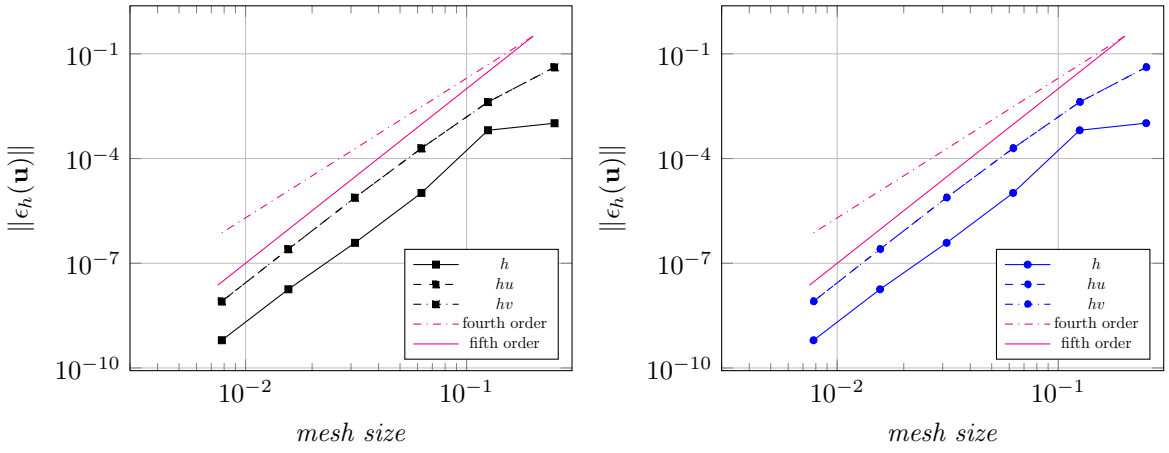
$$b(x,y) = 0.1 \sin(2\pi x) \cos(2\pi y), \qquad h(x,y) = 1 - b(x,y), \qquad u = v = 0. \tag{38}$$

This benchmark can also be used to test once again the order of accuracy of our discretization. Indeed, we expect the method to converge with a fifth order slope when not well-balanced and we expect machine precision errors for all the well-balanced tests. This simulation has been performed with four different settings: DeC5 and mPDeC5, well-balanced and not well-balanced. For all cases, we employed a fifth order WENO discretization for the spatial derivatives and CFL=0.9. Also for this test, we chose $\varepsilon = 10^{-30}$ to check the accuracy of the scheme with errors of the order of 10^{-10}. As expected, the error computed for the well-balanced simulations is exactly zero therefore we did not plot them along with the other results. The interested reader can run the simulations and test the properties of our method by downloading the code available at the reproducibility repository [18]. The Cartesian mesh employed for this convergence test are 4×4, 8×8, 16×16, 32×32, 64×64 and 128×128. The exact solution is presented in Figure 7, along with the 32×32 mesh. As can be noticed from Figure 8, mPDeC5 allows a fifth order convergence rate as theoretically proved with results almost identical to those of DeC5.

6.3 Wet-dry lake at rest

Now we test the capability of dealing with wet and dry regions of the scheme in a very simple context. We consider a bathymetry given by a bump

$$b(x,y) = \begin{cases} e^{1-\frac{1}{1-r^2}}, & \text{if } r^2 < 1, \\ 0, & \text{else}, \end{cases} \qquad \text{where } r^2 = x^2 + y^2, \tag{39}$$



(a) DeC5–WENO5 (b) mPDeC5–WENO5

Figure 8: Lake at rest: convergence tests without preserving the exact solution.

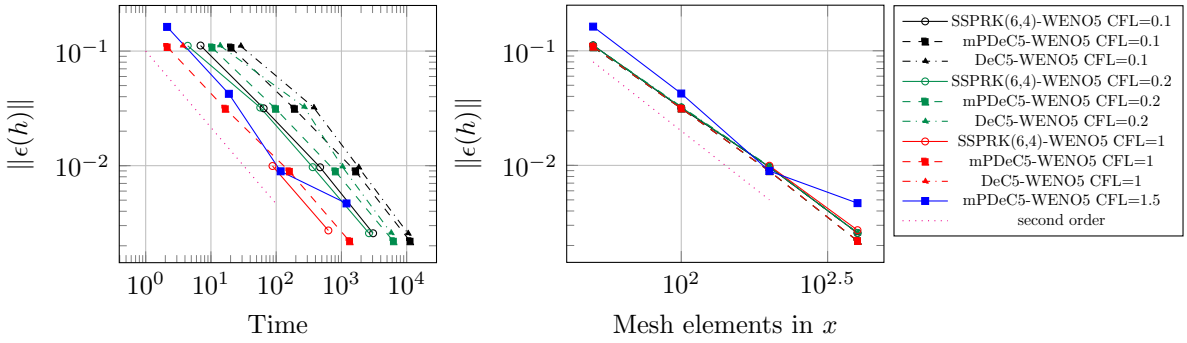


Figure 9: Wet and dry lake at rest test: error and computational time.

on the domain $[-5,5]\times[-2,2]$. The lake at rest solution is

$$h(x,y)=\max\{0.7-b(x,y),0\},\qquad u=v=0. \tag{40}$$

The maximum of the bathymetry is 1, hence, there is a dry island at the center of the domain. For practical purposed, we set the initial conditions to be

$$h_0(x,y)=\max\{0.7-b(x,y),\varepsilon\},\qquad u=v=0, \tag{41}$$

with $\varepsilon=10^{-6}$. We consider final time $T=1$.

First, we test the non-well-balanced schemes, to asses the capability of preserving the water height positivity and the accuracy of such methods. The positivity of the classical schemes is not preserved, even with the positivity limiter of remark 3.1. In the dry region, the SW model (2) does not hold and for the time integration schemes it is hard to verify hypotheses that guarantee the positivity of the solution. Hence, for the classical schemes, we force the positivity of the water height every time we need to compute the flux or to convert the variables from conservative to primitive and *vice versa*. On the other hand, the mPDeC scheme always preserve the positivity of the solution and none of these tricks is required.

In figure 9 we observe that for similar computational times, the mPDeC5-WENO5 gives the same accuracy of all classical schemes. For all schemes the accuracy is 2, as the solution is not $\mathcal{C}^1$ everywhere. The difference is that mPDeC5-WENO5 can be run up to CFL=1, without any problem, while for CFL=1 the

Table 1: Almost dry lake at rest: mPDeC5-WENO5

N_x	Error h	Order h	Error u	Order u	Error v	Order v
600	8.346e-05	3.899	7.759e-04	3.057	6.911e-04	3.091
700	4.166e-05	4.508	4.526e-04	3.498	3.648e-04	4.144
800	2.134e-05	5.007	2.533e-04	4.346	1.886e-04	4.941
1000	6.185e-06	5.551	7.406e-05	5.511	5.225e-05	5.753
1200	2.219e-06	5.621	2.478e-05	6.006	1.774e-05	5.924
1400	1.120e-06	4.438	1.034e-05	5.669	7.561e-06	5.532
1600	5.896e-07	4.804	5.200e-06	5.148	3.799e-06	5.155

Table 2: Almost dry lake at rest: SSPRK(6,4)-WENO5

N_x	Error h	Order h	Error u	Order u	Error v	Order v
600	8.378e-05	3.891	7.765e-04	3.057	6.907e-04	3.091
700	4.191e-05	4.493	4.532e-04	3.493	3.644e-04	4.147
800	2.160e-05	4.963	2.538e-04	4.342	1.884e-04	4.942
1000	6.426e-06	5.434	7.432e-05	5.504	5.228e-05	5.745
1200	2.569e-06	5.029	2.502e-05	5.972	1.797e-05	5.857
1400	1.376e-06	4.051	1.058e-05	5.586	7.805e-06	5.410
1600	9.062e-07	3.127	5.496e-06	4.901	4.096e-06	4.829

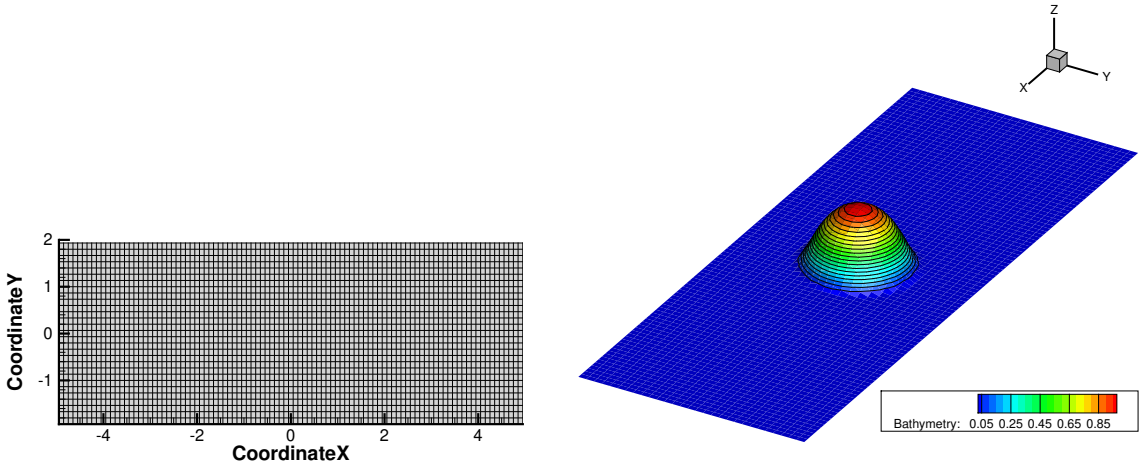
SSPRK(6,4)-WENO5 scheme, even with the checks on the positivity, can have problems and might have exploding velocities and water heights. Since the reconstruction does not guarantee the positivity for such high CFLs, it is not safe to run such simulations.
Adding the well-balanced technique we obtain machine precision errors for all schemes.

6.4 Almost dry lake at rest

Now we modify the previous test, in order to have a smooth solution and to be able to obtain a fifth order accuracy in the schemes. We consider again the bathymetry (39) on the domain $[-5,5]\times[-2,2]$. The lake at rest solution is defined, this time, as

$$h(x,y) = \max\{0.999 - b(x,y), 0\}, \qquad u = v = 0. \tag{42}$$

Notice that the bathymetry has a peak with value 1, but, depending on the mesh refinement, the peak will be lower. In most of the simulations, this test will be completely wet and $\mathcal{C}^\infty$. If we discretize the mesh with more than 600×180 elements, the water level will be below the threshold $\varepsilon = 10^{-6}$. As before, we initialize the water heigth at least equal to $\varepsilon = 10^{-6}$ and we let the schemes evolve up to final time $T = 1$.
We compare the mPDeC5-WENO5 and the SSPRK(6,4)-WENO5 schemes. The test is steady, hence, we expect only the spatial discretization to be the only responsible of the order of accuracy. For the SSPRK(6,4)-WENO5 to be run, we need to introduce some extra checks on the water height and computations of the flux, so that it does not become negative, while for the mPDeC5 time integration we do not need this type of extra corrections. In both cases we expect to have a small perturbation of the solution while computing the L_1 error, indeed, the initial condition set with minimum level at 10^{-6} should introduce an error, in very few cells, of this order.
In table 1 there is the error analysis for the mPDeC5-WENO5 method, while in table 2 there is the one referred to the SSPRK(6,4)-WENO5 method. Despite expecting the error of the initialization to become evident around error of 10^{-6}, for the mPDeC5 simulation, this small perturbation confined to very few cells does not propagate much and lead to very accurate results also for errors $\approx 5\cdot 10^{-7}$. Even the correction in remark 4.5 does not seem to affect the accuracy of the solution, probably because the water height never reaches values much lower than 10^{-6}. So, the order of accuracy stays very close to five, the expected one, even for almost dry solutions.
On the other side, in the SSPRK simulation the need of extra corrections in the flux every time the solution



(a) Computational mesh

(b) Bathymetry

Figure 10: Perturbation analysis over a steady solution: test case setting

is below 10^{-6} adds further errors that are visible at level of L_1 error around 10^{-6}. Indeed, it seems that the error of the water height starts plateauing close to that value. And there it loses the expected fifth order of accuracy.

6.5 Perturbation of the lake at rest

In order to better highlight the improvements one gets with the well-balanced implementation, a perturbation analysis is run on a problem with both wet and dry areas. This test case is run on the rectangular domain $[-5,5]\times[-2,2]$. The bathymetry is given by (39). The lake at rest solution that we are going to conserve with the well-balanced implementation is

$$h(x,y) = \max\{0.7 - b(x,y), \varepsilon\}, \qquad u = v = 0, \tag{43}$$

with $\varepsilon = 10^{-6}$, whereas the perturbation shape that we want to study is

$$\tilde{h}(x,y) = h(x,y) + \begin{cases} 0.05\, e^{1-\frac{1}{(1-\rho^2)^2}}, & \text{if } \rho^2 < 1, \\ 0, & \text{else}, \end{cases} \qquad \text{where } \rho^2 = 9((x+2)^2 + (x-0.5)^2). \tag{44}$$

Two simulations have been performed: one with the well-balanced correction and one without. Both simulations use WENO5 as spatial discretization and mPDeC5 for integrating the ODEs coming from the semi-discrete system. From this test on the tolerances of the positivity limiter and of the mPDeC5 divisions is set to $\varepsilon = 10^{-6}$. The computational mesh and bathymetry plot are shown in Figure 10.
The results obtained for the two implementations are displayed in Figure 11 where only the isolines of the water height h are shown. Four snapshots are presented at different times of the simulation, $t = 0, 0.25, 0.5, 1$. The results on the right-hand side of Figure 11 are those computed without the well-balancing correction. As can be noticed, in this case, the numerical error propagates from the nonconstant bathymetry area around the island placed in the middle of the domain. This error propagates and interacts with the perturbation, making the perturbation waves indistinguishable from the noise. This test case allows to better assess the well-balanced implementation already tested in the previous test case. Indeed, for this case, we have a dry area which involves a jump in the derivative of the water height causing a reduction of the order of accuracy given by the WENO method, whose limiters work with the high order derivatives of the solution. On the

other side, the simulation runs with the well-balanced correction allows to exactly preserve the lake at rest solution over which the perturbation analysis is carried out. This leads to a much better capturing of the perturbation, whose evolution is not influenced by the spurious disturbances coming from the wet-dry area.

6.6 Circular dry dam break problem

We simulate the break of a circular dam separating two basins with water heights $h_1 = 2.5$ and $h_2 = \varepsilon = 10^{-6}$, meaning that the water in the first basin is falling over a dry area which is all around it. The radius of the discontinuity is $r = 7$. A sketch of the initial condition, along with the computational mesh, is given in Figure 12. The computational domain is the square $[0, 40] \times [0, 40]$ discretized with 100×100 cells and the simulation is run until a final time $t_{end} = 0.9$. This is a somewhat challenging test for the mPDeC5–WENO5 method that has to face both the capture of a sharp discontinuity and the progressing wetting of a dry area while always maintaining its appealing properties. The results are printed for different times, $t = 0, 0.3, 0.6, 0.9$, in Figure 13 showing the evolution of the water height. The advantages of the mPDeC5 have been clearly proven by running this challenging test case with different CFL conditions. As a matter of fact, we managed to run this test case with a CFL up until 1.5 while ensuring positivity. In particular, the water height equation does not cause any CFL restriction due to the nature of the method used to solve it, which is unconditionally positive. However, since the momentum equations are solved by means of an explicit DeC scheme, the CFL must be bounded. It should be noticed that, in order to retain positivity in the spatial reconstruction, a positivity limiter has been implemented. For explicit SSPRK methods, this limiter has been proven to cause a huge restriction in the CFL condition, which now has to be less than $\frac{1}{12}\text{CFL}^{\text{SSPRK}}$. Nevertheless, since arbitrary high order DeC cannot be recast as a convex combination of explicit Euler method, there is no proof that the solution would stay positive also under that strict condition. On the other side, in the modified Patankar DeC framework, the limiter is only imposed on the water height equation which is unconditionally positive for any CFL by definition. This means that even when the limiter plays a role in the simulation, like in this case, the scheme stays positive with much higher CFL numbers. Furthermore, it should be underlined the fact that the system is linear implicit, which only requires a simple linear solver, e.g. Jacobi method. Even though implicit methods are much slower than their explicit counterpart, solving a linear problem in this case only yields to a 18% increase in the computational time with respect to the fully explicit DeC5, which is nothing compared to the CFL restriction imposed by the positivity limiter.

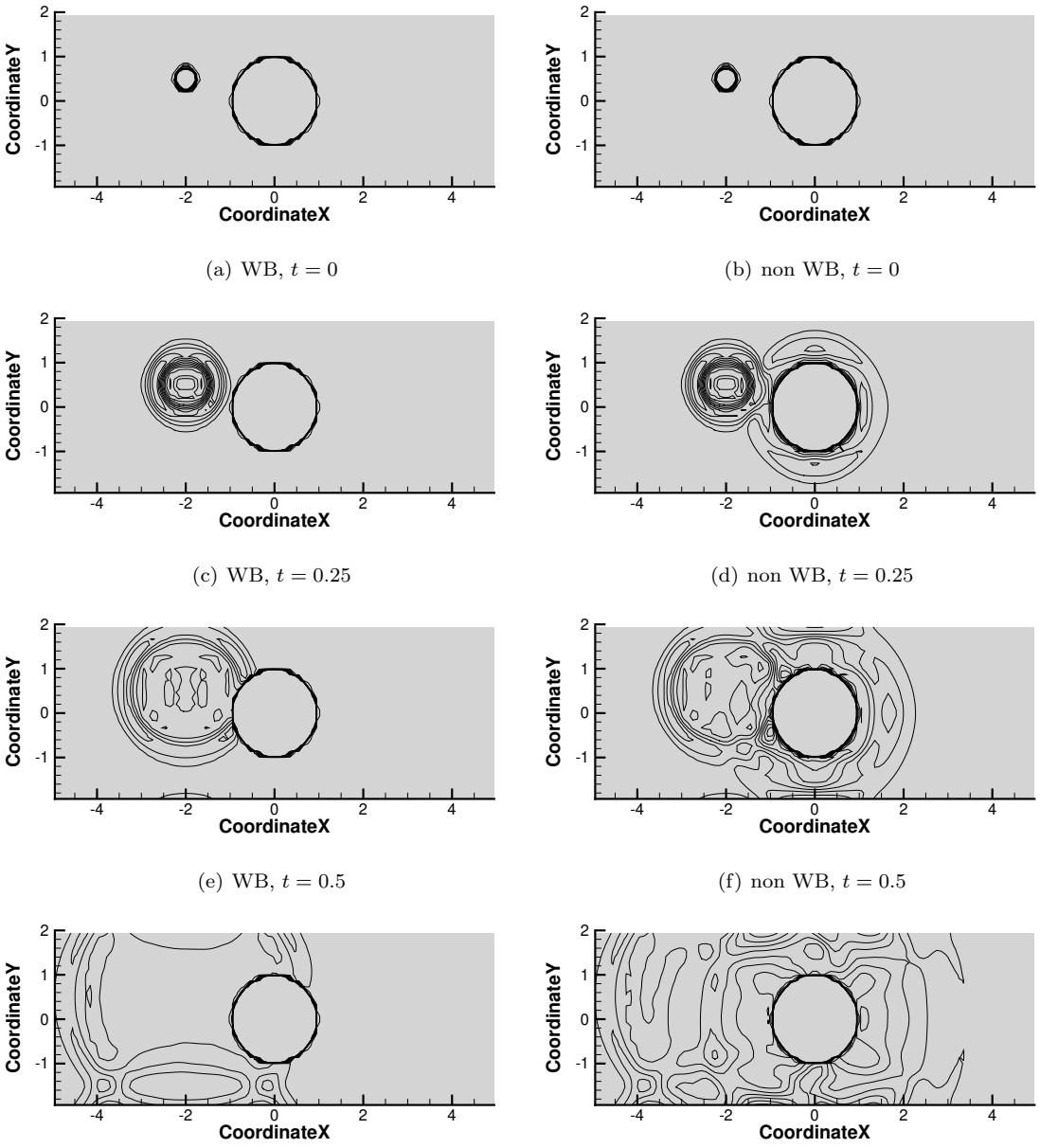
6.7 Circular wet dam break problem

Next, we consider a wet dam break problem with a setup similar to the previous one. In this case the water height of the two basins are $h_1 = 10$ and $h_2 = 0.5$, like it was done in [52]. In this case, the computational domain is the square $[0, 50] \times [0, 50]$ discretized by a 200×200 mesh. The simulation is run until $t_{end} = 0.8$ with CFL=1 and the solutions is displayed only for one quarter of the domain. Figure 14 displays the final snapshot of the solution at time $t_{end} = 0.8$ plotted both in 2D and 3D displaying the correct evolution of the water height and the time evolution of the water depth extracted along the diagonal of the portion of the domain studied. The method shows good properties such as discontinuities sharply captured, no oscillations and high order approximation of smooth features.

6.8 Wave over dry island

For the last test case, we simulate a wave crashing over a dry island showing the robustness of our method when facing more realistic simulations. The computational domain is the rectangle $[-5, 5] \times [-2, 2]$ discretized by a 400×120 mesh. The simulation can be reproduced by taking Eq. (39) for the bathymetry $b(x, y)$ and the following initial conditions

$$h(x,y) = 0.7 - b(x,y) + \begin{cases} 0.5\, e^{1-\frac{1}{(1-\rho^2)^2}}, & \text{if } \rho^2 < 1, \\ 0, & \text{else,} \end{cases} \qquad \text{where } \rho^2 = (x+2)^2, \qquad (u,v) = (1,0), \quad (45)$$



(a) WB, $t = 0$

(b) non WB, $t = 0$

(c) WB, $t = 0.25$

(d) non WB, $t = 0.25$

(e) WB, $t = 0.5$

(f) non WB, $t = 0.5$

(g) WB, $t = 1$

(h) non WB, $t = 1$

Figure 11: Perturbation analysis over a steady solution: water height h isolines for well-balanced (left-hand side) and non well-balanced (right-hand side) results.

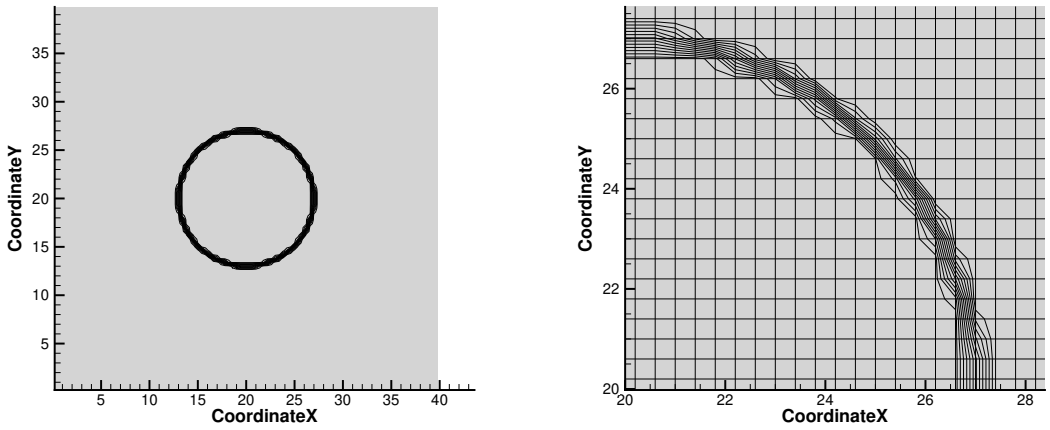
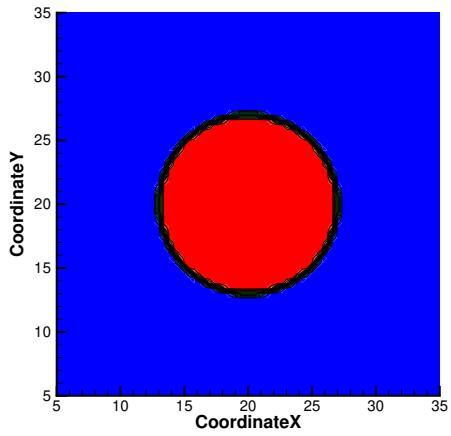


Figure 12: Circular dry dam break: computational domain and initial solution.

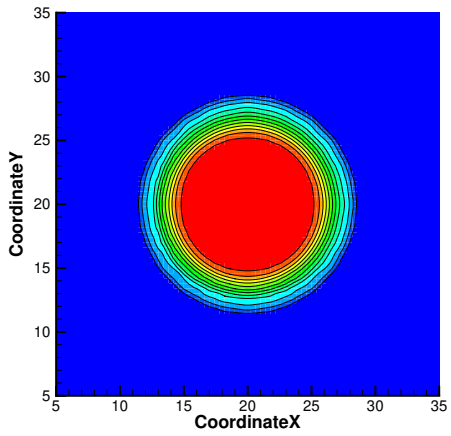
in case $h(x,y) \leq \varepsilon = 10^{-6}$, we set $h(x,y) = \varepsilon$ and $u = 0$. The simulation has been run with the WENO5−mPDeC5 scheme until a final time $t = 1$ with CFL=0.9 and the results are displayed in Figure 16 for different times $t = 0, 0.25, 0.5, 0.75, 1$.
In this case, the variable $\eta = h + b$ has been chosen to be plotted since it better represents the underlying physics. It should be noticed that the top of the island, which is dry at the beginning, gets wet and dry several times during the simulation while never giving rise to problems due to negative water height. This is instead something that we cannot ensure for the DeC5 case. The simulation starts with a background state moving with speed $u = 1$ which helps the wave traveling towards the island and immediately starts wetting the island from the left and drying it on the right. The wave breaks into two smaller waves respectively approaching and moving away from the island following the eigenvalues of the flux Jacobian. At time $t = 0.25$ a run up is happening where the traveling wave is trying to submerge the top of the island while, on the other side, a section of the island is drying. In consecutive times, the wave overtakes the island causing different interacting shock fronts and two symmetrical minimum points highlighted in dark blue at time $t = 1$. Several structures could been observed in this simulations and the repeated wetting/drying procedures never results in troubles for the mPDeC method.
In Figure 15 we depict the average number of Jacobi iteration needed for every linear system. The plot compares different CFL numbers and mesh refinements which keep the aspect ratio. The average is plotted with a confidence interval given by $\pm\frac{1}{2}$ standard deviation. We observe that the CFL number is very incisive in determining how many Jacobi iterations are needed. For low CFL it seems that larger matrices needs more iterations, but this is not uniform with all the tests, as shown in the unsteady vortex test.

7 Summary and Outlook

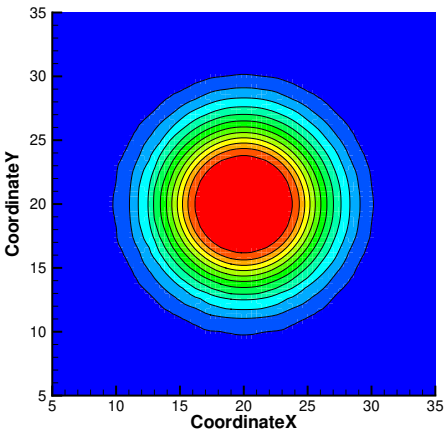
We have presented a new well-balanced, positivity preserving high order numerical method for solving the shallow water equations. By re-writing the WENO semi-discretization in terms of productions-destructions terms, we were able to use the modified Patankar Deferred Correction methods of arbitrarily high-order to ensure the unconditionally positivity of water height. The restriction on the CFL number comes only from the explicit DeC-solver for the momentum equations. The used CFL numbers are of the order of 1 and are much larger than the ones used in explicit SSPRK WENO methods in combination with positivity preserving limiters [65], where the CFL must be lowered to 1/6 or 1/12. One can relax further the CFL constraint by using implicit DeC or RK methods, though introducing more difficulties. Even if we have only presented



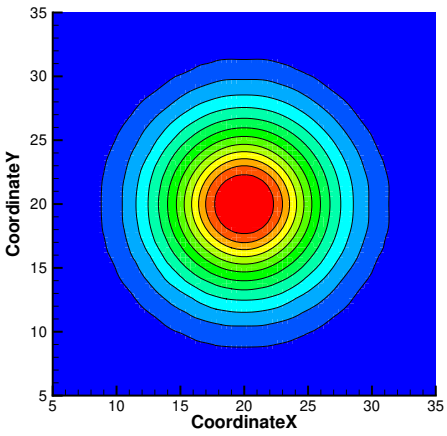
(a) $t = 0$



(b) $t = 0.3$

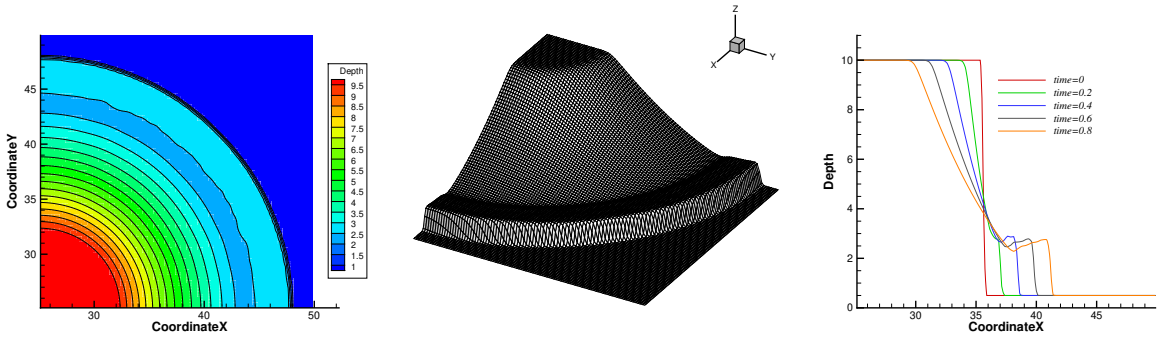


(c) $t = 0.6$



(d) $t = 0.9$

Figure 13: Circular dry dam break: water height h isocontours.



(a) Isocontours of h at time $t_{end} = 0.8$ (b) Plot of h at time $t_{end} = 0.8$ (c) h time evolution along the diagonal of the domain

Figure 14: Circular wet dam break

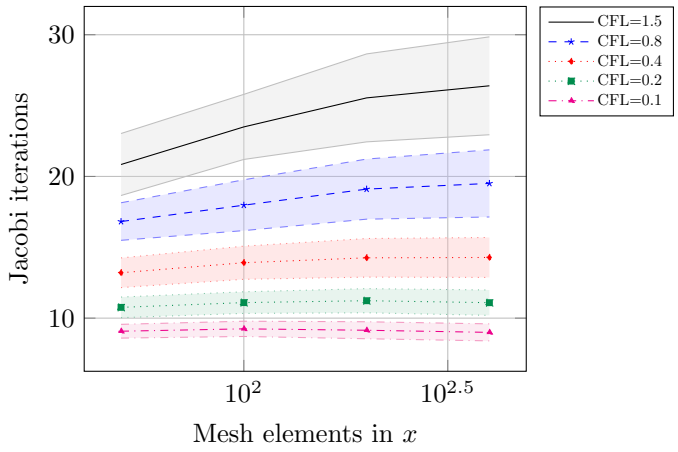
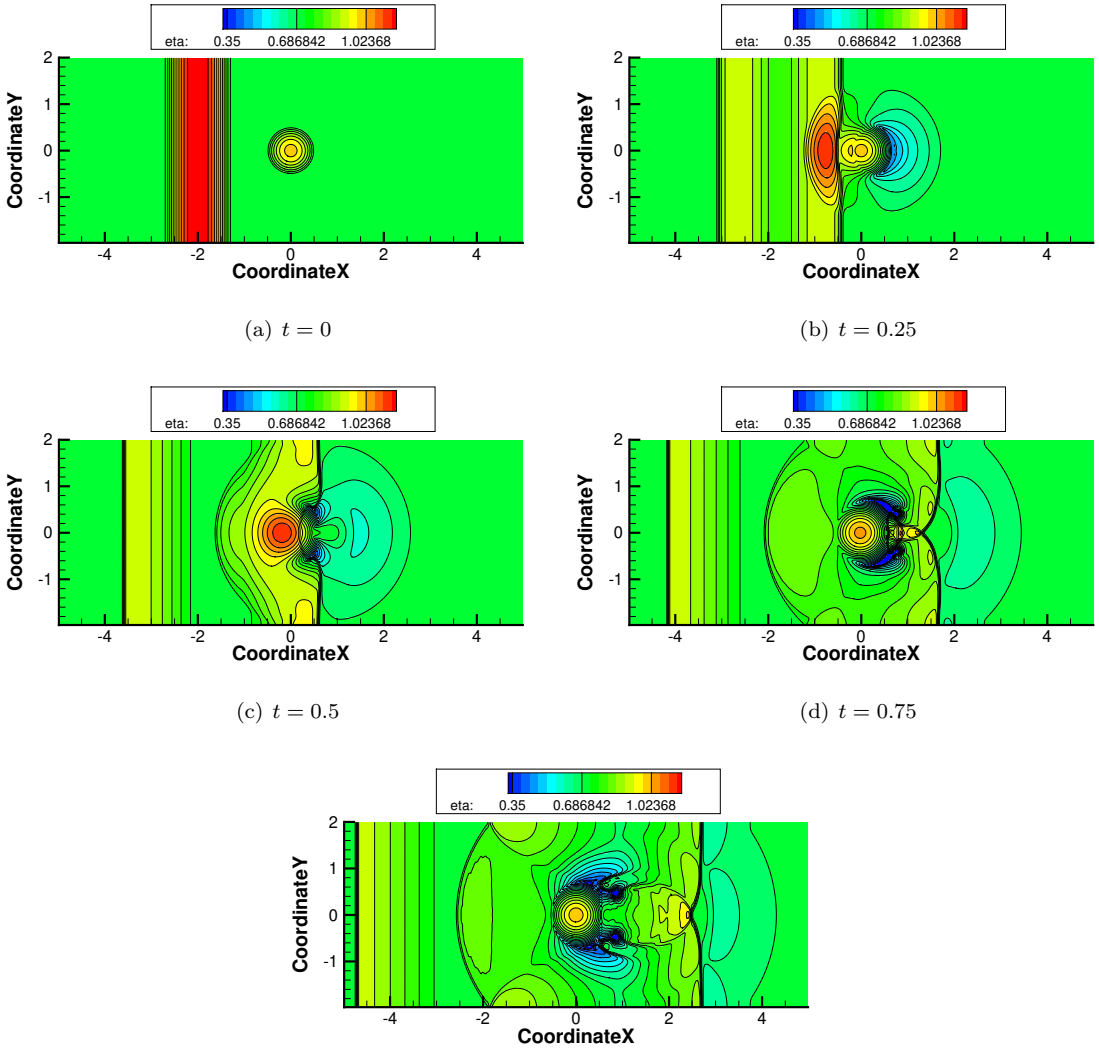


Figure 15: Wave over dry island: Jacobi iterations and confidence interval.



(a) $t = 0$

(b) $t = 0.25$

(c) $t = 0.5$

(d) $t = 0.75$

(e) $t = 1$

Figure 16: Wave over dry island: $\eta = h + b$ isocontours at different times.

a fifth order method in this manuscript, the approach is actually of arbitrary high-order. With classical explicit approaches, one must use SSPRK to guarantee positivity and, as known from literature [27], explicit (implicit) SSPRK methods exist only up to order four (sixth) for general cases.
By applying mPDeC, we avoid those issues and the price to pay is that of solving a (very sparse) linear system for the water height. However, as mentioned before, this increase in computational costs is around 18% percent in our numerical simulations, but the procedure can still be optimized. In the future, a detailed performance test in terms of accuracy and run-time is planned.
Additionally, in this work we have only considered positivity preservation and well-balanced properties. In the next step, we would like to extend our investigation to entropy conservative/stable methods. Here, various approaches exist as described *inter alia* in [2, 4, 14, 22, 23, 49], but from our perspective the convex limiting strategies seems the most promising one [28, 35]. In order to do so, the basic stability properties of Patankar methods as ODE solvers have to be first fully understood [32, 61].
Finally, herein we have focused on the shallow water system. However, the method can be easily adapted to more complex models, e.g. the shallow water equations together with biochemical processes like algae bloom in oceans, seas and open water cancels or the Euler equations of gas dynamics, where special treatments for the pressure positivity are necessary [67, 64]. Those extensions can and will be also considered in future works.

A Reconstruction of the primitive variable

The goal of this section is that of deepening the procedure to compute the coeffiecients of the polynomials and linear weights needed to compute the WENO procedure in Section 3. The results and the actual coefficients for WENO5 with 4 Gaussian quadrature points are written in Section B.
In the following assemble the system that allow to find the coefficients for the WENO polynomial at any quadrature point. The symbolic Matlab script coded for this purpose is also available at [18]. With few adjustments, the script can be used to compute all the ingredients needed for a WENO reconstruction of arbitrary high order with arbitrary high order quadrature formulae.
The reconstruction of the primitive variables is needed in a high order finite volume method, as only cell averages are available from the previous time step. For the sake of simplicity we are going to work in a simpler one-dimensional scalar framework since the reconstruction is done dimension-by-dimension. We consider a scalar function $u(x)$ whose cell averages u_i are known. We aim at reconstructing this variable as a polynomial $v(x)$, where the polynomial may vary in different points. In particular, it will be useful to use a primitive of $v(x)$ which we denote by $\mathcal{P}(x) = \int_{x_0}^{x} v(s)\mathrm{d}s$. Indeed, we can impose that for every cell average, we have

$$u_i = \frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} u(x)\ \mathrm{d}x \overset{!}{=} \int_{x_{i-1/2}}^{x_{i+1/2}} v(x)\mathrm{d}x = \frac{\mathcal{P}(x_{i+1/2}) - \mathcal{P}(x_{i-1/2})}{\Delta x}. \tag{46}$$

The stencil considered for this example is the one used for the WENO5 reconstruction, which is made up by five cell averages as denoted in Figure 17.

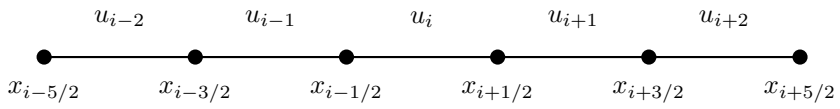


Figure 17: Stencil of five cell averages for WENO5 reconstruction.

The next step consists in using interpolating polynomials $\varphi_{j-1/2}$, e.g. Lagrange polynomials at cell interfaces, to approximate the primitive $\mathcal{P}(x)$, i.e.,

$$\mathcal{P}(x) = \sum_{j=k_1}^{k_2+1} a_{j-1/2}\,\varphi_{j-1/2}(x) \qquad \text{where} \qquad \varphi_{j-1/2}(x) = \prod_{\ell=k_1, i\neq j}^{k_2+1} \frac{x - x_{\ell-1/2}}{x_{j-1/2} - x_{\ell-1/2}} \tag{47}$$

where k_1 and k_2 are the extreme indexes of the considered stencil. So, given then k_2-k_1+1 cell averages, the degree of the polynomial $\mathcal{P}$ will be k_2-k_1+2. For instance, when working with WENO5, the reconstruction is composed by a linear combination of three lower order polynomials of degree 3, so with $\mathcal{P}\in\mathbb{P}_4$, obtained through three cell averages. Then, the 3 polynomials, which then depend on the whole 5 cells stencil, will be be combined with weights which depends on the quadrature points into a fifth order accurate reconstruction. In order to have this result, we need to compute the aforementioned three lower order polynomials and a high order polynomial made up by information coming from the whole stencil.
Let us begin with the procedure to compute the high order polynomial with all available cell averages, i.e., $k_1=-2$ and $k_2=2$. The lower order polynomials can be easily computed following the same approach explained hereafter. In this case $\mathcal{P}(x)$ is a sixth order polynomial and $v(x)=\mathcal{P}'(x)$ is a fifth order polynomial which gives the right accuracy order. Using (46) for all cell averages in the stencil with the definition of $\mathcal{P}$ given in (47) and using the property of Lagrangian polynomials $\varphi_{j-1/2}(x_{\ell-1/2})=\delta_{j,\ell}$, we obtain a system of equations for the coefficients $a_{j-1/2}$ with solution

$$a_{j-1/2}=\begin{cases}0, & \text{if } j=k_1,\\ \sum_{\ell=-k_1}^{-k_1+j-1} u_{i+\ell}, & j=k_1+1,\dots,k_2+1.\end{cases} \tag{48}$$

Finally, the expression for the high order, *ho*, approximation polynomial $v^{ho}(x)$ can be written as

$$v^{ho}(x)=\sum_{j=k_1}^{k_2+1} a_{j-1/2}\varphi'_{j-1/2}(x)=\sum_{\ell=-k_1}^{k_2} c_\ell^{ho}(x)\,u_{i+\ell}=\sum_{\ell=-2}^{2} c_\ell^{ho}(x)\,u_{i+\ell}, \tag{49}$$

where c_ℓ^{ho} are obtained collecting all the coefficients and basis functions related to $u_{i+\ell}$. In this way, for any quadrature point $\tilde{\xi}\in[x_{i-1/2},x_{i+1/2}]$ we can evaluate this high order polynomial $v^{ho}(\tilde{\xi})$. Following the same procedure for three lower order polynomials, *lo*, associated to the 3-cells stencils $S^0=\{u_i,u_{i+1},u_{i+2}\}$, $S^1=\{u_{i-1},u_i,u_{i+1}\}$ and $S^2=\{u_{i-2},u_{i-1},u_i\}$, we obtain an expression for these low order polynomials

$$v_j^{lo}(x)=\sum_{\ell=-j}^{2-j} c_{j\ell}^{lo}(x)\,u_{i+\ell}. \tag{50}$$

The last step concerns the computation of the ideal linear weights. These are the weights that allow to recover the high order reconstruction from a linear combination of the lower order ones for a given quadrature point $\tilde{\xi}$. Therefore, we need to find the linear weights d_j such that

$$v^{ho}(\tilde{\xi})=\sum_{j=0}^{2} d_j\,v_j^{lo}(\tilde{\xi}) \quad\Longleftrightarrow\quad \sum_{\ell=-2}^{2} c_\ell^{ho}(\tilde{\xi})\,u_{i+\ell}=\sum_{j=0}^{2}\sum_{\ell=-j}^{2-j} d_j\,c_{j\ell}^{lo}(\tilde{\xi})\,u_{i+\ell},\quad \forall u_{i+\ell}. \tag{51}$$

Since (51) must hold for any quintuplet $\{u_{i+\ell}\}_{\ell=-2}^{2}$, we can write a system of five equations in the 3 linear weights d_j for each quadrature point $\tilde{\xi}$, i.e.,

$$\sum_{j=0}^{2} d_j\,c_{j\ell}^{lo}(\tilde{\xi})=c_\ell^{ho}(\tilde{\xi}),\quad \forall \ell=-2,\dots,2. \tag{52}$$

This is an overdetermined system with five equations and only three unknowns d_j that can be easily solved by means of a least squares method. Moreover, the solutions found verify exactly all the equations, implying that some of the equations are linearly dependent.

B WENO recontruction ($r=3$) with four-point Gaussian quadrature rule

The goal of this section is to present the fifth order WENO reconstruction with four-point Gaussian quadrature rule. Up to our knowledge, there is no reference in literature that explicitly define linear weights and

polynomial coefficients needed for this WENO reconstruction. Reference [59] well described the fifth-order WENO5 with two-point Gaussian quadrature rule, which unfortunately does not allow to go beyond fourth order. Instead, with the four-point Gaussian quadrature rule, one could reach even eighth order.
Let us consider a one-dimensional cell $[\xi_{i-1/2}, \xi_{i+1/2}]$, we hereafter provide the expressions for

$$q(\xi^-_{i+1/2})\ ,\ \ q(\xi^+_{i-1/2})\ ,\ \ q\left(\xi_i \pm \frac{\Delta\xi}{2}\sqrt{\frac{3}{7}+\frac{2}{7}\sqrt{\frac{6}{5}}}\right)\ ,\ \ q\left(\xi_i \pm \frac{\Delta\xi}{2}\sqrt{\frac{3}{7}-\frac{2}{7}\sqrt{\frac{6}{5}}}\right), \tag{53}$$

which are used for the first sweep (first two terms corresponding to the two boundaries) and the second sweep (the last two terms corresponding to the 4 quadrature points). For $r = 3$ we have only three candidate stencil for the reconstruction

$$S_0 = (i, i+1, i+2)\ ,\quad S_1 = (i-1, i, i+1)\ ,\quad S_2 = (i-2, i-1, i). \tag{54}$$

The corresponding smoothness indicators are given by:

$$\begin{aligned}
\beta_0 &= \frac{13}{12}\left(q_i - 2q_{i+1} + q_{i+2}\right)^2 + \frac{1}{4}\left(3q_i - 4q_{i+1} + q_{i+2}\right)^2,\\
\beta_1 &= \frac{13}{12}\left(q_{i-1} - 2q_i + q_{i+1}\right)^2 + \frac{1}{4}\left(q_{i-1} - q_{i+1}\right)^2,\\
\beta_2 &= \frac{13}{12}\left(q_{i-2} - 2q_{i-1} + q_i\right)^2 + \frac{1}{4}\left(q_{i-2} - 4q_{i-1} + 3q_i\right)^2.
\end{aligned}$$

The optimal weights d_m for the left boundary extrapolated value $q^-_{i+1/2}$ at $x_{i+1/2}$ are

$$d_0 = \frac{3}{10}\ ,\quad d_1 = \frac{3}{5}\ ,\quad d_2 = \frac{1}{10} \tag{55}$$

and $q^-_{i+1/2}$ is given by

$$q^-_{i+1/2} = \frac{1}{6}\omega_0\left(-q_{i+2} + 5q_{i+1} + 2q_i\right) + \frac{1}{6}\omega_1\left(-q_{i-1} + 5q_i + 2q_{i+1}\right) + \frac{1}{6}\omega_2\left(2q_{i-2} - 7q_{i-1} + 11q_i\right). \tag{56}$$

The optimal weights d_m for the right boundary extrapolated value $q^+_{i-1/2}$ at $x_{i-1/2}$ are

$$d_0 = \frac{1}{10}\ ,\quad d_1 = \frac{3}{5}\ ,\quad d_2 = \frac{3}{10} \tag{57}$$

and $q^+_{i-1/2}$ is given by

$$q^+_{i-1/2} = \frac{1}{6}\omega_0\left(2q_{i+2} - 7q_{i+1} + 11q_i\right) + \frac{1}{6}\omega_1\left(-q_{i+1} + 5q_i + 2q_{i-1}\right) + \frac{1}{6}\omega_2\left(-q_{i-2} + 5q_{i-1} + 2q_i\right). \tag{58}$$

For the first Gaussian quadrature point $\xi^q_1 = \xi_i - \frac{\Delta\xi}{2}\sqrt{\frac{3}{7}+\frac{2}{7}\sqrt{\frac{6}{5}}}$, the optimal weights are:

$$\begin{aligned}
d_0 &= \frac{269\sqrt{42}\sqrt{2\sqrt{30}+15}}{50428} - \frac{1751\sqrt{35}\sqrt{2\sqrt{30}+15}}{504280} - \frac{411\sqrt{30}}{100856} + \frac{21855}{100856},\\
d_1 &= \frac{411\sqrt{30}}{50428} + \frac{28573}{50428},\\
d_2 &= \frac{1751\sqrt{35}\sqrt{2\sqrt{30}+15}}{504280} - \frac{269\sqrt{42}\sqrt{2\sqrt{30}+15}}{50428} - \frac{411\sqrt{30}}{100856} + \frac{21855}{100856}.
\end{aligned} \tag{59}$$

The reconstructed value can be computed from the three polynomials associated to each stencil:

$$p_0(\xi_1^q) = \left(\frac{\sqrt{5}\sqrt{6}}{140} + \frac{\sqrt[3]{\frac{2\sqrt{5}\sqrt{6}}{35}+\frac{3}{7}}}{4} + \frac{85}{84}\right) q_i + \left(-\frac{\sqrt{5}\sqrt{6}}{70} - \sqrt{\frac{2\sqrt{5}\sqrt{6}}{35}+\frac{3}{7}} - \frac{1}{42}\right) q_{i+1} + \left(\frac{\sqrt{5}\sqrt{6}}{140} + \frac{\sqrt{\frac{2\sqrt{5}\sqrt{6}}{35}+\frac{3}{7}}}{4} + \frac{1}{84}\right) q_{i+2},$$

$$p_1(\xi_1^q) = \left(\frac{\sqrt{5}\sqrt{6}}{140} + \frac{\sqrt{\frac{2\sqrt{5}\sqrt{6}}{35}+\frac{3}{7}}}{4} + \frac{1}{84}\right) q_{i-1} + \left(\frac{41}{42} - \frac{\sqrt{5}\sqrt{6}}{70}\right) q_i + \left(\frac{\sqrt{5}\sqrt{6}}{140} - \frac{\sqrt{\frac{2\sqrt{5}\sqrt{6}}{35}+\frac{3}{7}}}{4} + \frac{1}{84}\right) q_{i+1},$$

$$p_2(\xi_1^q) = \left(\frac{\sqrt{5}\sqrt{6}}{140} - \frac{\sqrt{\frac{2\sqrt{5}\sqrt{6}}{35}+\frac{3}{7}}}{4} + \frac{1}{84}\right) q_{i-2} + \left(\sqrt{\frac{2\sqrt{5}\sqrt{6}}{35}+\frac{3}{7}} - \frac{\sqrt{5}\sqrt{6}}{70} - \frac{1}{42}\right) q_{i-1} + \left(\frac{\sqrt{5}\sqrt{6}}{140} - \frac{\sqrt[3]{\frac{2\sqrt{5}\sqrt{6}}{35}+\frac{3}{7}}}{4} + \frac{85}{84}\right) q_i.$$

For the second Gaussian quadrature point $\xi_2^q = \xi_i - \frac{\Delta\xi}{2}\sqrt{\frac{3}{7} - \frac{2}{7}\sqrt{\frac{6}{5}}}$, the optimal weights are:

$$\begin{aligned} d_0 &= \frac{411\sqrt{30}}{100856} - \frac{269\sqrt{42}\sqrt{15-2\sqrt{30}}}{50428} - \frac{1751\sqrt{35}\sqrt{15-2\sqrt{30}}}{504280} + \frac{21855}{100856}, \\ d_1 &= \frac{28573}{50428} - \frac{411\sqrt{30}}{50428}, \\ d_2 &= \frac{1751\sqrt{35}\sqrt{15-2\sqrt{30}}}{504280} + \frac{269\sqrt{42}\sqrt{15-2\sqrt{30}}}{50428} + \frac{411\sqrt{30}}{100856} + \frac{21855}{100856}. \end{aligned} \tag{60}$$

The reconstructed value can be then computed from the three polynomials associated to the three stencil:

$$p_0(\xi_2^q) = \left(\frac{\sqrt[3]{\frac{3}{7}-\frac{2\sqrt{5}\sqrt{6}}{35}}}{4} - \frac{\sqrt{5}\sqrt{6}}{140} + \frac{85}{84}\right) q_i + \left(\frac{\sqrt{5}\sqrt{6}}{70} - \sqrt{\frac{3}{7}-\frac{2\sqrt{5}\sqrt{6}}{35}} - \frac{1}{42}\right) q_{i+1} + \left(\frac{\sqrt{\frac{3}{7}-\frac{2\sqrt{5}\sqrt{6}}{35}}}{4} - \frac{\sqrt{5}\sqrt{6}}{140} + \frac{1}{84}\right) q_{i+2},$$

$$p_1(\xi_2^q) = \left(\frac{\sqrt{\frac{3}{7}-\frac{2\sqrt{5}\sqrt{6}}{35}}}{4} - \frac{\sqrt{5}\sqrt{6}}{140} + \frac{1}{84}\right) q_{i-1} + \left(\frac{\sqrt{5}\sqrt{6}}{70} + \frac{41}{42}\right) q_i + \left(\frac{1}{84} - \frac{\sqrt{\frac{3}{7}-\frac{2\sqrt{5}\sqrt{6}}{35}}}{4} - \frac{\sqrt{5}\sqrt{6}}{140}\right) q_{i+1},$$

$$p_2(\xi_2^q) = \left(\frac{1}{84} - \frac{\sqrt{\frac{3}{7}-\frac{2\sqrt{5}\sqrt{6}}{35}}}{4} - \frac{\sqrt{5}\sqrt{6}}{140}\right) q_{i-2} + \left(\frac{\sqrt{5}\sqrt{6}}{70} + \sqrt{\frac{3}{7}-\frac{2\sqrt{5}\sqrt{6}}{35}} - \frac{1}{42}\right) q_{i-1} + \left(\frac{85}{84} - \frac{\sqrt[3]{\frac{3}{7}-\frac{2\sqrt{5}\sqrt{6}}{35}}}{4} - \frac{\sqrt{5}\sqrt{6}}{140}\right) q_i.$$

For the third Gaussian quadrature point $\xi_3^q = \xi_i + \frac{\Delta\xi}{2}\sqrt{\frac{3}{7} - \frac{2}{7}\sqrt{\frac{6}{5}}}$, the optimal weights are:

$$\begin{aligned} d_0 &= \frac{1751\sqrt{35}\sqrt{15-2\sqrt{30}}}{504280} + \frac{269\sqrt{42}\sqrt{15-2\sqrt{30}}}{50428} + \frac{411\sqrt{30}}{100856} + \frac{21855}{100856}, \\ d_1 &= \frac{28573}{50428} - \frac{411\sqrt{30}}{50428}, \\ d_2 &= \frac{411\sqrt{30}}{100856} - \frac{269\sqrt{42}\sqrt{15-2\sqrt{30}}}{50428} - \frac{1751\sqrt{35}\sqrt{15-2\sqrt{30}}}{504280} + \frac{21855}{100856}. \end{aligned} \tag{61}$$

The reconstructed value can be then computed from the three polynomials associated to the three stencil:

$$p_0(\xi_3^q) = \left(\frac{85}{84} - \frac{\sqrt[3]{\frac{3}{7}-\frac{2\sqrt{5}\sqrt{6}}{35}}}{4} - \frac{\sqrt{5}\sqrt{6}}{140}\right) q_i + \left(\frac{\sqrt{5}\sqrt{6}}{70} + \sqrt{\frac{3}{7}-\frac{2\sqrt{5}\sqrt{6}}{35}} - \frac{1}{42}\right) q_{i+1} + \left(\frac{1}{84} - \frac{\sqrt{\frac{3}{7}-\frac{2\sqrt{5}\sqrt{6}}{35}}}{4} - \frac{\sqrt{5}\sqrt{6}}{140}\right) q_{i+2},$$

$$p_1(\xi_3^q) = \left(\frac{1}{84} - \frac{\sqrt{\frac{3}{7}-\frac{2\sqrt{5}\sqrt{6}}{35}}}{4} - \frac{\sqrt{5}\sqrt{6}}{140}\right) q_{i-1} + \left(\frac{\sqrt{5}\sqrt{6}}{70} + \frac{41}{42}\right) q_i + \left(\frac{\sqrt{\frac{3}{7}-\frac{2\sqrt{5}\sqrt{6}}{35}}}{4} - \frac{\sqrt{5}\sqrt{6}}{140} + \frac{1}{84}\right) q_{i+1},$$

$$p_2(\xi_3^q) = \left(\frac{\sqrt{\frac{3}{7}-\frac{2\sqrt{5}\sqrt{6}}{35}}}{4} - \frac{\sqrt{5}\sqrt{6}}{140} + \frac{1}{84}\right) q_{i-2} + \left(\frac{\sqrt{5}\sqrt{6}}{70} - \sqrt{\frac{3}{7}-\frac{2\sqrt{5}\sqrt{6}}{35}} - \frac{1}{42}\right) q_{i-1} + \left(\frac{\sqrt[3]{\frac{3}{7}-\frac{2\sqrt{5}\sqrt{6}}{35}}}{4} - \frac{\sqrt{5}\sqrt{6}}{140} + \frac{85}{84}\right) q_i.$$

For the fourth Gaussian quadrature point $\xi_4^q = \xi_i + \frac{\Delta\xi}{2}\sqrt{\frac{3}{7}+\frac{2}{7}\sqrt{\frac{6}{5}}}$, the optimal weights are:

$$
\begin{aligned}
d_0 &= \frac{1751\sqrt{35}\sqrt{2\sqrt{30}+15}}{504280} - \frac{269\sqrt{42}\sqrt{2\sqrt{30}+15}}{50428} - \frac{411\sqrt{30}}{100856} + \frac{21855}{100856},\\
d_1 &= \frac{411\sqrt{30}}{50428} + \frac{28573}{50428},\\
d_2 &= \frac{269\sqrt{42}\sqrt{2\sqrt{30}+15}}{50428} - \frac{1751\sqrt{35}\sqrt{2\sqrt{30}+15}}{504280} - \frac{411\sqrt{30}}{100856} + \frac{21855}{100856}.
\end{aligned}
\tag{62}
$$

The reconstructed value can be then computed from the three polynomials associated to the three stencil:

$$
\begin{aligned}
p_0(\xi_4^q) &= \left(\frac{\sqrt{5}\sqrt{6}}{140} - \frac{3\sqrt{\frac{2\sqrt{5}\sqrt{6}}{35}+\frac{3}{7}}}{4} + \frac{85}{84}\right) q_i + \left(\sqrt{\frac{2\sqrt{5}\sqrt{6}}{35}+\frac{3}{7}} - \frac{\sqrt{5}\sqrt{6}}{70} - \frac{1}{42}\right) q_{i+1} + \left(\frac{\sqrt{5}\sqrt{6}}{140} - \frac{\sqrt{\frac{2\sqrt{5}\sqrt{6}}{35}+\frac{3}{7}}}{4} + \frac{1}{84}\right) q_{i+2},\\
p_1(\xi_4^q) &= \left(\frac{\sqrt{5}\sqrt{6}}{140} - \frac{\sqrt{\frac{2\sqrt{5}\sqrt{6}}{35}+\frac{3}{7}}}{4} + \frac{1}{84}\right) q_{i-1} + \left(\frac{41}{42} - \frac{\sqrt{5}\sqrt{6}}{70}\right) q_i + \left(\frac{\sqrt{5}\sqrt{6}}{140} + \frac{\sqrt{\frac{2\sqrt{5}\sqrt{6}}{35}+\frac{3}{7}}}{4} + \frac{1}{84}\right) q_{i+1},\\
p_2(\xi_4^q) &= \left(\frac{\sqrt{5}\sqrt{6}}{140} + \frac{\sqrt{\frac{2\sqrt{5}\sqrt{6}}{35}+\frac{3}{7}}}{4} + \frac{1}{84}\right) q_{i-2} + \left(-\frac{\sqrt{5}\sqrt{6}}{70} - \sqrt{\frac{2\sqrt{5}\sqrt{6}}{35}+\frac{3}{7}} - \frac{1}{42}\right) q_{i-1} + \left(\frac{\sqrt{5}\sqrt{6}}{140} + \frac{3\sqrt{\frac{2\sqrt{5}\sqrt{6}}{35}+\frac{3}{7}}}{4} + \frac{85}{84}\right) q_i.
\end{aligned}
$$

For all Gaussian quadrature points the solution in ξ can be easily built by assembling the three polynomials.

Acknowledgements

M. Ciallella is funded by an Inria PhD fellowship. L. Micalizzi is supported by SNF, project number 200020_175784. D. Torlo is funded by an Inria Postdoc. P. Öffner gratefully acknowledge the support of the Gutenberg Research College and also wants to thank Mario Ricchiuto for his invitation to Inria Bordeaux.
All authors would like to thank Jonatan Núñez for sharing his high-order FV-WENO code on his repository [43]. We have started our work by adapting his code.

References

[1] R. Abgrall. High order schemes for hyperbolic problems using globally continuous approximation and avoiding mass matrices. *Journal of Scientific Computing*, 73(2-3):461–494, 2017.

[2] R. Abgrall. A general framework to construct schemes satisfying additional conservation relations. application to entropy conservative and entropy dissipative schemes. *Journal of Computational Physics*, 372:640–666, 2018.

[3] R. Abgrall, E. L. Mélédo, P. Öffner, and D. Torlo. Relaxation deferred correction methods and their applications to residual distribution schemes. *arXiv preprint arXiv:2106.05005*, 2021.

[4] R. Abgrall, J. Nordström, P. Öffner, and S. Tokareva. Analysis of the SBP-SAT stabilization for finite element methods part II: entropy stability. *Communications on Applied Mathematics and Computation*, pages 1–23, 2021.

[5] R. Abgrall, P. Öffner, and H. Ranocha. Reinterpretation and extension of entropy correction terms for residual distribution and discontinuous Galerkin schemes. *arXiv:1908.04556*, 2019.

[6] R. Abgrall and D. Torlo. High order asymptotic preserving deferred correction implicit-explicit schemes for kinetic models. *SIAM Journal on Scientific Computing*, 42(3):B816–B845, 2020.

[7] F. Bassi and S. Rebay. A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier–Stokes equations. *Journal of Computational Physics*, 131(2):267–279, 1997.

[8] J. P. Berberich, P. Chandrashekar, and C. Klingenberg. High order well-balanced finite volume methods for multi-dimensional systems of hyperbolic balance laws. *Computers & Fluids*, 219:104858, 2021.

[9] C. Berthon and C. Chalons. A fully well-balanced, positive and entropy-satisfying Godunov-type method for the shallow-water equations. *Mathematics of Computation*, 85(299):1281–1307, 2016.

[10] A. Bollermann, G. Chen, A. Kurganov, and S. Noelle. A well-balanced reconstruction of wet/dry fronts for the shallow water equations. *Journal of Scientific Computing*, 56(2):267–290, 2013.

[11] H. Burchard, E. Deleersnijder, and A. Meister. A high-order conservative Patankar-type discretisation for stiff systems of production–destruction equations. *Applied Numerical Mathematics*, 47(1):1–30, 2003.

[12] J. Casper and H. Atkins. A finite-volume high-order ENO scheme for two-dimensional hyperbolic systems. *Journal of Computational Physics*, 106(1):62–76, 1993.

[13] L. Cea and M. E. Vázquez-Cendón. Unstructured finite volume discretisation of bed friction and convective flux in solute transport models linked to the shallow water equations. *Journal of Computational Physics*, 231(8):3317–3339, 2012.

[14] T. Chen and C.-W. Shu. Review of entropy stable discontinuous Galerkin methods for systems of conservation laws on unstructured simplex meshes. *CSIAM Transactions on Applied Mathematics*, 1:1–52, 2020.

[15] Y. Cheng, A. Chertock, M. Herty, A. Kurganov, and T. Wu. A new approach for designing moving-water equilibria preserving schemes for the shallow water equations. *Journal of Scientific Computing*, 80(1):538–554, 2019.

[16] A. Chertock, S. Cui, A. Kurganov, and T. Wu. Steady state and sign preserving semi-implicit Runge–Kutta methods for ODEs with stiff damping term. *SIAM Journal on Numerical Analysis*, 53(4):2008–2029, 2015.

[17] A. Christlieb, B. Ong, and J.-M. Qiu. Integral deferred correction methods constructed with high order Runge-Kutta integrators. *Mathematics of Computation*, 79(270):761–783, 2010.

[18] M. Ciallella, L. Micalizzi, P. Öffner, and D. Torlo. Modified Patankar Deferred Correction WENO Code for Shallow Water Equations. https://github.com/accdavlo/sw-mpdec.git, October 2021.

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

[20] M. Dumbser. Arbitrary high order pnpm schemes on unstructured meshes for the compressible Navier–Stokes equations. *Computers & Fluids*, 39(1):60–76, 2010.

[21] A. Dutt, L. Greengard, and V. Rokhlin. Spectral Deferred Correction Methods for Ordinary Differential Equations. *BIT Numerical Mathematics*, 40(2):241–266, 2000.

[22] T. C. Fisher, M. H. Carpenter, J. Nordström, N. K. Yamaleev, and C. Swanson. Discretely conservative finite-difference formulations for nonlinear conservation laws in split form: Theory and boundary conditions. *Journal of Computational Physics*, 234:353–375, 2013.

[23] U. S. Fjordholm, S. Mishra, and E. Tadmor. Arbitrarily high-order accurate entropy stable essentially nonoscillatory schemes for systems of conservation laws. *SIAM Journal on Numerical Analysis*, 50(2):544–573, 2012.

[24] E. Gaburro. A unified framework for the solution of hyperbolic PDE systems using high order direct Arbitrary-Lagrangian–Eulerian schemes on moving unstructured meshes with topology change. *Archives of Computational Methods in Engineering*, 28(3):1249–1321, 2021.

[25] E. Gaburro, W. Boscheri, S. Chiocchetti, C. Klingenberg, V. Springel, and M. Dumbser. High order direct Arbitrary-Lagrangian-Eulerian schemes on moving Voronoi meshes with topology changes. *Journal of Computational Physics*, 407:109167, 2020.

[26] S. Godunov. A finite difference method for the computation of discontinuous solutions of the equations of fluid dynamics. *Sbornik: Mathematics*, 47(8-9):357–393, 1959.

[27] S. Gottlieb, D. I. Ketcheson, and C.-W. Shu. *Strong stability preserving Runge-Kutta and multistep time discretizations*. World Scientific, 2011.

[28] H. Hajduk. Monolithic convex limiting in discontinuous Galerkin discretizations of hyperbolic conservation laws. *Computers & Mathematics with Applications*, 87:120–138, 2021.

[29] M. Han Veiga, P. Öffner, and D. Torlo. DeC and ADER: Similarities, Differences and a Unified Framework. *Journal of Scientific Computing*, 87(1):1–35, 2021.

[30] J. Huang and C.-W. Shu. Positivity-preserving time discretizations for production–destruction equations with applications to non-equilibrium flows. *Journal of Scientific Computing*, 78(3):1811–1839, 2019.

[31] J. Huang, W. Zhao, and C.-W. Shu. A third-order unconditionally positivity-preserving scheme for production–destruction equations with applications to non-equilibrium flows. *Journal of Scientific Computing*, pages 1–42, 2018.

[32] T. Izgin, S. Kopecz, and A. Meister. On Lyapunov stability of positive and conservative time integrators and application to second order modified Patankar–Runge–Kutta schemes. *arXiv preprint arXiv:2202.01099*, 2022.

[33] S. Kopecz and A. Meister. Unconditionally positive and conservative third order modified Patankar–Runge–Kutta discretizations of production–destruction systems. *BIT Numerical Mathematics*, pages 1–38, 2018.

[34] S. Kopecz and A. Meister. On the existence of three-stage third-order modified Patankar–Runge–Kutta schemes. *Numerical Algorithms*, pages 1–12, 2019.

[35] D. Kuzmin and M. Q. de Luna. Entropy conservation property and entropy stabilization of high-order continuous Galerkin approximations to scalar conservation laws. *Computers & Fluids*, 213:104742, 2020.

[36] Y. Liu, C.-W. Shu, and M. Zhang. Strong stability preserving property of the deferred correction time discretization. *Journal of Computational Mathematics*, pages 633–656, 2008.

[37] M. Lukáčová-Medvidová, S. Noelle, and M. Kraft. Well-balanced finite volume evolution Galerkin methods for the shallow water equations. *Journal of Computational Physics*, 221(1):122–147, 2007.

[38] Y. Mantri and S. Noelle. Well-balanced discontinuous Galerkin scheme for $2\times$ 2 hyperbolic balance law. *Journal of Computational Physics*, 429:110011, 2021.

[39] A. Meister and S. Ortleb. On unconditionally positive implicit time integration for the DG scheme applied to shallow water flows. *International Journal for Numerical Methods in Fluids*, 76(2):69–94, 2014.

[40] A. Meister and S. Ortleb. A positivity preserving and well-balanced DG scheme using finite volume subcells in almost dry regions. *Applied Mathematics and Computation*, 272:259–273, 2016.

[41] M. L. Minion. Semi-implicit spectral deferred correction methods for ordinary differential equations. *Commun. Math. Sci.*, 1(3):471–500, 09 2003.

[42] S. Noelle, Y. Xing, and C.-W. Shu. High-order well-balanced finite volume WENO schemes for shallow water equation with moving water. *Journal of Computational Physics*, 226(1):29–58, 2007.

[43] J. Núñez-de la Rosa. High-order finite volume solver for the shallow water equations. https://github.com/jbnunezd/fv-solver-sw, November 2020.

[44] J. Núnez-De La Rosa and C.-D. Munz. Hybrid DG/FV schemes for magnetohydrodynamics and relativistic hydrodynamics. *Computer Physics Communications*, 222:113–135, 2018.

[45] P. Öffner. *Approximation and Stability Properties of Numerical Methods for Hyperbolic Conservation Laws*. Habilitation, University Zurich, 2020.

[46] P. Öffner and D. Torlo. Arbitrary high-order, conservative and positivity preserving Patankar-type deferred correction schemes. *Applied Numerical Mathematics*, 153:15–34, 2020.

[47] S. Patankar. *Numerical heat transfer and fluid flow*. CRC press, 1980.

[48] B. Perthame and C.-W. Shu. On positivity preserving finite volume schemes for Euler equations. *Numerische Mathematik*, 73(1):119–130, 1996.

[49] H. Ranocha. Shallow water equations: Split-form, entropy stable, well-balanced, and positivity preserving numerical methods. *GEM – International Journal on Geomathematics*, 8(1):85–133, 04 2017.

[50] M. Ricchiuto. On the C-property and generalized C-property of residual distribution for the shallow water equations. *Journal of Scientific Computing*, 48(1):304–318, 2011.

[51] M. Ricchiuto. An explicit residual based approach for shallow water flows. *Journal of Computational Physics*, 280:306–344, 2015.

[52] M. Ricchiuto and A. Bollermann. Stabilized residual distribution for shallow water simulations. *Journal of Computational Physics*, 228(4):1071–1115, 2009.

[53] M. Ricchiuto and D. Torlo. Analytical travelling vortex solutions of hyperbolic equations for validating very high order schemes. *arXiv preprint arXiv:2109.10183*, 2021.

[54] Y. Saad. *Iterative methods for sparse linear systems*. SIAM, 2003.

[55] J. Shi, C. Hu, and C.-W. Shu. A technique of treating negative weights in WENO schemes. *Journal of Computational Physics*, 175(1):108–127, 2002.

[56] C.-W. Shu. Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws. In *Advanced numerical approximation of nonlinear hyperbolic equations*, pages 325–432. Springer, 1998.

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

[58] A. Thomann, G. Puppo, and C. Klingenberg. An all speed second order well-balanced IMEX relaxation scheme for the Euler equations with gravity. *Journal of Computational Physics*, 420:109723, 2020.

[59] V. A. Titarev and E. F. Toro. Finite-volume WENO schemes for three-dimensional conservation laws. *Journal of Computational Physics*, 201(1):238–260, 2004.

[60] D. Torlo. *Hyperbolic problems: high order methods and model order reduction*. PhD thesis, PhD thesis, University Zurich, 2020.

[61] D. Torlo, P. Öffner, and H. Ranocha. Issues with positivity-preserving Patankar-type schemes. *arXiv:2108.07347*, 2021.

[62] B. Van Leer. Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method. *Journal of computational Physics*, 32(1):101–136, 1979.

[63] M. H. Veiga, D. A. Velasco-Romero, Q. Wenger, and R. Teyssier. An arbitrary high-order spectral difference method for the induction equation. *Journal of Computational Physics*, 438:110327, 2021.

[64] C. Wang, X. Zhang, C.-W. Shu, and J. Ning. Robust high order discontinuous Galerkin schemes for two-dimensional gaseous detonations. *Journal of Computational Physics*, 231(2):653–665, 2012.

[65] Y. Xing and C.-W. Shu. A survey of high order schemes for the shallow water equations. *J. Math. Study*, 47(3):221–249, 2014.

[66] X. Zhang and C.-W. Shu. On positivity-preserving high order discontinuous Galerkin schemes for compressible Euler equations on rectangular meshes. *Journal of Computational Physics*, 229(23):8918–8934, 2010.

[67] X. Zhang and C.-W. Shu. Positivity-preserving high order finite difference WENO schemes for compressible euler equations. *Journal of Computational Physics*, 231(5):2245–2258, 2012.