INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 20, Number 2, Pages 229–266

doi: 10.4208/ijnam2023-1010

ADAPTIVE CENTRAL-UPWIND SCHEME ON TRIANGULAR GRIDS FOR THE SHALLOW WATER MODEL WITH VARIABLE DENSITY

THUONG NGUYEN

Abstract. In this paper, we construct a robust adaptive central-upwind scheme on unstructured triangular grids for two-dimensional shallow water equations with variable density. The method is well-balanced, positivity-preserving, and oscillation free at the curve where two types of fluid merge. The proposed approach is an extension of the adaptive well-balanced, positivity-preserving scheme developed in Epshteyn and Nguyen (arXiv preprint arXiv:2011.06143, 2020). In particular, to preserve "lake-at-rest" steady states, we utilize the Riemann Solver with appropriately rotated coordinates to obtain the point values in neighborhood of the fluid interface. In addition, to improve the efficiency of an adaptive method in the multi-fluid flow, the curve of density discontinuity is reconstructed by using the level set method and volume fraction method. To demonstrate the accuracy, high-resolution, and efficiency of the new adaptive central-upwind scheme, several challenging tests for Shallow water models with variable density are performed.

Key words. Shallow water equations with variable density, central-upwind scheme, well-balanced and positivity-preserving scheme, adaptive algorithm, interface tracking, Riemann solver, weak local residual error estimator, unstructured triangular grid.

1. Introduction

The main goal of this paper is to develop an adaptive well-balanced positivity-preserving central-upwind scheme on triangular grids for shallow water equations with variable density (SWEDs). The two-dimensional (2-D) system of SWEDs can be written as,

$$w_t + (hu)_x + (hv)_y = 0, \tag{1a}$$

$$(hu)_t + \left(hu^2 + \frac{g}{2\rho_0}h^2\rho\right)_x + (huv)_y = -\frac{g}{\rho_0}h\rho B_x, \tag{1b}$$

$$(hv)_t + (huv)_x + \left(hv^2 + \frac{g}{2\rho_0}h^2\rho\right)_y = -\frac{g}{\rho_0}h\rho B_y, \tag{1c}$$

$$(h\rho)_t + (hu\rho)_x + (hv\rho)_y = 0, \tag{1d}$$

where t is the time, x and y are spatial coordinates ($(x,y) \in \Omega$), $h(x,y,t)$ is the water height, $B(x,y)$ is the bottom topography, $w(x,y,t) := h + B$ is the water level, $\rho(x,y,t)$ is the density, $u(x,y,t)$ and $v(x,y,t)$ are the x- and y-components of the flow velocity, g is the constant gravitational acceleration, and ρ_0 is the reference density. The system (1a)–(1d) was proposed in [10, 36, 37, 15] as a variation of the Saint-Venant equations to model multi-phase flows in estuaries or deep ocean currents. The derivation of the system is based on hydrostatic approximation which eliminates the variability in the z-direction. The design of robust and accurate

Received by the editors on March 17, 2022; accepted on September 8, 2022.
2000 *Mathematics Subject Classification.* 76M12, 65M08, 35L65, 86-08, 86A05.

numerical algorithms for computing the solutions of SWEDs system is an important and challenging problem that has been extensively studied in the recent years.

A number of numerical schemes for balance laws have been introduced in recent years, [28, 25, 21, 6, 27, 8, 7, 9, 24, 23, 3, 26, 4, 5, 40, 29]. Most of them utilize a Riemann problem solver for the upwind evolution of the calculated solution. However, as discussed in [9], the eigensystem of the system (1a)–(1d) may be incomplete due to the resonance phenomenon. Hence, it may be very difficult to design a reliable upwind scheme for the SWEDs. In our paper, we therefore use central-upwind schemes which are Riemann-problem-solver free methods, [22, 25, 29]. Central-upwind schemes have been referred to "black-box" solvers for general multidimensional systems of hyperbolic systems of conservation laws. In our prior work [13], we have derived a successful adaptive central-upwind method for Saint-Venant system on triangular grids. We then adapt the developed adaptive scheme in [13] to the new system (1a)–(1d).

Similar to the Saint-Venant system, a good method for SWEDs system should preserve the non-negativity of h and ρ, which is called the positivity-preserving property. In addition, the scheme must ensure a well-balanced property obtained when the numerical method preserve "lake-at-rest" steady-state solutions. Otherwise, the numerical method may lead to significant oscillations. Note that, the system (1a)–(1d) admits the following two "lake-at-rest" steady-state solutions, [9]:

$$w := h + B = \max\big\{C, B(x,y)\big\}, \quad C = \text{Const}, \quad \rho = \mathrm{P} \equiv \text{Const}, \quad \mathrm{u} \equiv \mathrm{v} \equiv 0, \tag{2}$$

and

$$B \equiv \text{Const}, \quad \mathrm{h}^2\rho \equiv \text{Const}, \quad \mathrm{u} \equiv \mathrm{v} \equiv 0. \tag{3}$$

Preserving the solution (3) is a big challenge for numerically solving the system (1a)–(1d) since using the conventional central-upwind methods may not ensure the variable $h^2\rho$, so-called variable pressure, to be constant at the contact waves. It is more difficult to prevent the density oscillation when working on the unstructured triangular grids. Several numerical methods have been proposed for compressible flows, see [46, 31, 9], but only a few efforts, see [9], can simultaneously ensure two types of lake-at-rest states. Therefore, we consider the approach in [9] which is derived to solve the shallow water model with horizontal temperature gradients on rectangular meshes (the system in [9] has similar properties with the SWEDs). In [9], the proposed second-order semi-discrete central-upwind scheme is capable of preserving the ąřlake at restąś steady state (2) and (3) as well as the positivity of the water depth and the temperature (the variable temperature is equivalent to the variable density in our work). In particular, to preserve the second type of ąřlake at restąś steady state (3) and suppress the pressure oscillations across the interface, an efficient interface tracking method is performed. The main idea of the interface approach in [9] is to completely avoid to use the information from the cells where two types of fluids are numerically mixed, so-called "mixed" cells when evolving the solution in the neighborhood of the interface. The data in the ąřmixedąś cells is replaced by the interpolated values that are calculated using the reliable information from the nearby ąřsingle fluidąś cells. Namely, the point values in "mixed" cells are obtained by using the approximated solution of the 1-D Riemann problems between the reliable ąřsingle fluidąś cell averages. However, the central-upwind method and the interface tracking in [9] are designed for structured rectangular grids. In practice, one needs to deal with complicated geometries, where the use of triangular grids could be advantageous or even unavoidable. Hence, in this study, we extend the interface tracking method [9] from the rectangular grids

to unstructured triangular grids by utilizing the idea of the rotated coordinates proposed in [2]. The developed central-upwind scheme in our work is then well-balanced, positivity-preserving, and applicable on triangular meshes.

In addition to achieving the accuracy of the solution, minimizing computational cost is also one of the major challenge in the modeling and numerical analysis of hydrodynamics. The traditional numerical schemes for the system (1a)–(1d) are based on the use of very fine fixed meshes to reconstruct delicate features of the solution. However, this can lead to high computational cost, as well as poor resolution of all small scale features of the problem. In many engineering and scientific applications, it is beneficial to use adaptive meshes for improving the accuracy of the approximation at a much lower cost. There is some very recent effort on the design of adaptive well-balanced and positivity-preserving central-upwind schemes on quad-tree grids for shallow water models [38, 15, 14], but no research has been done for the development of such adaptive schemes for the shallow water with variable density on unstructured triangular grids. Therefore, another goal of this work is to design an adaptive numerical algorithm for the shallow water equations with variable density.

As a part of the mesh reconstruction, we will need to project the data from the old mesh onto a new adaptive mesh. Since we avoid using the cell averages of mixed cells, the data of a new cell which is reconstructed from an old mixed cell is calculated based on the location of the density jumps and the data from the nearby "reliable" cells. Therefore, the interface separating different fluid phases must be accurately tracked or captured. In each "mixed" cell, we will reconstruct an approximate interface, which is called "interface reconstruction". Many methods have been developed for this purpose in the last two decades, such as the level set methods [32, 39], the volume of fluid methods [17, 35], and the front tracking methods [45, 44]. The key idea is to firstly approximate the normal vector of the interface segment and then find its endpoints in each "mixed" cell. In particular, in the level set methods, the level set function is defined at each point as the signed distance from that point to the interface. Since the level set function is continuous, it is then possible to accurately compute its gradients which is essential for finding the normal vector of the interface. However, the level set method may not conserve the mass or the volume of the fluid, see [32, 39, 49]. Hence, in order to obtain more conservative interface reconstruction, some approaches [17, 35] consider the volume of fraction function which is the ratio of the volume of one type of fluid to the total volume of the cell. Though the volume of fraction approaches are able to efficiently conserve the mass, it is hard to capture the interface normal vectors with a high order of accuracy due to the discontinuities at the fluid jumps. In summary, each method has its own strengths and drawbacks. Therefore, the coupled level set/volume of fraction methods have been developed in [42, 49, 18, 17, 41]. Different from previous works, the coupled methods incorporate both the level set and the volume fraction methods for reconstructing the interface and is therefore superior to either method alone. Namely, in [42, 49, 18, 17, 41], this approach has a second-order accuracy by using level set method to obtain the interface normal vectors and is also conservative thanks to utilizing the volume of fraction method for tracking the endpoints of the interface. In addition, this interface tracking method is applicable on different types of grid. Notable previous works on this method have been performed on 1D grids, see [1], on 2D grids such as structured rectangular grids, see [18, 17, 41], and unstructured triangular grids, see [49], and also on 3D grids, see [42]. Here, in our work on the unstructured triangular meshes,

we will use the interface reconstruction proposed in [49] as a part of the adaptive mesh reconstruction.

In summary, we first design the well-balanced positivity-preserving central-upwind scheme to accurately simulate the multi-phase flows on unstructured triangular grids. The well-balanced property is achieved by considering the point value reconstructions proposed in [9] at the fluid interfaces. Next, we adapt the adaptive algorithm in space and time in [13] from the Saint-Venant system to the SWEDs system to obtain high accuracy with low cost. The interface reconstruction developed in [49] and and some special techniques are performed at the interface of fluids to prevent pressure oscillation.

The developed scheme in this research clearly verifies the high versatility and efficiency of the adaptive scheme which was proposed in our prior work [13]. Moreover, the proposed adaptive central-upwind method can be extended to apply for different models of multi-phase flows as well as for various types of geometrical discretization. This research also opens many posibilities of developing high-resolution, accurate, and efficient schemes for the system (1a)-(1d). For example, most of current numerical simulations of multi-phase flows fail to preserve the lake-at-rest states and the positivity of water height h and density ρ in the problems containing both wet and dry states. In [29], some special techniques were derived to handle such wet/dry states, but they are used for the Saint-Venant system and are applicable only on triangular grids. Hence, it is urging to obtain efficient numerical methods on triangular meshes for the compressible shallow water models; such task is fulfilled in this research.

This paper is organized as follows. In Section 2, we present a well-balanced positivity-preserving central-upwind scheme on unstructured triangular grids for SWEDs which serves as the underlying discretization for the developed adaptive algorithm. Next, in Section 3.1, we present the procedure to detect mixed cells. The interface approximation will be briefly reviewed in Section 3.2. In Section 3.3, we discuss the cell averages correction, which is used to prevent the density diffusion around the density jumps. We summarize the adaptive central-upwind method in Section 4.1. We discuss a strategy of adaptive mesh refinement in Section 4.2. In Section 4.3, we present an adaptive second-order strong stability preserving Runge-Kutta method, employed as a part of time evolution for the adaptive central-upwind scheme. We develop a local posteriori error estimator in Section 4.4 which is used as a robust indicator for the adaptive mesh refinement in our work. Finally, in Section 5, we illustrate the high accuracy and efficiency of the developed adaptive central-upwind scheme on a number of challenging tests for multi-fluid shallow water models.

2. The Central-Upwind Scheme for SWEDs in Triangular Mesh

In this section, we focus on developing the central-upwind method for the SWEDs system (1a)-(1d), [15]. In particular, we will extend the adaptive scheme from the Saint-Venant system in [13] to the SWEDs system. The developed scheme will eliminate the oscillation appearing at the interface and ensure the well-balanced and positivity-preserving properties.

We first rewrite the SWEDs system (1a)-(1d) into the vector form as,

$$\mathbf{U}_t + \mathbf{F}(\mathbf{U}, B)_x + \mathbf{G}(\mathbf{U}, B)_y = \mathbf{S}(\mathbf{q}, B), \tag{4}$$

where

$$U = (w, hu, hv, h\rho),$$

and the fluxes and source term are:

$$
\begin{aligned}
\mathbf{F}(\mathbf{U},B) &= (hu, \frac{(hu)^2}{w-B} + \frac{g}{2\rho_0}\rho(w-B)^2, \frac{(hu)(hv)}{w-B}, hu\rho)^T,\\
\mathbf{G}(\mathbf{q},B) &= (hv, \frac{(hu)(hv)}{w-B}, \frac{(hv)^2}{w-B} + \frac{g}{2\rho_0}\rho(w-B)^2, hv\rho)^T,\\
\mathbf{S}(\mathbf{U},B) &= (0, -g\frac{\rho}{\rho_0}(w-B)B_x, -g\frac{\rho}{\rho_0}(w-B)B_y, 0)^T.
\end{aligned}
\tag{5}
$$

In our research, we consider the triangular mesh illustrated in Fig. 2.1 with the following notations.

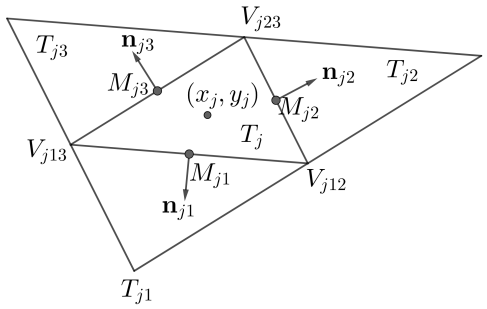


Figure 2.1. A typical triangular cell with three neighbors.

$\mathcal{T} := \{T_j\}_j$ is an unstructured triangulation of the computational domain Ω;
$T_j \in \mathcal{T}$ is a triangular cell of size $|T_j|$ with the barycenter (x_j, y_j);
$V_{j\kappa} = (\widetilde{x}_{j\kappa}, \widetilde{y}_{j\kappa})$, $\kappa = 12, 23, 31$ are the three vertices of T_j;
T_{jk}, $k = 1, 2, 3$ are the neighboring triangles that share a common side with T_j;
ℓ_{jk} is the length of the common side of T_j and T_{jk}, and M_{jk} is its midpoint;
$\boldsymbol{n}_{jk} := (\cos(\theta_{jk}), \sin(\theta_{jk}))^\top$ is the outer unit normal to the kth side of T_j.

Next, the bottom topography B is replaced with its continuous piecewise linear approximation $\widetilde{B}$ given by

$$
\begin{vmatrix}
x - \widetilde{x}_{j12} & y - \widetilde{y}_{j12} & \widetilde{B}(x,y) - \widehat{B}_{j12}\\
\widetilde{x}_{j23} - \widetilde{x}_{j12} & \widetilde{y}_{j23} - \widetilde{y}_{j12} & \widehat{B}_{j23} - \widehat{B}_{j12}\\
\widetilde{x}_{j13} - \widetilde{x}_{j12} & \widetilde{y}_{j13} - \widetilde{y}_{j12} & \widehat{B}_{j13} - \widehat{B}_{j12}
\end{vmatrix} = 0, \quad (x,y) \in T_j,
$$

where, in the case of continuous bottom topography, $\widehat{B}_{j\kappa} := B(V_{j\kappa})$ $\kappa = 12, 23, 31$. Then, denote:

$$
B_{jk} := \widetilde{B}(M_{jk}), \quad B_j := \widetilde{B}(x_j, y_j) = \frac{1}{3}(\widehat{B}_{j12} + \widehat{B}_{j23} + \widehat{B}_{j13}).
$$

At time t, define by $\overline{\boldsymbol{U}}_j(t)$ the approximation of the cell averages of the solution,

$$
\overline{\boldsymbol{U}}_j(t) \approx \frac{1}{|T_j|} \iint\limits_{T_j} \boldsymbol{U}(x,y,t)\, dxdy.
$$

From now on, in order to shorten the formula, we suppress the time-dependence in the algorithm. All indexed quantities used in the following formula will be computed at time t. Then, as shown in [29, 5], the semi-discrete second-order

central-upwind scheme for the Saint-Venant system (1a)-(1d) on triangular grids is written as the following system of ODEs,

$$\frac{d\overline{\boldsymbol{U}}_j}{dt} = -\frac{1}{|T_j|}\big[\boldsymbol{H}_{j1}+\boldsymbol{H}_{j2}+\boldsymbol{H}_{j3}\big]+\overline{\boldsymbol{S}}_j, \tag{6}$$

where the numerical fluxes through the edges of the triangular cell T_j are

$$\boldsymbol{H}_{jk} = \frac{\ell_{jk}\cos(\theta_{jk})}{a^{\text{in}}_{jk}+a^{\text{out}}_{jk}}\Big[a^{\text{in}}_{jk}\boldsymbol{F}(\boldsymbol{U}_{jk}(M_{jk}),B_{jk})+a^{\text{out}}_{jk}\boldsymbol{F}(\boldsymbol{U}_j(M_{jk}),B_{jk})\Big] \tag{7}$$

$$+\frac{\ell_{jk}\sin(\theta_{jk})}{a^{\text{in}}_{jk}+a^{\text{out}}_{jk}}\Big[a^{\text{in}}_{jk}\boldsymbol{G}(\boldsymbol{U}_{jk}(M_{jk}),B_{jk})+a^{\text{out}}_{jk}\boldsymbol{G}(\boldsymbol{U}_j(M_{jk}),B_{jk})\Big] \tag{8}$$

$$-\ell_{jk}\frac{a^{\text{in}}_{jk}a^{\text{out}}_{jk}}{a^{\text{in}}_{jk}+a^{\text{out}}_{jk}}\big[\boldsymbol{U}_{jk}(M_{jk})-\boldsymbol{U}_j(M_{jk})\big],\quad k=1,2,3. \tag{9}$$

Here, $\boldsymbol{U}_j(M_{jk})$ and $\boldsymbol{U}_{jk}(M_{jk})$ are the reconstructed point values of $\boldsymbol{U}$ at the middle points of the edges M_{jk}. To obtain these values, first, a piecewise linear reconstruction of the variables $\boldsymbol{\Upsilon}:=(w,u,v,\rho)^\top$ is computed as,

$$\widetilde{\boldsymbol{\Upsilon}}(x,y)=\sum_j \boldsymbol{\Upsilon}_j(x,y)\chi_{T_j},\quad \boldsymbol{\Upsilon}_j(x,y):=\boldsymbol{\Upsilon}_j+(\widehat{\boldsymbol{\Upsilon}}_x)_j(x-x_j)+(\widehat{\boldsymbol{\Upsilon}}_y)_j(y-y_j), \tag{10}$$

where χ_{T_j} is the characteristic function of the cell T_j, $\boldsymbol{\Upsilon}_j$ are the point values of $\boldsymbol{\Upsilon}$ at the cell centers, and $(\widehat{\boldsymbol{\Upsilon}}_x)_j$ and $(\widehat{\boldsymbol{\Upsilon}}_y)_j$ are the limited partial derivatives (see Section 3 in [29] for more details of the computation). In order to compute the cell center point values of the density, $\rho_j\approx\rho(x_j,y_j,t)$ in cell T_j and velocities $u_j\approx u(x_j,y_j,t)$ and $v_j\approx v(x_j,y_j,t)$, we use the desingularization procedure presented in [29]. After that, the second and third components of the point values $\boldsymbol{U}_j(M_{jk})$ and $\boldsymbol{U}_{jk}(M_{jk})$ are obtained from, $\boldsymbol{\Upsilon}_j(M_{jk})$ and $\boldsymbol{\Upsilon}_{jk}(M_{jk})$,

$$(hu)_j(M_{jk})=(w_j(M_{jk})-B_{jk})\,u_j(M_{jk}),(hu)_{jk}(M_{jk})=(w_{jk}(M_{jk})-B_{jk})\,u_{jk}(M_{jk}),$$
$$(hv)_j(M_{jk})=(w_j(M_{jk})-B_{jk})\,v_j(M_{jk}),(hv)_{jk}(M_{jk})=(w_{jk}(M_{jk})-B_{jk})\,v_{jk}(M_{jk}).$$
$$(h\rho)_j(M_{jk})=(w_j(M_{jk})-B_{jk})\,\rho_j(M_{jk}),(h\rho)_{jk}(M_{jk})=(w_{jk}(M_{jk})-B_{jk})\,\rho_{jk}(M_{jk}).$$

However, in some numerical examples, the oscillation may appear when we use the piecewise linear approximation (10) to obtain the point values in some cells. It may occur due to the appearance of local extrema values at middle points or as a consequence of the density discontinuity. To prevent this oscillation, we will use some techniques that will be discussed later in Sections 2.1 and 2.2.

In (9), a^{in}_{jk} and a^{out}_{jk} are the one-sided local speeds of propagation in the directions $\pm\boldsymbol{n}_{jk}$. These speeds are related to the largest and smallest eigenvalues of the Jacobian matrix $J_{jk}=\cos(\theta_{jk})\,\frac{\partial\boldsymbol{F}}{\partial\boldsymbol{U}}+\sin(\theta_{jk})\,\frac{\partial\boldsymbol{G}}{\partial\boldsymbol{U}}$, denoted by $\lambda_+[J_{jk}]$ and $\lambda_-[J_{jk}]$, respectively, and are defined by

$$\begin{aligned} a^{\text{in}}_{jk} &= -\min\{\lambda_-[J_{jk}(\boldsymbol{U}_j(M_{jk}))],\,\lambda_-[J_{jk}(\boldsymbol{U}_{jk}(M_{jk})],\,0\},\\ a^{\text{out}}_{jk} &= \max\{\lambda_+[J_{jk}(\boldsymbol{U}_j(M_{jk}))],\,\lambda_+[J_{jk}(\boldsymbol{U}_{jk}(M_{jk})],\,0\}, \end{aligned} \tag{11}$$

where

$$\lambda_\pm[J_{jk}(\boldsymbol{U}_j(M_{jk}))]=\cos(\theta_{jk})u_j(M_{jk})+\sin(\theta_{jk})v_j(M_{jk})\pm\sqrt{\frac{g}{\rho_0}h_j(M_{jk})\rho_j(M_{jk})},$$
$$\lambda_\pm[J_{jk}(\boldsymbol{U}_{jk}(M_{jk}))]=\cos(\theta_{jk})u_{jk}(M_{jk})+\sin(\theta_{jk})v_{jk}(M_{jk})$$
$$\pm\sqrt{\frac{g}{\rho_0}h_{jk}(M_{jk})\rho_{jk}(M_{jk})}.$$

Remark: In order to avoid division by 0 (or by a very small positive number), the numerical flux (9) is replaced with

$$\begin{aligned}\boldsymbol{H}_{jk} =&\frac{\ell_{jk}\cos(\theta_{jk})}{2}\left[\boldsymbol{F}(\boldsymbol{U}_{jk}(M_{jk}),B_{jk})+\boldsymbol{F}(\boldsymbol{U}_{j}(M_{jk}),B_{jk})\right]\\ &+\frac{\ell_{jk}\sin(\theta_{jk})}{2}\left[\boldsymbol{G}(\boldsymbol{U}_{jk}(M_{jk}),B_{jk})+\boldsymbol{G}(\boldsymbol{U}_{j}(M_{jk}),B_{jk})\right]\end{aligned}$$

wherever $a^{\text{in}}_{jk}+a^{\text{out}}_{jk}<\sigma$. In all of the reported numerical examples in Section 5, we have taken $\sigma=10^{-6}$.

Finally, the cell average of the source term $\boldsymbol{S}_j$ in (6),

$$\overline{\boldsymbol{S}}_j(t)\approx\frac{1}{|T_j|}\iint\limits_{T_j}\boldsymbol{S}\big(\boldsymbol{U}(x,y,t),B(x,y)\big)\,dxdy,$$

has to be discretized in a well-balanced manner presented later in Section 2.4

2.1. The Limiter for Piecewise Linear Approximation. As discussed above, we may observe the oscillation when using piecewise linear approximation (10) due to the appearance of local extrema at midpoints M_{jk}, see [29]. Hence, in order to maintain the numerical stability of the scheme, the following monotonicity condition must be satisfied.

$$\min\left(\boldsymbol{\Upsilon}^{(i)}_j,\boldsymbol{\Upsilon}^{(i)}_{jk}\right)\le\boldsymbol{\Upsilon}^{(i)}_j(M_{jk})\le\max\left(\boldsymbol{\Upsilon}^{(i)}_j,\boldsymbol{\Upsilon}^{(i)}_{jk}\right),\quad k=1,2,3, \tag{12}$$

where $\boldsymbol{\Upsilon}^{(i)}_j(M_{jk})$ is the i-th component of the point values at midpoint M_{jk} which is obtained by the linear reconstruction (10). Here, $\boldsymbol{\Upsilon}^{(i)}_j$ and $\boldsymbol{\Upsilon}^{(i)}_{jk}$ are the center point value in cell T_j and the neighboring cell T_{jk}, see [20]. To ensure that the reconstructed point value $\boldsymbol{\Upsilon}^{(i)}_j(M_{jk})$ is between two cell averages $\boldsymbol{\Upsilon}^{(i)}_j$ and $\boldsymbol{\Upsilon}^{(i)}_{jk}$, in [29, 4], the gradients $\widehat{\boldsymbol{\Upsilon}}_x$ and $\widehat{\boldsymbol{\Upsilon}}_y$ are set to zero for cells violating at least one of the inequalities (12). However, using zero gradients may reduce the convergence rate in numerical simulation. Hence, in our research, instead of zero gradients, we consider the idea of the positivity-preserving limiter developed in [33, 47] to correct the piecewise linear reconstruction. The details for this correction are presented as follows.

Consider an arbitrary cell T_j with the polynomial $\boldsymbol{\Upsilon}^{(i)}_j(x,y)$ that does not satisfy at least one of local extrema conditions (12). In that case, we will replace $\boldsymbol{\Upsilon}^{(i)}_j(x,y)$ by

$$(\boldsymbol{\Upsilon}^{(i)}_j)^*(x,y)=\theta(\boldsymbol{\Upsilon}^{(i)}_j(x,y)-\boldsymbol{\Upsilon}^{(i)}_j)+\boldsymbol{\Upsilon}^{(i)}_j, \tag{13}$$

where $\theta\in[0,1]$ such that $(\boldsymbol{\Upsilon}^{(i)}_j)^*(x,y)$ satisfies the inequalities (12) for all $k=1,2,3$. We then need to solve for the constant θ. Plugging (13) into the inequalities (12), we have

$$\theta\in\left[\min(\alpha_{jk},\beta_{jk}),\max(\alpha_{jk},\beta_{jk})\right],\quad k=1,2,3 \tag{14}$$

or

$$\theta\in[\max_k(\min(\alpha_{jk},\beta_{jk})),\min_k(\max(\alpha_{jk},\beta_{jk}))] \tag{15}$$

where $\alpha_{jk} = \dfrac{\min\left(\boldsymbol{\Upsilon}_j^{(i)}, \boldsymbol{\Upsilon}_{jk}^{(i)}\right) - \boldsymbol{\Upsilon}_j^{(i)}}{\boldsymbol{\Upsilon}_j^{(i)}(M_{jk}) - \boldsymbol{\Upsilon}_j^{(i)}}, \quad \beta_{jk} = \dfrac{\max\left(\boldsymbol{\Upsilon}_j^{(i)}, \boldsymbol{\Upsilon}_{jk}^{(i)}\right) - \boldsymbol{\Upsilon}_j^{(i)}}{\boldsymbol{\Upsilon}_j^{(i)}(M_{jk}) - \boldsymbol{\Upsilon}_j^{(i)}}.$

Hence we can take

$$\theta = \min(\min_k(\max(\alpha_{jk}, \beta_{jk})), 1), \tag{16}$$

Note that $\theta \geq 0$ since $\theta = 0$, which corresponds to the constant approximation for $(\boldsymbol{\Upsilon}_j^{(i)})^*(x, y)$, is one solution of the inequalities (12). Hence, with this optimized limiter, the point values stay within the given range. In addition, the approximated values of water depth and density at middle points are also positive as $\boldsymbol{\Upsilon}_j^{(i)}(M_{jk}) \geq \min\left(\boldsymbol{\Upsilon}_j^{(i)}, \boldsymbol{\Upsilon}_{jk}^{(i)}\right) > 0$. Therefore, the stability is ensured without using zero-gradients.

2.2. Riemann Solver for Mixed Cells. As mentioned in Section 1 and [9], a good scheme should not develop spurious pressure oscillations in the neighborhood of density jumps. The oscillations appear when we numerically solve the compressible multifluid systems by using conventional Godunov-type finite-volume methods. We define the cells which contains the curves of density discontinuity as "mixed" cells. Note that the quantities in mixed cells are calculated as an artificial numerical mixture of two different fluids. Hence, the cell averages of mixed cells may have no or very little physical sense and become 'unreliable' [9]. Consequently, using the cell averages in mixed cells to obtain the linear approximation (10) in the neighborhood of the interface will lead to unexpected pressure oscillations. In [9], to eliminate the oscillation, instead of the linear approximation (10), the point values in the "mixed" cells are calculated by using the approximated solution of 1-D Riemann problems. Namely, according to this approach, the 1-D Riemann Solver is applied in the x-direction to approximate the point values on the vertical boundaries of each rectangular mixed cell. Similarly, for the point values on horizontal sides of the rectangular, 1-D Riemann problem is considered in y-direction. This leads to the idea of applying Riemann Solver approach in the direction of the normal vector, so called "normal direction", to get the middle point values on each side of the mixed cells in triangular meshes. We can understand this idea as rotating the reference coordinate. Recently, a method using rotated coordinate frame to solve multi-dimensional Riemann problems has been proposed in [2]. In such approach, the 2-D Riemann Problem is converted to 1-D problem in a particular direction in order to easily obtain the corresponding solution in 2-D. Therefore, in our research, we will calculate the point values in triangles based on the intermediate states of 1-D Riemann problem in the normal direction as follows.

Assume that triangle T_j is a mixed cell which has the outward unit normal vector $\mathbf{n}_{jk} = (\cos(\theta_{jk}), \sin(\theta_{jk}))$ on side k. We will consider a new reference frame (x', y') such that the new horizontal axis is in the direction of the outward normal vector $\mathbf{n}_{jk}$ and the origin $(0, 0)$ is at the middle point M_{jk} of side k as shown in Fig. 2.2.

Suppose T_L and T_R to be the two closest single-fluid cells to T_j such that T_L and T_R stay on different sides of the k-th edge of cell T_j. We assume that T_R stays on the side where the outward normal vector $\mathbf{n}_{jk}$ is pointing. Let $\boldsymbol{\Upsilon}_L = (w_L, u_L, v_L, \rho_L)$ and $\boldsymbol{\Upsilon}_R = (w_R, u_R, v_R, \rho_R)$ be the center point values of single-fluid cells T_L and T_R. Note that (u_L, v_L) and (u_R, v_R) are the values of velocities in the Cartesian coordinate system (x, y). The projections of velocities (u_L, v_L) and (u_R, v_R) onto the rotated frame (x', y') are

$$u'_L = u_L \cos(\theta_{jk}) + v_L \sin(\theta_{jk}), \quad v'_L = -u_L \sin(\theta_{jk}) + v_L \cos(\theta_{jk}),$$

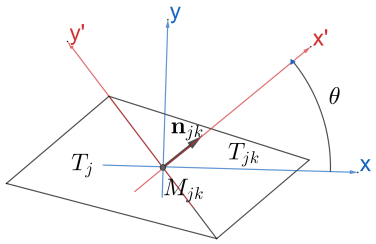


Figure 2.2. An example of the rotated coordinates (x', y') where x'-axis is in the direction of normal vector $\mathbf{n}_{jk}$ and the new origin is at midpoint M_{jk} of side k.

$$u'_R = u_R\cos(\theta_{jk}) + v_R\sin(\theta_{jk}), \quad v'_R = -u_R\sin(\theta_{jk}) + v_R\cos(\theta_{jk}).$$

Similar to [2, 9], to compute the point value at midpoint M_{jk} on side k of triangle T_j, we have to solve the following 1-D Riemann problems between states $\boldsymbol{\Upsilon}_L$ an $\boldsymbol{\Upsilon}_R$ in direction $\mathbf{n}_{jk}$.

$$\begin{cases} w_t + (hu)_{x'} = 0 \\ (hu)_t + \left(\dfrac{(hu)^2}{w-B} + \frac{g}{2}\frac{\rho}{\rho_0}(w-B)\right)_{x'} = -g\dfrac{\rho}{\rho_0}(w-B)B_{x'}, \\ (hv)_t + \left(\dfrac{(hu)(hv)}{w-B}\right)_{x'} = 0, \\ (h\frac{\rho}{\rho_0})_t + \left(uh\dfrac{\rho}{\rho_0}\right)_{x'} = 0, \end{cases} \tag{17}$$

subject to the following initial condition

$$(w,u,v,\rho,B)(x',0) = \begin{cases} (\boldsymbol{\Upsilon}_L)' := (w_L, u'_L, v'_L, \rho_L, B_L) & \text{if} \quad x' < 0, \\ (\boldsymbol{\Upsilon}_R)' := (w_R, u'_R, v'_R, \rho_R, B_R) & \text{if} \quad x' > 0. \end{cases} \tag{18}$$

The details of the approximate Riemann solver for the above Riemann problem can be seen in [9]. Suppose $(\boldsymbol{\Upsilon}^*_L)' := (w^*_L, (u^*_L)', (v^*_L)', \rho^*_L)$ and $(\boldsymbol{\Upsilon}^*_R)' := (w^*_R, (u^*_R)', (v^*_R)', \rho^*_R)$ are the intermediate states calculated by Riemann solver for (17)-(18) in (x', y') coordinates. We recall that in [9], the point values at midpoints on left and right boundaries of a rectangular mixed cell are obtained respectively based on the left and right Riemann intermediate states,$(\boldsymbol{\Upsilon}^*_L)'$ and $(\boldsymbol{\Upsilon}^*_R)'$. Here, in triangular grids, we can choose either $(\boldsymbol{\Upsilon}^*_L)'$ or $(\boldsymbol{\Upsilon}^*_R)'$ to calculate the point values at the middle point M_{jk} of cell T_j. Note that T_L and T_R are both single-fluid cells nearby mixed cell T_j. However, if the neighboring cell T_{jk} shown in Fig. 2.2 is a single-fluid cell, then $T_R \equiv T_{jk}$ and $M_{jk} \in T_R$, which means T_R is closer to M_{jk} compared to T_L. Hence, in our work, we select the right intermediate values $(\boldsymbol{\Upsilon}^*_R)'$ and the information from T_R to compute the point values $\boldsymbol{\Upsilon}_j(M_{jk})$. From [9], we first calculate the sound of speed $c^*_R = h^2\dfrac{g\rho}{2\rho_0}$. If $h^*_R > 0, \rho^*_R > 0$ and $(u^*)'_R - c^*_R < 0$, then we replace the piecwise linear approximation in (10) by $\boldsymbol{\Upsilon}_j(M_{jk}) = (\boldsymbol{\Upsilon}^*_R)'$, otherwise $\boldsymbol{\Upsilon}_j(M_{jk}) = (\boldsymbol{\Upsilon}_R)'$ (more details of the point values correction can be

seen in [9]). Finally, we convert the computed solution back to original Cartesian coordinate (x, y) as

$$\boldsymbol{\Upsilon}_j(M_{jk}) = \begin{cases} \boldsymbol{\Upsilon}_R^*, & \text{if} \quad h_R^* > 0, \rho_R^* > 0, (u^*)_R' - c_R^* < 0, \\ \boldsymbol{\Upsilon}_R, & \text{otherwise}, \end{cases} \tag{19}$$

where $\boldsymbol{\Upsilon}_R^* := (w_R^*, (u_R^*)' \cos(\theta_{jk}) - (v_R^*)' \sin(\theta_{jk}), (u_R^*)' \sin(\theta_{jk}) + (v_R^*)' \cos(\theta_{jk}), \rho_R^*)$.

From [9], in "lake at rest" solutions (2) and (3), we have $\boldsymbol{\Upsilon}_R^* = \boldsymbol{\Upsilon}_R$ and the steady state solutions are then preserved by using the discretization of the source term presented in Section 2.4.

2.3. Positivity-preserving condition. Next, we will discuss the restriction of time step to preserve the positivity of water depth h and the variable $h\rho$. Consider the Forward Euler (FE) equation

$$\overline{U}_j^{n+1} = \overline{U}_j^n - \frac{1}{|T_j|} \sum_{k=1}^{3} \Delta t_{jk} H_{jk} + \Delta t \, \bar{S}_j.$$

From [4], the time step condition to guarantee the positivity of water height h is

$$\Delta t < \frac{1}{6a} \min_{jk}(r_{jk}), \tag{20}$$

where $a = \max_{jk}(a_{jk}^{\text{in}}, a_{jk}^{\text{out}})$. One can clearly see that the time step size (20) will also achieve nonnegative water height h in the density shallow water model (1a)-(1d), see [4, 29]. We now use the idea from [4] to find the CFL-type condition for the positivity-preserving property of the last variable $h\rho$. To this end, we apply the forward Euler discretization to the last component of the scheme.

$$\begin{aligned} (\overline{h\rho})_j^{n+1} = (\overline{h\rho})_j^n &- \frac{\Delta t}{|T_j|} \sum_{k=1}^{3} \frac{\ell_{jk} \cos(\rho_{jk})}{a_{jk}^{\text{in}} + a_{jk}^{\text{out}}} \Big[a_{jk}^{\text{in}} (h\rho u)_{jk}(M_{jk}) + a_{jk}^{\text{out}} (h\rho u)_j(M_{jk}) \Big] \\ &- \frac{\Delta t}{|T_j|} \sum_{k=1}^{3} \frac{\ell_{jk} \sin(\rho_{jk})}{a_{jk}^{\text{in}} + a_{jk}^{\text{out}}} \Big[a_{jk}^{\text{in}} (h\rho v)_{jk}(M_{jk}) + a_{jk}^{\text{out}} (h\rho v)_j(M_{jk}) \Big] \\ &+ \frac{\Delta t}{|T_j|} \sum_{k=1}^{3} \ell_{jk} \frac{a_{jk}^{\text{in}} a_{jk}^{\text{out}}}{a_{jk}^{\text{in}} + a_{jk}^{\text{out}}} \big[(h\rho)_{jk}(M_{jk}) - (h\rho)_j(M_{jk}) \big]. \end{aligned} \tag{21}$$

Note that

$$(\overline{h\rho})_j^n = h_j \rho_j = \left(\frac{1}{3} \sum_{m=1}^{3} h_j(M_{jm}) \right) \left(\frac{1}{3} \sum_{s=1}^{3} \rho_j(M_{js}) \right) = \frac{1}{9} \sum_{m,s} h_j(M_{jm}) \rho_j(M_{js}). \tag{22}$$

Plug (22) to (21), we have

$$\begin{aligned} (\overline{h\rho})_j^{n+1} = &\frac{1}{9} \sum_{m,s} h_j(M_{jm}) \rho_j(M_{js}) \\ &- \frac{\Delta t}{|T_j|} \sum_{k=1}^{3} \frac{\ell_{jk} \cos(\rho_{jk})}{a_{jk}^{\text{in}} + a_{jk}^{\text{out}}} \Big[a_{jk}^{\text{in}} h_{jk}(M_{jk}) \rho_{jk}(M_{jk}) u_{jk}(M_{jk}) \end{aligned}$$

$$
\begin{aligned}
&+a_{jk}^{\text{out}}h_j(M_{jk})\rho_j(M_{jk})u_j(M_{jk})\Big]\\
&-\frac{\Delta t}{|T_j|}\sum_{k=1}^{3}\frac{\ell_{jk}\sin(\rho_{jk})}{a_{jk}^{\text{in}}+a_{jk}^{\text{out}}}\Big[a_{jk}^{\text{in}}h_{jk}(M_{jk})\rho_{jk}(M_{jk})v_{jk}(M_{jk})\\
&+a_{jk}^{\text{out}}h_j(M_{jk})\rho_j(M_{jk})v_j(M_{jk})\Big]\\
&+\frac{\Delta t}{|T_j|}\sum_{k=1}^{3}\ell_{jk}\frac{a_{jk}^{\text{in}}a_{jk}^{\text{out}}}{a_{jk}^{\text{in}}+a_{jk}^{\text{out}}}\left[h_{jk}(M_{jk})\rho_{jk}(M_{jk})-h_j(M_{jk})\rho_j(M_{jk})\right]\\
=&\frac{1}{9}\sum_{m\neq s}h_j(M_{jm})\rho_j(M_{js})\\
&+\sum_{k=1}^{3}h_j(M_{jk})\rho_j(M_{jk})\Big[\frac{1}{9}-\frac{\Delta t}{|T_j|}\frac{\ell_{jk}a_{jk}^{\text{out}}}{a_{jk}^{\text{in}}+a_{jk}^{\text{out}}}\left(a_{jk}^{\text{in}}+u_j^{\perp}(M_{jk})\right)\Big]\\
&+\sum_{k=1}^{3}h_{jk}(M_{jk})\rho_{jk}(M_{jk})\frac{\Delta t}{|T_j|}\frac{\ell_{jk}a_{jk}^{\text{in}}}{a_{jk}^{\text{in}}+a_{jk}^{\text{out}}}\left(a_{jk}^{\text{out}}-u_{jk}^{\perp}(M_{jk})\right).
\end{aligned}
$$

From the definitions of the local speeds (11) we obtain that

$$u_j^{\perp}(M_{jk})=\cos(\theta_{jk})u_j(M_{jk})+\sin(\theta_{jk})v_j(M_{jk})\le a_{jk}^{\text{out}},0\le a_{jk}^{\text{in}},\text{and}0\le a_{jk}^{\text{out}}.$$

Besides, the corrected reconstruction in Section 2.1 guarantees that $h_{jk}(M_{jk})\ge 0$ and $\rho_{jk}(M_{jk})\ge 0$ for all j and $k=1,2,3$. Therefore, all terms in the first and third sum on the RHS of (2.3) are nonnegative. To enforce the second sum on the RHS of (2.3) to be nonegative, we then need for all j and $k=1,2,3$:

$$\frac{\Delta t}{|T_j|}\frac{\ell_{jk}a_{jk}^{\text{out}}}{a_{jk}^{\text{in}}+a_{jk}^{\text{out}}}\left(a_{jk}^{\text{in}}+u_j^{\perp}(M_{jk})\right)\le\frac{\Delta t}{|T_j|}\frac{\ell_{jk}a_{jk}^{\text{out}}}{a_{jk}^{\text{in}}+a_{jk}^{\text{out}}}\left(a_{jk}^{\text{in}}+a_{jk}^{\text{out}}\right)=\frac{\Delta t}{|T_j|}l_{jk}a_{jk}^{\text{out}}<\frac{1}{9}.$$

Hence we conclude that all terms in the second sum on the RHS of (2.3) are also nonnegative under the following CFL-type condition.

$$\Delta t<\frac{1}{18a}\min_{jk}(r_{jk})=\frac{\min_{jk}\left(\dfrac{2|T_j|}{l_{jk}}\right)}{18\max_{jk}(a_{jk}^{\text{in}},a_{jk}^{\text{out}})}<\frac{1}{9}\min_{jk}\left(\frac{|T_j|}{l_{jk}a_{jk}^{\text{out}}}\right),\tag{23}$$

where $a=\max_{jk}(a_{jk}^{\text{in}},a_{jk}^{\text{out}})$ and $r_{jk}=\dfrac{2|T_j|}{l_{jk}}$ is the k-th altitude of triangle T_j. This completes the proof. This proof is still valid if one uses a higher-order SSP ODE solver (either the Runge-Kutta or the multistep one), because such solvers can be written as a convex combination of several forward Euler steps.

2.4. Well-balanced discretization of source term. In this section, we first develop a well-balanced discretization of the source terms, which guarantees the first type of "lake at rest" steady-state solutions,

$$w=\max\big\{C,B(x,y)\big\},\quad C=\text{Const},\quad \rho=\text{P}\equiv\text{Const},\quad \text{u}\equiv\text{v}\equiv 0,\tag{24}$$

are exactly preserved by the resulting central-upwind scheme. This means that the source discretization $\overline{\mathbf{S}}_j$ should exactly balance the numerical fluxes so that the right-hand side (RHS) of (1a)-(1d) vanishes at "lake at rest" steady states.

To this end, we substitute the "lake at rest" state (24) into FE scheme of the system (it is still correct with the adaptive Runge-Kutta scheme) and conclude that a well-balanced quadrature for $\bar{\mathbf{S}}_j$ should satisfy the following two conditions:

$$-\frac{g}{2|T_j|}\sum_{k=1}^{3}\ell_{jk}\cos(\theta_{jk})\cdot\frac{\Delta t_{jk}^{(\rho)}}{\Delta t}\cdot\frac{P}{\rho_0}\big[C-B(M_{jk})\big]^2+\bar{S}_j^{(2)}=0, \tag{25}$$

$$-\frac{g}{2|T_j|}\sum_{k=1}^{3}\ell_{jk}\sin(\theta_{jk})\cdot\frac{\Delta t_{jk}^{(\rho)}}{\Delta t}\cdot\frac{P}{\rho_0}\big[C-B(M_{jk})\big]^2+\bar{S}_j^{(3)}=0, \tag{26}$$

where

$$\bar{S}_j^{(2)}\approx-\frac{g}{|T_j|\rho_0}\iint\limits_{T_j}P(C-B(x,y))B_x(x,y)\,dxdy,$$
$$\bar{S}_j^{(3)}\approx-\frac{g}{|T_j|\rho_0}\iint\limits_{T_j}P(C-B(x,y))B_y(x,y)\,dxdy.$$

In order to derive a well-balanced quadrature, similar to [4, 29], we first apply Green's formula, $\iint_{T_j}\operatorname{div}\mathcal{G}\,dxdy=\int_{\partial T_j}\mathcal{G}\cdot\mathbf{n}\,ds$, to the vector field $\mathcal{G}=(\frac{1}{2}\rho(x,y)(w(x,y)-B(x,y))^2,0)$ and obtain

$$\begin{aligned}
&-\iint\limits_{T_j}\rho(x,y)(w(x,y)-B(x,y))B_x(x,y)\,dxdy\\
&=\sum_{k=1}^{3}\int\limits_{(\partial T_j)_k}\rho(x,y)\frac{(w(x,y)-B(x,y))^2}{2}\cos(\theta_{jk})\,ds\\
&-\iint\limits_{T_j}\rho(x,y)(w(x,y)-B(x,y))w_x(x,y)\,dxdy\\
&-\iint\limits_{T_j}\rho_x(x,y)\frac{(w(x,y)-B(x,y))^2}{2}\,dxdy,
\end{aligned} \tag{27}$$

where $(\partial T_j)_k$ is the k-th side of the triangle T_j, $k=1,2,3$. The double integrals are approximated using the trapezoidal rule. This results in the following quadrature for $\bar{S}_j^{(2)}$:

$$\begin{aligned}
\bar{S}_j^{(2)}&=\frac{g}{2|T_j|}\sum_{k=1}^{3}\ell_{jk}\cos(\theta_{jk})\cdot\frac{\Delta t_{jk}^{(\rho)}}{\Delta t}\cdot\frac{\rho(M_{jk})}{\rho_0}\big[w(M_{jk})-B(M_{jk})\big]^2\\
&-\frac{g}{3\rho_0}\Big[\rho_{j12}(w_{j12}-\widehat{B}_{j12})\,w_x(V_{12})+\rho_{j23}(w_{j23}-\widehat{B}_{j23})\,w_x(V_{23})\\
&+\rho_{j13}(w_{j13}-\widehat{B}_{j13})\,w_x(V_{j13})\Big]-\frac{g}{6\rho_0}\Big[\rho_x(V_{j12})(w_{j12}-\widehat{B}_{j12})^2\\
&+\rho_x(V_{j23})(w_{j23}-\widehat{B}_{j23})^2+\rho_x(V_{j13})(w_{j13}-\widehat{B}_{j13})^2\Big].
\end{aligned} \tag{28}$$

A similar quadrature for $\bar{S}_j^{(3)}$ is

$$
\begin{aligned}
\bar{S}_j^{(3)} &= \frac{g}{2|T_j|}\sum_{k=1}^{3}\ell_{jk}\sin(\theta_{jk})\cdot\frac{\Delta t_{jk}^{(\rho)}}{\Delta t}\cdot\frac{\rho(M_{jk})}{\rho_0}\big[w(M_{jk})-B(M_{jk})\big]^2\\
&-\frac{g}{3\rho_0}\Big[\rho_{j12}(w_{j12}-\widehat{B}_{j12})\,w_y(V_{j12})+\rho_{j23}(w_{j23}-\widehat{B}_{j23})\,w_y(V_{j23})\\
&+\rho_{j13}(w_{j13}-\widehat{B}_{j13})\,w_y(V_{j13})\Big]-\frac{g}{6\rho_0}\Big[\rho_y(V_{j12})(w_{j12}-\widehat{B}_{j12})^2\\
&+\rho_y(V_{j23})(w_{j23}-\widehat{B}_{j23})^2+\rho_y(V_{j13})(w_{j13}-\widehat{B}_{j13})^2\Big].
\end{aligned}
\tag{29}
$$

Notice that the piecewise linear reconstruction procedure ensures that at the steady state (24), $\nabla\rho(V_{j\kappa})=0, u_x=v_y=0$ and $\nabla w(V_{j\kappa})=0$ throughout the entire computational domain. This implies that $(w_{j\kappa}-\widehat{B}_{j\kappa})\,w_x(V_{j\kappa})\equiv(w_{j\kappa}-\widehat{B}_{j\kappa})\,w_y(V_{j\kappa})\equiv 0$ and $\rho_x(V_{j\kappa})(w_{j\kappa}-\widehat{B}_{j\kappa})^2\equiv\rho_y(V_{j\kappa})(w_{j\kappa}-\widehat{B}_{j\kappa})^2\equiv 0$. Therefore, the quadratures (28) and (29) satisfy the desired well-balanced requirements (25) and (26).

Next, we consider the "lake at rest" situation

$$
B\equiv\text{Const},\quad \mathrm{h}^2\rho=\mathrm{Q}\equiv\text{Const},\quad \mathrm{u}\equiv\mathrm{v}\equiv 0.
\tag{30}
$$

Note that in the region occupied by only one fluid, the density ρ is a constant and water surface w is then also a constant based on (30). Hence, in single-fluid cells, the steady state (30) is equivalent to the solution (24) which is maintained by the discretization of source term (28)-(29). In addition, in mixed cells, the point values are obtained by the data from nearby single-fluid cells thanks to the Riemann Solver approximation presented in Section 2.2, see also [9]. Therefore, the discretization of source term (28)-(29), is also capable of preserving the steady state solution (30).

3. Interface Tracking

To achieve an efficient adaptive scheme for multi-fluid flows, it is essential to accurately capture the curve where two types of fluid join simultaneously with the flow field evolution. The interface tracking is important in preventing excessive numerical diffusion of variable density, see Section 3.3. We also need the location of density jumps to exactly update the cell averages in the adaptive mesh reconstruction, Section 4. Therefore, we desire a simple and effective interface reconstruction for the system (1a)-(1d). Over the last two decades, many methods have been proposed for this purpose such as the level set method [32, 39], the volume of fluid method [17, 35], and the front tracking method [45, 44]. Among various versions of interface tracking, we have considered the approach described in [49, 48] for our work due to its simplicity, robustness, and high efficiency. In particular, the interface is obtained by using both the level set and the volume fraction functions. The details of the interface reconstruction is described as follows.

3.1. Mixed Cell Detecting by Level Set Function. In the first part of this section, we will discuss the method of detecting the mixed cells in the triangular grids. For this end, we consider the level set function ϕ which is defined such that it is positive in one fluid, is negative in the other fluid, and has zero value at the interface. Very often, the level set value of each grid point is initialized by the signed distance from that point to the curve of density discontinuities, see [29, 32, 39, 49].

As discussed in [29, 32, 39, 49], level set function is evolved by the velocity (u, v) of the flow field as

$$\phi_t + u\phi_x + v\phi_y = 0. \tag{31}$$

The equation (31) can be rewritten in conservation form as follows.

$$\phi_t + (u\phi)_x + (v\phi)_y = (u_x + v_y)\phi. \tag{32}$$

We have used the central-upwind scheme presented in Section 2 to solve for ϕ, where the point values $\phi(M_{jk}), k = 1, 2, 3$ in triangle T_j are approximated by the piecewise linear reconstruction (10). The integral of the source term in (32) can be approximated by midpoint rule. Note that using the central-upwind scheme to solve (31) only give us the cell averages of ϕ in each cell and does not point out which cells contain the interface $\phi = 0$. We have to provide a numerical method for detecting mixed cells. However, a triangle is a single-fluid cell, also called as "reliable" cell, if the level set values at its vertices are either all positive or all negative. Hence, we will locate each node in the grid based on its point value of level set as presented below.

Without the loss of generality, we assume that $\phi(x, y)$ is positive in fluid 1 and $\phi(x, y)$ is negative in fluid 2. The point value of level set function $\phi^*_{j\kappa}$ at each vertex $V_{j\kappa}$ of cell T_j is approximated by extrapolation [9]

$$\phi^*_{j\kappa} = \frac{\sum_{i=0}^{m^*_\kappa} c_i \phi^i_{j\kappa}}{\sum_{i=0}^{m^*_\kappa} c_i}, \tag{33}$$

where $\phi^i_{j\kappa}$, is the cell averages of level set function in cells $T^i_{j\kappa}$ which have common vertex at $V_{j\kappa}$, and the weight c_i is inversely proportional to the distance between the center of cells $T^i_{j\kappa}, i = 0, 1, ..., m^*_\kappa$ and vertex $V_{j\kappa}$. If $\phi^*_{j\kappa} > 0$, we have the vertex $V_{j\kappa}$ staying in fluid 1. Otherwise, if $\phi^*_{j\kappa} < 0$, the vertex $V_{j\kappa}$ is in fluid 2. Finally, based on the physical meaning of the level set function, a triangle T_j is naturally marked as a mixed cell if it has two vertices staying in different fluids.

3.2. Interface Reconstruction – An overview. In most works of interface tracking, see [32, 39, 17, 35, 45, 44], the curve of density discontinuity in a mixed cell is approximated as a linear line segment of the form $\mathbf{n} \cdot x = \alpha$, where $\mathbf{n}$ is the normal vector of the interface, x is the location of a point, and α is the line constant. A variety of methods for computing normal vector $\mathbf{n}$ and parameter α have been briefly introduced and compared in [49, 34]. Based on the advantages and disadvantages of conventional interface tracking methods, an innovative approach has been proposed in [49]. The approach employs both the level set and volume of fluid methods to denote the density segment in each flagged cell by its two endpoints. Namely, the interface normal vector is calculated from the level set function while the exact location of two endpoints of the segment is determined by enforcing mass conservation based on the volume fraction. In addition, the governing equations of the level set functions and the volume of fluid function, see equations (31) and (35), are discretized conservatively. Hence, the interface reconstruction exactly conserves the volume of fluid. In our work, we will apply this interface tracking method in the adaptive triangular mesh due to its simplicity, accuracy, and versatility. In the

following, we will give a brief overview of the interface reconstruction originally developed in [49]. More details of this method can be seen in [49].

Once the mixed cells are detected, we will first compute the interface normal vector in each flagged cell based on the level set function and a least square problem. Namely, for each mixed cell T_j with center at (x_j, y_j), we consider a stencil that consists of all centers $(x_i, y_i), i = 1, 2, ..., N$ of N cells that share at least a vertex with T_j (to simplify the computation, we do not place the vertices of cell T_j in the stencil as in [49]). A quadratic function for function ϕ of a point (x, y) is then given in the generic form

$$\phi = ax'^2 + bx'y' + cy'^2 + dx' + ey' + g,$$

where $(x' := x - x_j, y' := x - x_j)$ is the location of point (x, y) in the local coordinate frame x' and y' obtained by shifting the original coordinate such that the new origin is at the center of T_j. The coefficients a, b, c, d, e, and g are determined by using the least squares method with the linear system

$$Qs = r,$$

where

$$Q = \begin{bmatrix} x_1'^2 & x_1'y_1' & y_1'^2 & x_1' & y_1' & 1 \\ x_2'^2 & x_2'y_2' & y_2'^2 & x_2' & y_2' & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_N'^2 & x_N'y_N' & y_N'^2 & x_N' & y_N' & 1 \end{bmatrix}, \quad s = \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ g \end{bmatrix}, \quad \phi = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_n \end{bmatrix}, \tag{34}$$

$(x_i' = x_i - x_j, y_i' = y_i - y_j), i = 1, ..., N$, are the local coordinates of the $i-$th node in the stencil and $\phi_i, i = 1, ..., N$, is the cell average of level set function in cell T_i corresponding to node i-th. Once the coefficients are known, we compute the unit normal vector by $\mathbf{n} = \left(\frac{d}{\sqrt{d^2 + e^2}}, \frac{e}{\sqrt{d^2 + e^2}} \right)$. The system (34) is solvable as explained in [49]. In addition, since the level set function is continuous, the normal calculation is second-order accurate.

Next, we continue to determine the exact coordinates of two endpoints of the interface by using the volume of fluid approach (VOF). The VOF method has been widely used and developed for interface tracking in [17, 35, 48, 49]. In the VOF method, the volume fraction function, denoted by f, is defined as the ratio of the volume of one fluid, called fluid 1, in each cell to the total volume of the cell. Hence, f is unity if the cell is a single-fluid cell in fluid 1, and is zero if the cell only contains fluid 2. For mixed cells, we have $0 < f < 1$. The function f is advected by

$$f_t + uf_x + vf_y = 0. \tag{35}$$

The main idea of the VOF method proposed in [48, 49] is to reconstruct the interface segment by determining the coordinates of its endpoints. In particular, the two endpoints must form a segment which has the normal vector $\mathbf{n}$ as computed above and the interface truncates the cell with the given volume fraction. Fig. 3.3 is an illustration for two cases of a mixed cell T_j where the interface I_1I_2 splits T_j into two parts, $T_j^{(1)}$ and $T_j^{(2)}$, respectively occupied by fluid 1 and fluid 2.

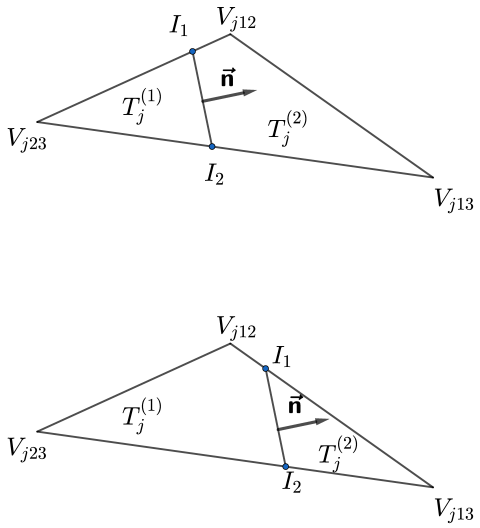


Figure 3.3. An example of mixed cell $T_j = V_{j12}V_{j23}V_{j13}$ with the interface segment I_1I_2 and the normal vector $\vec{\mathbf{n}}$.

We then have

$$I_1I_2 \cdot \mathbf{n} = 0, \quad \frac{T_j^{(1)}}{T_j} = \overline{f}_j,$$

where $\mathbf{n}$ is the normal vector of the interface computed by (34) and $\overline{f}_j$ is the cell average of volume fraction in mixed cell T_j obtained by using the central-upwind method, see Section 2, to solve (35). This interface reconstruction is explicit, accurate, and capable of conserving the volume of the fluid. The details of endpoint calculation can be seen in [49]. Finally, the reconstructed interface will be used in the adaptive mesh reconstruction as discussed in the following sections.

3.3. The Cell Averages Correction. The idea of correcting the solution in the neighborhood of the interface has been used in numerical methods for multifluid flows [9, 49] to improve the computed solution. This technique prevents the diffusion of density emerging when we use the central-upwind method to solve compressible systems. In our work, we will also perform the correction procedure such that the local conservation is ensured based on the location of density jumps. Namely, suppose we have two types of fluid, fluid 1 and fluid 2, in the flow. After determining the single-fluid cells and mixed cells by using the point values of the level set function, see Section 3.1, the cell average correction will proceed as follows.

- If a cell T_j is a single fluid cell in fluid $i = 1, 2$, we correct the cell averages of $h\rho$ by $\overline{h\rho}_j^{new} = h_j\rho_j^{(i)}$, where $\rho_j^{(i)}$ is the value of density in fluid $i = 1, 2$. $h_j = \overline{w}_j - B_j$ is the cell average of water depth in T_j. To preserve the mass of $h\rho$ in the domain, we equally split the change of $\overline{h\rho}$ in T_j into the nearby mixed cells. Namely, if the neighboring cell $T_{jk}, k = 1, 2, 3$ of T_j is a mixed cell, we obtain the new cell average of $h\rho$ in T_{jk} by

$$\overline{h\rho}_{jk}^{new} = \overline{h\rho}_{jk} + \frac{\overline{h\rho}_j - \overline{h\rho}_j^{new}}{n_{mix}}, \tag{36}$$

where $\overline{h\rho}_j$ and $\overline{h\rho}_{jk}$ are the old cell averages computed by using the central-upwind scheme in triangles T_j and T_{jk}, and n_{mix} is the number of mixed cells surrounding the cell T_j.

- If T_j is a mixed cell, except the update (36) from its nearby single-fluid cells, no further correction is needed.

Due to the fact that the interface normally moves from one cell to its adjacent cells, only cells surrounding the interface are updated and the loss of mass from this correction is negligible. In addition, for cells in the "lake at rest" area, this correction will not change the existing solution therefore maintaining the well-balanced properties.

4. Adaptive Central-Upwind Scheme –An Overview

The traditional numerical methods for system (1a)–(1d) consider very fine fixed meshes to reconstruct delicate features of the solution. However, this can lead to high computational cost, as well as to a poor accuracy of small scale characteristics of the problem. Therefore, we will use adaptive meshes to improve the accuracy of the approximation at a much lower cost. More specifically, we apply the adaptive techniques in space and time originally developed in our prior work [13]. Some adjustments are made on the adaptive time restriction and error indicator in order to ensure the stability and the efficiency of the adaptive scheme on the considered system (1a)-(1d). In this section, we will briefly review the adaptive algorithm in [13] which we adapt to the system (1a)-(1d).

4.1. Adaptive Central-Upwind Algorithm. From [13], the adaptive central-upwind algorithm for the system (1a)- (1d) is described briefly by the following steps.

Step 0. At time $t = t^0$, generate the initial uniform grid $\mathcal{T}^{0,0}$.

Step 1. On mesh $\mathcal{T}^{n,\mathcal{M}_n}$, evolve the cell averages $\overline{\boldsymbol{U}}^n$ of the solution from time t^n to $\overline{\boldsymbol{U}}^{n+1}$ at the next time level t^{n+1} using adaptive central-upwind scheme (41a)-(41b), see Section 4:

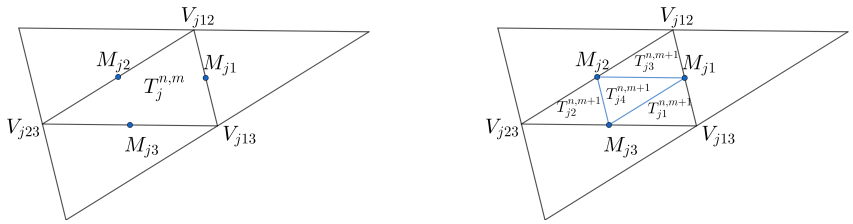
- At time t^n, determine the level $l = 0, 1, ..., L$ of each triangle $T_j^{n,\mathcal{M}_n} \in \mathcal{T}^{n,\mathcal{M}_n}$, (37), Section 4.3.
- At each time level $t_l^{n,p}, p = 0, 1, ..., \mathcal{P}_l - 1$, perform the piecewise polynomial reconstruction (10) for single-fluid cells and apply the Riemann Solver (17) on the mixed cells to calculate the point values, Section 2, Section 3.3.
- At each time level $t_l^{n,p}, p = 0, 1, ..., \mathcal{P}_l - 1$, compute the one-sided local speeds of propagation using (11), Section 2.
- At time t^n, calculate the reference time step Δt using (38), Section 4.3.
- At each time level $t_l^{n,p}, p = 0, 1, ..., \mathcal{P}_l - 1$, compute the local time step for each cell level, (40), Section 4.3.
- At each time level $t_l^{n,p}, p = 0, 1, ..., \mathcal{P}_l - 1$, compute numerical fluxes and source term in the adaptive central-upwind scheme (41a)-(41b), (9), (28)-(29), Section 2, Section 4.3.

Step 2. On mesh $\mathcal{T}^{n,\mathcal{M}_n}$, compute WLR error using (45) in Section 4.4 and determine the refinement/de-refinement status for each cell, Section 4.4.

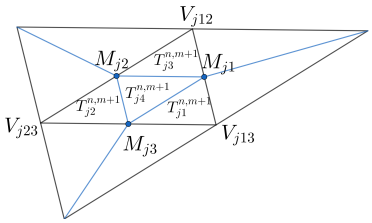
Step 3. Generate the new adaptive mesh $\mathcal{T}^{n+1,\mathcal{M}_{n+1}}$ at t^{n+1}, Section 4.2. This step includes coarsening of some cells, refinement of some cells, and the appropriate projection of the cell averages from the mesh $\mathcal{T}^{n,\mathcal{M}_n}$ at t^n onto a new adaptive mesh $\mathcal{T}^{n+1,\mathcal{M}_{n+1}}$ at time t^{n+1}, Section 4.2.

Step 4. Repeat **Step 1** - **Step 3** until final time.

4.2. Adaptive Mesh Refinement/Coarsening. The purpose of mesh reconstruction is to obtain adaptive grids which delivers higher accuracy with a lower computational cost. An efficient adaptive mesh should have small cells in the regions with large errors and large cells in the other regions. In particular, at time t^n, we start with the given mesh, denoted as $\mathcal{T}^{n,m} = \{T_j^{n,m}\}$, where $T_j^{n,m}$ is a triangular cell with the barycenter $(x_j^{n,m}, y_j^{n,m})$, and index $m = 0, 1, 2...$ is the level of refinement ($\mathcal{T}^{n,0} \equiv \mathcal{T}^{0,0}$ for all n and $\mathcal{T}^{0,0}$ represents the initial mesh with no refinement). The triangular cells in the mesh $\mathcal{T}^{n,m}$ are flagged for refinement/de-refinement based on the weak local residual (WLR) error estimator, see Section 4.4. On grid $\mathcal{T}^{n,m}$, we apply "regular refinement" described in [13] on the triangles flagged for refinement to obtain a new mesh $\mathcal{T}^{n,m+1}$ with the refinement level $m+1$. Namely, each flagged triangle ("parent" triangle) is split into four smaller triangles ("children" triangles) by inserting a new node at the mid-point of each edge of the "parent" triangle. Fig. 4.4 (a) is an illustration of the "regular refinement", where a flagged cell $T_j^{n,m}$ is refined to obtain the "children" cells $T_{j_s}^{n,m+1}, s = 1, 2, 3, 4$ by using the mid-points of the sides. In addition, due to the insertion of new nodes on the edges of the non-flagged triangles adjacent to refined triangles, we must also refine these neighboring cells by creating a new edge between the hanging node and the opposite corner as illustrated in Fig. 4.4 (b).



(a) Triangle $T_j^{n,m}$ (left) is split into four "children" cells $T_{js}^{n,m+1}, s = 1, 2, 3, 4$ (right).



(b) Refinement in the neighboring cells of $T_j^{n,m}$.

Figure 4.4. An outline of the "regular refinement".

In practice, there may be some cells having very large WLR error (45), and we need to reach a higher level of refinement for those cells to improve the accuracy. This can be done by repeating the refinement for the flagged triangles in the refined mesh $\mathcal{T}^{n,m+1}$ to get the mesh with higher level $\mathcal{T}^{n,m+2}, m = 0, 1, 2, ...$, see [13] for the illustration of the multi-level refinement.

Note that in the numerical simulations of the wave phenomena, the regions of the domain flagged for refinement changes over time evolution and the refinement in some cells may become no longer needed. Hence, in [13], the de-refinement/coarsening procedure is performed to deactivate unnecessarily fine cells in the grid. Namely, at time t^n, we deactivate "children" cells in the mesh $\mathcal{T}^{n,m+1}, m = 0, 1, ..., M_n - 1$ based on the WLR and activate the corresponding "parent" cell from the mesh $\mathcal{T}^{n,m}$ back. More details of refinement/de-refinement process can be seen in [13].

Finally, at time t^n, a hierarchical system of grids $\mathcal{S}^n = \{\mathcal{T}^{n,0}, \mathcal{T}^{n,1}, \mathcal{T}^{n,2}, ..., \mathcal{T}^{n,\mathcal{M}_n}\}$ is obtained, where $\mathcal{T}^{n,m}$, $m = 1, 2, ..., \mathcal{M}_n$ is the grid with the level of refinement m reconstructed from the grid $\mathcal{T}^{n,m-1}$. We will use the final mesh $\mathcal{T}^{n,\mathcal{M}_n} \in \mathcal{S}^n$ for adaptive central-upwind scheme (41a)-(41b) to evolve the numerical solution from time t^n to time t^{n+1}. Next, at t^{n+1}, we generate a new adaptive grid $\mathcal{T}^{n+1,\mathcal{M}_{n+1}} \in \mathcal{S}^{n+1}$ from the mesh $\mathcal{T}^{n,\mathcal{M}_n}$. After the mesh reconstruction at time t^{n+1}, the obtained cell averages $\overline{\boldsymbol{U}}^{n+1}$ on the mesh $\mathcal{T}^{n,\mathcal{M}_n}$ need to be accurately projected on the new mesh $\mathcal{T}^{n+1,\mathcal{M}_{n+1}}$, using the ideas as summarized briefly below.

Case 1. At t^{n+1}, a triangle $T_j^{n+1,\mathcal{M}_{n+1}} \in \mathcal{T}^{n+1,\mathcal{M}_{n+1}}$ is the same cell as in the grid $\mathcal{T}^{n,\mathcal{M}_n}$, we will maintain the cell averages for that triangle at t^{n+1}.

Case 2. A cell $T_j^{n+1,\mathcal{M}_{n+1}} \in \mathcal{T}^{n+1,\mathcal{M}_{n+1}}$ is obtained by de-refining some finer cells $T_{j_s}^{n,\mathcal{M}_n} \in \mathcal{T}^{n,\mathcal{M}_n}, s = 1, 2, .., S$. The cell average of solution, $\overline{U}_j^{n+1}$ in the cell $T_j^{n+1,\mathcal{M}_{n+1}}$, is computed as

$$\overline{U}_j^{n+1} = \frac{1}{|T_j^{n+1,\mathcal{M}_{n+1}}|} \sum_{s=1}^{S} \overline{U}_{j_s}^n |T_{j_s}^{n,\mathcal{M}_n}|,$$

where $\overline{U}_{j_s}^n$ is the solution in $T_{j_s}^{n,\mathcal{M}_n}$.

Case 3. A triangle $T_j^{n+1,\mathcal{M}_{n+1}} \in \mathcal{T}^{n+1,\mathcal{M}_{n+1}}$ is obtained from the refinement of the cell $T_i^{n,\mathcal{M}_n} \in \mathcal{T}^{n,\mathcal{M}_n}$. If $T_i^{n,\mathcal{M}_n}$ is a single-fluid cell, the cell averages at t^{n+1} in $\mathcal{T}^{n+1,\mathcal{M}_{n+1}}$ are approximated by using the the piecewise linear reconstruction (10) of the solution at t^{n+1} in the triangle $T_i^{n,\mathcal{M}_n}$. If $T_i^{n,\mathcal{M}_n}$ contains the density discontinuity, we will compute the cells averages in the "son" cell $T_j^{n+1,\mathcal{M}_{n+1}}$ based on the interface tracking, see Section 3.2, and the information of the nearby single-fluid cells. For example, suppose that from the reconstructed interface in "dad" cell $T_i^{n,\mathcal{M}_n}$, the "child" cell $T_j^{n+1,\mathcal{M}_{n+1}}$ completely lies in one fluid, called fluid 1. Hence, triangle $T_j^{n+1,\mathcal{M}_{n+1}}$ is a single-fluid cell and the cell averages $\overline{U}_j^{n+1}$ are set to be equal to the cell averages of another "reliable" cell which is in fluid 1 and is the closest cell to the "dad" cell $T_i^{n,\mathcal{M}_n}$.

Remark:

The update of cell averages in case 3 does not ensure the mass conservation which means the total mass of "son" cells in the adaptive grid is not equal to the mass of their "dad" cell. However, the volume of fluid and the exact location of the interface are conserved. In addition, the reconstructing method also maintains the steady state solution for "lake at rest" situations, see example 2 Section 5, since the values of the cell averages are obtained by using the information from nearby "reliable" cells.

4.3. Second-order Adaptive Time Evolution. Note that using a global time step in the adaptive algorithm may lead to a very small time step due to the presence

of much finer cells in the mesh. To reduce the computational cost, we consider the approach based on the adaptive time step from [13, 11, 12, 30]. The main idea of this approach is to group cells into different levels based on the cell sizes and applying local time step on each level to evolve from t^n to t^{n+1}. Recently, in our work on the shallow water model [13], we have derived a simple and efficient adaptive time evolution algorithm based on the second-order strong stability preserving Runge-Kutta methods (SSPRK2) in [16, 11, 12, 30]. This method is also capable to perform on the multi-fluid system (1a)-(1b). The algorithm can be briefly described by an example as illustrated in Fig. 4.5.

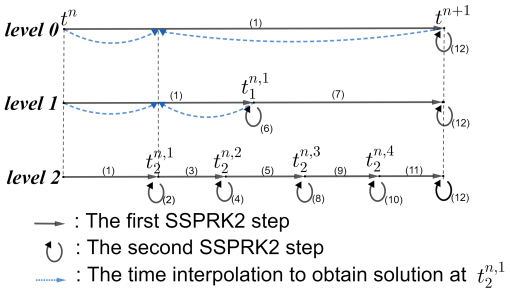


Figure 4.5. The example of SSPRK2 on mesh with three cell levels, $l = 0, 1, 2$.

First, we group all cells in the grid $\mathcal{T}^{n,\mathcal{M}_n}$ at time t^n in cell levels $l = 0, 1, .., L$ based on their sizes. Namely, a cell $T_j^{n,\mathcal{M}_n} \in \mathcal{T}^{n,\mathcal{M}_n}$ belongs to the level l, if l is the smallest positive integer satisfying,

$$2^l \geq \frac{\max\limits_j \left(\min\limits_k (r_{jk}) \right)}{\min\limits_k (r_{j,k})}, \tag{37}$$

where $r_{jk}, k = 1, 2, 3$ are three altitudes of triangle $T_j^{n,\mathcal{M}_n}$. Next, at t^n, we define the reference time step Δt as the local time step on the coarsest level $l = 0$ of cells in the mesh $\mathcal{T}^{n,\mathcal{M}_n}$ by considering the CFL-type condition (23) locally on level $l = 0$.

$$\Delta t \equiv \Delta t_0^{n,0} = \frac{0.9 \max\limits_j \left(\min\limits_k (r_{jk}) \right)}{18 a_{max}}, \tag{38}$$

where

$$a_{max} := \max_{j,k}(a_{jk}^{\text{in}}, a_{jk}^{\text{out}}), \tag{39}$$

and $(a_{jk}^{\text{in}}, a_{jk}^{\text{out}})$ are the local one-sided speeds of propagation (11) at t^n for sides $k = 1, 2, 3$ in the triangle $T_j^{n,\mathcal{M}_n} \in \mathcal{T}^{n,\mathcal{M}_n}$. We set, $t^{n+1} = t^n + \Delta t$.

Next, assume that $\mathcal{P}_l$ is the number of steps needed for the higher levels $l = 1, ..L$ to evolve from t^n to t^{n+1}, namely $[t^n, t^{n+1}] = \cup [t_l^{n,p}, t_l^{n,p+1}], p = 0, ..., \mathcal{P}_l - 1$ with $t_l^{n,0} \equiv t^n$, $t_l^{n,\mathcal{P}_l} \equiv t^{n+1} \quad \forall l$. At $t_l^{n,p}$, the local time step for cells on these levels $l = 1, ..., L$ is calculated by

$$\Delta t_l^{n,p} = \frac{2^{-l} \Delta t}{\max(\mu_l^{n,p}, 1)}, \tag{40}$$

where parameter $\mu_l^{n,p}$ takes into account change in the local one-sided speeds of the propagation,

$$\mu_l^{n,p} = \frac{\max\limits_{j,k}(a_{jk}^{\text{in}}, a_{jk}^{\text{out}})_l^{n,p}}{a_{max}},$$

where $(a_{jk}^{\text{in}}, a_{jk}^{\text{out}})_l^{n,p}$ are the local one-sided speeds of propagation of the cell $T_j^{n,\mathcal{M}_n}$ in the level l at $t_l^{n,p}$. Therefore, on each cell $T_j^{n,\mathcal{M}_n}$ of level l, for each substep $[t_l^{n,p}, t_l^{n,p+1}] \equiv [t_l^{n,p}, t_l^{n,p} + \Delta t_l^{n,p}], p = 0, 1, 2, 3, ..., \mathcal{P}_l - 1$ of the evolution from t^n to t^{n+1}, we apply the following two adaptive steps of the SSPRK2 method, see [16, 11, 12, 30],

$$\text{(41a)} \qquad \overline{\boldsymbol{U}}_j^{(1)} = \overline{\boldsymbol{U}}_j^{n,p} - \Delta t_l^{n,p}\left(\frac{1}{|T_j^{n,\mathcal{M}_n}|}\sum_{k=1}^{3}\boldsymbol{H}_{jk}^{n,p} - \overline{\boldsymbol{S}}_j^{n,p}\right) := \boldsymbol{R}(\overline{\boldsymbol{U}}_j^{n,p}, \Delta t_l^{n,p}),$$

$$\text{(41b)} \qquad \overline{\boldsymbol{U}}_j^{n,p+1} = \frac{1}{2}\overline{\boldsymbol{U}}_j^{n,p} + \frac{1}{2}\boldsymbol{R}(\overline{\boldsymbol{U}}_j^{(1)}, \Delta t_l^{n,p}).$$

The flux term $\boldsymbol{H}_{jk}^{n,p}$ in (41a) -(41b) is the flux (9) computed at $t = t_l^{n,p}$. The source term $\overline{\boldsymbol{S}}_j^{n,p}$ in (41a) -(41b) is the source (28) - (29) computed at $t = t_l^{n,p}$ with the time step $\Delta t_l^{n,p}$. Note that, $\overline{\boldsymbol{U}}_j^{n,0} \equiv \overline{\boldsymbol{U}}_j^n$ and $\overline{\boldsymbol{U}}_j^{n,\mathcal{P}_l} \equiv \overline{\boldsymbol{U}}_j^{n+1}$.

Remark:

If cells from different cell levels are neighbors, we use linear interpolation in time to match the time levels of such cells, see also Fig. 3.6, for the illustration of the interpolation.

4.4. A Posteriori Error Estimator. To create a robust indicator for the adaptive mesh refinement, in our prior work [13], we have derived local error estimator from the idea of Weak Local Residual (WLR) presented in [43, 19]. This error indicator has shown its advantages in accurately capturing the regions with large error in numerical simulation for Saint-Venant system of shallow water model. Hence, for the adaptive central-upwind scheme for SWEDs, we will extend the error estimator by applying the computation performed in [13] for the last equation in the system (1d).

Let us recall that from the weak form of the mass conservation equation (1a), in [13], the WLR errors $E_i^{w,n+\frac{1}{2}}$ at each node N_i on mesh $\mathcal{T}^{n,\mathcal{M}_n}$ is given by the formula,

$$\begin{aligned}
E_i^{w,n+\frac{1}{2}} &= \frac{1}{\Delta}(\mathcal{U}_i^{w,n+\frac{1}{2}} + \mathcal{F}_i^{w,n+\frac{1}{2}} + \mathcal{G}_i^{w,n+\frac{1}{2}}),\\
\mathcal{U}_i^{w,n+\frac{1}{2}} &= \sum_{c=1}^{C_i}\frac{1}{3}|T_{j_c}^{n,\mathcal{M}_n}|(\overline{w}_{j_c}^n - \overline{w}_{j_c}^{n+1}),\\
\mathcal{F}_i^{w,n+\frac{1}{2}} &= \sum_{c=1}^{C_i} a_c^{(i)}\frac{\Delta t}{2}|T_{j_c}^{n,\mathcal{M}_n}|((\overline{hu})_{j_c}^n + (\overline{hu})_{j_c}^{n+1}),\\
\mathcal{G}_i^{w,n+\frac{1}{2}} &= \sum_{c=1}^{C_i} b_c^{(i)}\frac{\Delta t}{2}|T_{j_c}^{n,\mathcal{M}_n}|((\overline{hv})_{j_c}^n + (\overline{hv})_{j_c}^{n+1}),
\end{aligned} \tag{42}$$

where C_i is the number of triangles $T_{j_c}^{n,\mathcal{M}_n}$ having common vertex at node N_i. Here, the quantity $(a_c^{(i)}, b_c^{(i)})$ is the gradient of the linear piece restricted to $T_{j_c}^{n,\mathcal{M}_n}$,

$$\begin{aligned} a_c^{(i)} &= \frac{\widetilde{y}_2 - \widetilde{y}_3}{(\widetilde{y}_3 - \widetilde{y}_i)(\widetilde{x}_2 - \widetilde{x}_i) - (\widetilde{y}_2 - \widetilde{y}_i)(\widetilde{x}_3 - \widetilde{x}_i)}, \\ b_c^{(i)} &= \frac{\widetilde{x}_3 - \widetilde{x}_2}{(\widetilde{y}_3 - \widetilde{y}_i)(\widetilde{x}_2 - \widetilde{x}_i) - (\widetilde{y}_2 - \widetilde{y}_i)(\widetilde{x}_3 - \widetilde{x}_i)}, \end{aligned} \tag{43}$$

where $N_i = (\widetilde{x}_i, \widetilde{y}_i), (\widetilde{x}_2, \widetilde{y}_2)$, and $(\widetilde{x}_3, \widetilde{y}_3)$ are the three vertices of triangle $T_{j_c}^{n,\mathcal{M}_n}$.

Now, by applying the same calculation in [13] on the weak form of the last equations in the system (1d), we then define the WLR error of variable $h\rho$ at node N_i in the grid as,

$$\begin{aligned} E_i^{h\rho,n+\frac{1}{2}} &= \frac{1}{\Delta}(\mathcal{U}_i^{h\rho,n+\frac{1}{2}} + \mathcal{F}_i^{h\rho,n+\frac{1}{2}} + \mathcal{G}_i^{h\rho,n+\frac{1}{2}}), \\ \mathcal{U}_i^{h\rho,n+\frac{1}{2}} &= \sum_{c=1}^{C_i} \frac{1}{3}|T_{j_c}^{n,\mathcal{M}_n}|(\overline{h\rho}_{j_c}^{n} - \overline{h\rho}_{j_c}^{n+1}), \\ \mathcal{F}_i^{h\rho,n+\frac{1}{2}} &= \sum_{c=1}^{C_i} a_c^{(i)} \frac{\Delta t}{2}|T_{j_c}^{n,\mathcal{M}_n}|((\overline{hu\rho})_{j_c}^{n} + (\overline{hu\rho})_{j_c}^{n+1}), \\ \mathcal{G}_i^{h\rho,n+\frac{1}{2}} &= \sum_{c=1}^{C_i} b_c^{(i)} \frac{\Delta t}{2}|T_{j_c}^{n,\mathcal{M}_n}|((\overline{hv\rho})_{j_c}^{n} + (\overline{hv\rho})_{j_c}^{n+1}). \end{aligned} \tag{44}$$

Hence, the error in a cell $T_j^{n,\mathcal{M}_n} \in \mathcal{T}^{n,\mathcal{M}_n}$ takes into account both WLR errors of water surface w and of variable $h\rho$ as,

$$e_j = \max_k \left(\left| E_{jk}^{w,n+\frac{1}{2}} \right|, \left| E_{jk}^{h\rho,n+\frac{1}{2}} \right| \right), \quad k = 1, 2, 3, \tag{45}$$

where $E_{jk}^{w,n+\frac{1}{2}}$ and $E_{jk}^{h\rho,n+\frac{1}{2}}$ are the WLR errors computed in (42) and (44) at a node k of triangle T_j.

The error e_j in each cell $T_j^{n,\mathcal{M}_n} \in \mathcal{T}^{n,\mathcal{M}_n}$ is compared to the error tolerance computed by

$$\omega = \sigma \max_j(e_j), \tag{46}$$

where $\sigma < 1$ is a given problem-dependent constant. Based on the error comparison, the cell is then either "flagged" for refinement/de-refinement or "no-change".

5. Numerical Experiments

In this section, we will verify the computational efficiency of the designed adaptive central-upwind scheme. We compare the results of the adaptive method with the results of the central-upwind scheme without the adaptivity, see Section 4, on uniform triangular meshes (example of such uniform triangular mesh is outlined in Fig. 5.6). To this end, in all examples, we calculate the L^1-errors and a ratio, $\mathcal{R}_{CPU} = \frac{CPU_{uniform}}{CPU_{adaptive}}$, which is the ratio of the CPU times of the central-upwind algorithm without the mesh reconstruction to the CPU time of the adaptive central-upwind algorithm. To compare L^1-errors, as well as to compare the CPU times and to compute $\mathcal{R}_{CPU}$, we consider uniform mesh and the adaptive mesh with the same size of the smallest cells. Namely, in Tables 5.1-5.8, L^1-errors and $\mathcal{R}_{CPU}$ are computed by using the uniform meshes $2 \times N \times N, N = 100, 200, 400$

and the corresponding adaptive meshes which are reconstructed from the coarser uniform mesh $2 \times N/2^{\mathcal{M}} \times N/2^{\mathcal{M}}$ ($\mathcal{M} = 1, 2$ is the highest level of refinement in the adaptive mesh). To compute the L^1-errors, the reference solution is obtained by applying the central-upwind method without implementing adaptivity techniques on the uniform mesh with $2 \times 800 \times 800$ triangles. In all experiments, we consider a zero-order extrapolation at all boundaries. In addition, we use the gravitational acceleration, $g = 1.0$ and the reference density, $\rho_0 = 997$ [15] for all examples. The desingularization parameters τ and ε for calculations of the velocity components u and v are set $\tau = \max_j\{|T_j^{n,\mathcal{M}_n}|^2\}$ and $\varepsilon = 10^{-4}$ (see Section 2.1 formula (2.6) in [29]).

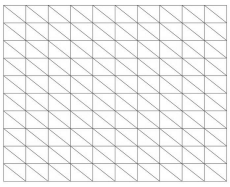


Figure 5.6. An outline of uniform triangular mesh.

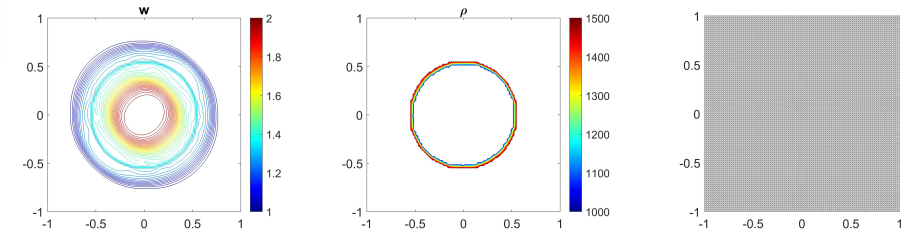
5.1. Example 1. In the first example taken from [9], we will compare the performance of the adaptive central-upwind scheme and the central-upwind scheme without adaptivity on uniform triangular meshes. We also verify experimentally the advantages of the interface reconstruction in compressing the diffusion of variable density at the interface of fluids and improving the accuracy of computed solutions.

We consider the bottom topography $B(x, y, 0) := 0$ and the following initial condition,

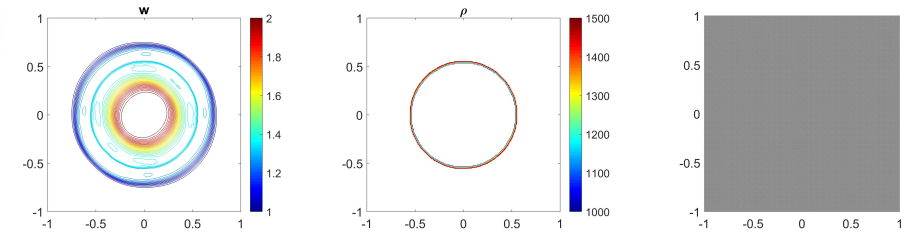
$$(w, u, v, \rho)^T(x, y, 0) = \begin{cases} (2, 0, 0, 1.5\rho_0), & \text{if} \quad x^2 + y^2 < 0.5, \\ (1, 0, 0, \rho_0), & \text{otherwise.} \end{cases} \tag{47}$$

The data is simulated in the domain $[-1, 1] \times [-1, 1]$. The error tolerance (46) for the mesh refinement in this example is set to $\omega = 0.01 \max_j(e_j)$.

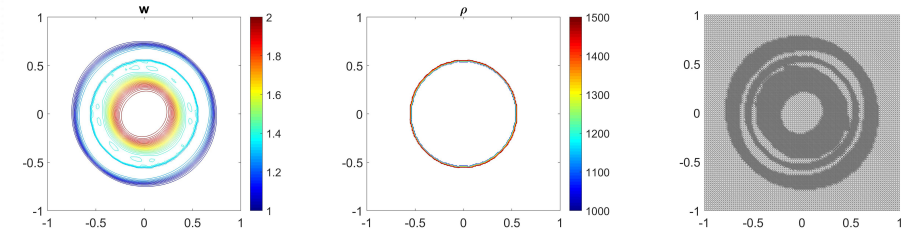
In Fig. 5.7 and Fig. 5.8, we show the numerical solution of the water surface (first column) and the density (second column) at time $t = 0.15$. The solution is calculated by using the central-upwind scheme on uniform meshes on Fig. 5.7 (a, b) and Fig. 5.8 (a,b) and using the adaptive central-upwind scheme on Fig. 5.7 (c, d) and Fig. 5.8 (c,d). The adaptive meshes in Fig. 5.7 (third column) are obtained from the uniform mesh $2 \times 100 \times 100$, Fig. 5.7 (a). The adaptive mesh with one level of refinement $\mathcal{M} = 1$ (as the highest level of refinement) is on Fig. 5.7 (c), and with two levels of refinement $\mathcal{M} = 2$ (as the highest level of refinement) is on Fig. 5.7 (d). As can be observed in Fig. 5.7, both w and ρ are much sharper resolved by using the adaptive central-upwind scheme. We note also, that by increasing the level of refinement from $\mathcal{M} = 1$ to $\mathcal{M} = 2$, the number of cells in the mesh increases from $40,292$ cells with $\mathcal{M} = 1$ to $60,326$ cells with $\mathcal{M} = 2$, but the accuracy is clearly improved with higher resolution as seen in Fig. 5.7 (c) and Fig. 5.7 (d). Also from the adaptive meshes in Fig. 5.7 (third column), one can easily see that only cells in the region having steep gradients are refined. This means that the WLR error estimator accurately detects regions in the domain for adaptive refinement/coarsening.



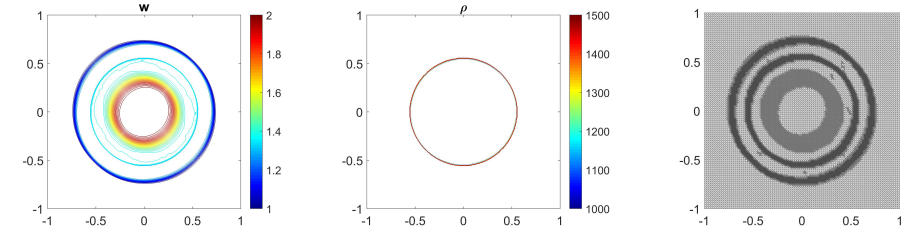
(a) Uniform mesh $2 \times 100 \times 100$.



(b) Uniform mesh $2 \times 200 \times 200$.



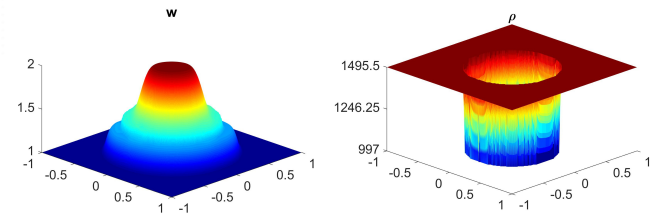
(c) Adaptive mesh with $\mathcal{M} = 1$.



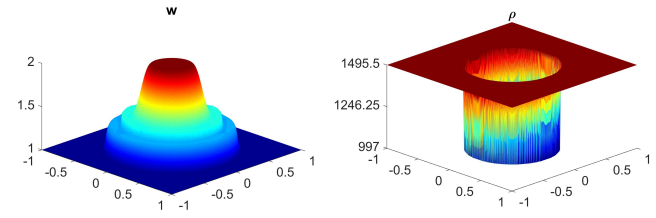
(d) Adaptive mesh with $\mathcal{M} = 2$.

Figure 5.7. Example 1: Contour plots of computational water surface $w(x, y, 0.15)$ (first column) and density $\rho(x, y, 0.15)$ (second column) of the IVP (47) with the corresponding meshes (third column).

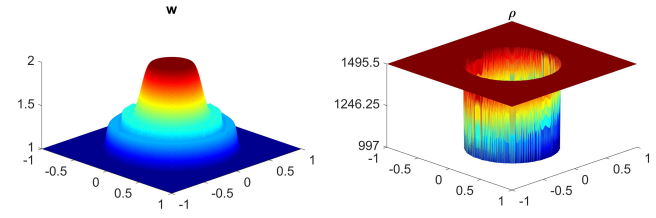
Next, in Table 5.1 we calculate the L^1-errors obtained on the adaptive grids and on the fixed uniform grids. The errors obtained in the uniform meshes are approximate to the errors calculated in the corresponding adaptive meshes (the adaptive meshes have the same size of the smallest cells with the uniform meshes). However, the adaptive scheme uses fewer cells than the central-upwind scheme which does not consider the mesh reconstruction. In addition, in Table 5.2, we also



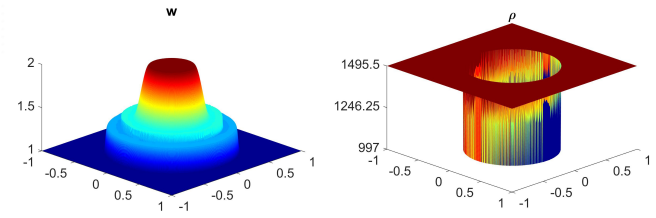
(a) Uniform mesh $2 \times 100 \times 100$.



(b) Uniform mesh $2 \times 200 \times 200$.



(c) Adaptive mesh with $\mathcal{M} = 1$.



(d) Adaptive mesh with $\mathcal{M} = 2$.

Figure 5.8. Example 1: 3-D plots of computational water surface $w(x, y, 0.15)$ (first column) and density $\rho(x, y, 0.15)$ (second column) of the IVP (47).

compute the $\mathcal{R}_{CPU}$ ratio to compare the computational cost of the two methods. The results in Table 5.1 and Table 5.2 show that the adaptive scheme produces similar accuracy as the scheme designed on fixed uniform triangular meshes, but at a less computational cost.

Table 5.1. Example 1: L^1-errors of the water surface w of the IVP (47) at $t = 0.15$, and the convergence rates of the central-upwind scheme without adaptivity (uniform mesh $2 \times N \times N, N = 100, 200, 400$) and the adaptive scheme (the corresponding adaptive mesh is reconstructed from the uniform mesh $2 \times N/2^{\mathcal{M}} \times N/2^{\mathcal{M}}$).

algorithm without adaptivity			adaptive algorithm					
			one level $\mathcal{M} = 1$			two levels $\mathcal{M} = 2$		
cells	L^1-error	rate	cells	L^1-error	rate	cells	L^1-error	rate
$2 \times 100 \times 100$	0.0256		13,516	0.0257		12,800	0.0265	
$2 \times 200 \times 200$	0.0127	1.01	40,292	0.0128	1.00	38,284	0.0133	0.99
$2 \times 400 \times 400$	0.0047	1.43	153,152	0.0049	1.39	60,326	0.0050	1.41

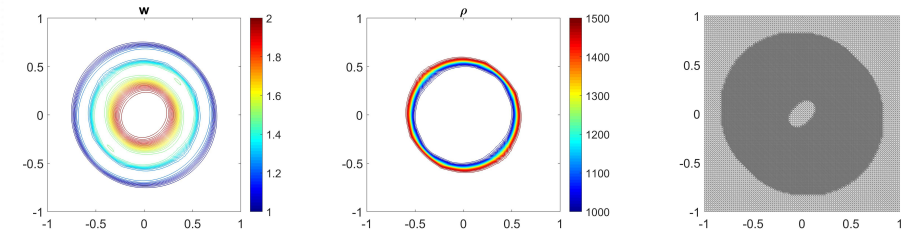
Table 5.2. Example 1: $\mathcal{R}_{CPU}$ ratio for the IVP (47) at $t = 0.15$, where for adaptive central-upwind scheme, we consider the total CPU times and CPU times without the grid generation.

uniform mesh (cells)	adaptive mesh $\mathcal{M} = 1$ (cells)	$\mathcal{R}_{CPU}$ with $\mathcal{M} = 1$		adaptive mesh $\mathcal{M} = 2$ (cells)	$\mathcal{R}_{CPU}$ with $\mathcal{M} = 2$	
		total	without grid generation		total	without grid generation
$2 \times 100 \times 100$	13,516	1.81	2.01	12,800	1.83	1.94
$2 \times 200 \times 200$	40,292	1.82	2.14	38,284	2.14	2.30
$2 \times 400 \times 400$	153,152	1.98	2.29	60,326	2.33	2.50
$\mathcal{R}_{CPU}$ average:		1.87	2.15		2.10	2.27

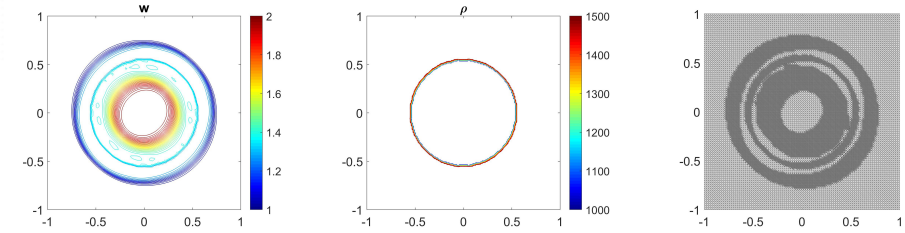
In addition, we will use this example to show that interface reconstruction presented in Section 3.2 plays an important part in preserving the sharpness of the solution as well as improving the accuracy of the adaptive central-upwind scheme. On Fig. 5.9 (a), we plot the water surface w (first) and density ρ (second column) at $t = 0.15$ by using the adaptive central-upwind method, but implemented without the interface tracking technique. We then compare the results shown in Fig. 5.9 (a) to the results calculated by using the adaptive scheme with the interface tracking, Fig. 5.9 (b). The adaptive meshes on Fig. 5.9 (third column) are reconstructed from the uniform mesh $2 \times 100 \times 100$ with one level of refinement. As can be seen in Fig. 5.9 (a), both w and ρ are very scattered around the contact wave when we do not track and reconstruct the interface. Meanwhile, in Fig. 5.9 (b), the proposed adaptive scheme, though using an adaptive mesh with fewer cells (10828 cells fewer), provides more accurate results.

In the next numerical test, we replace the flat bottom with the bottom topography that consists of two Gaussian shaped humps as

$$B(x,y,t) = \begin{cases} 0.5e^{-100(x+0.5)^2+(y+0.5)^2}, & \text{if} \quad x < 0, \\ 0.6e^{-100(x-0.5)^2+(y-0.5)^2}, & \text{if} \quad x \geq 0. \end{cases} \tag{48}$$



(a) Adaptive scheme without interface tracking on adaptive mesh with 51768 cells.



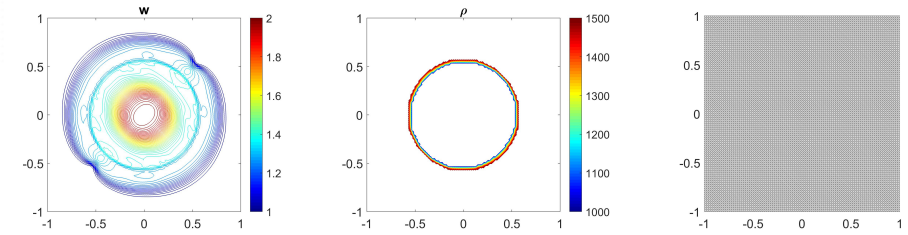
(b) Adaptive scheme with interface tracking on adaptive mesh with 40940 cells

Figure 5.9. Example 1: Computational water surface $w(x, y, 0.15)$ (first column), density $\rho(x, y, 0.15)$ (second column) of the IVP (47), and the corresponding adaptive meshes (third column) obtained by using the proposed adaptive central-upwind scheme (bottom) and using the adaptive scheme without interface tracking (top).

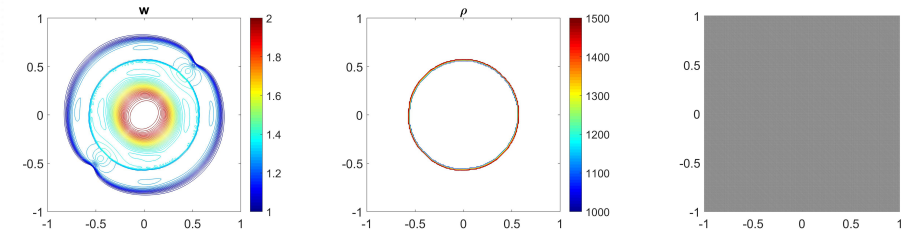
The purpose of this test is to illustrate the performance of the adaptive algorithm in situations having irregular bottom topography. In Fig. 5.10, we show the contour plots of the water surface w (first column) and density ρ (second column) obtained at $t = 0.2$ by using the central-upwind scheme with and without adaptivity. The computed solutions of the water surface exhibit reflecting waves where the flow meets the submerged humps. Clearly, from the plots of the adaptive meshes in Fig. 5.10 (third column), the meshes are adapted to the behavior of the flow. Hence, the WLR error estimator is capable to exactly detect the location of the steep local gradients in the solution.

We then recompute the accuracy of the solution for this example 47-48, see Table 5.3, and the CPU time ratio, see Table 5.4. The results show that the adaptive scheme uses fewer cells and takes a smaller CPU time to achieve the approximately small L^1-errors as computed by the scheme without adaptivity. Therefore, the advantages of the adaptive scheme is maintained in examples with irregular bottom level.

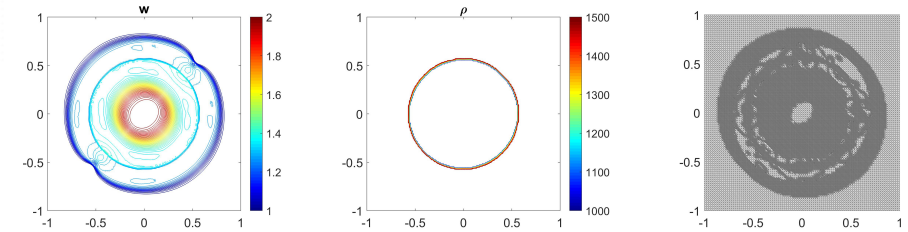
5.2. Example 2. The second numerical example here was proposed in [9] to verify the capability of the adaptive scheme in preserving the steady state solution in "lake at rest' problems, (2) and (3). In particular, the initial data consists of two ąřlake at restąś states of type (2) connected through the density jump corresponding to the ąřlake at restąś state of type (3) as



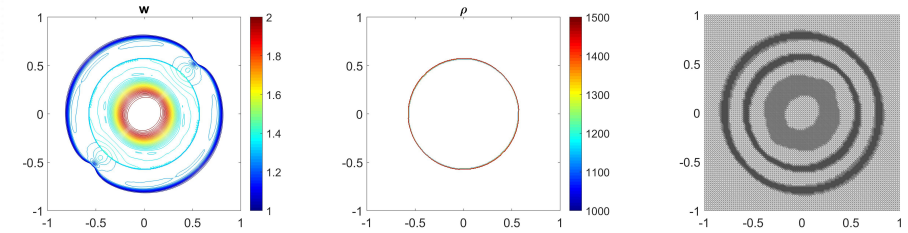
(a) Uniform mesh $2 \times 100 \times 100$.



(b) Uniform mesh $2 \times 200 \times 200$.



(c) Adaptive mesh with $\mathcal{M} = 1$.



(d) Adaptive mesh with $\mathcal{M} = 2$.

Figure 5.10. Example 1: Contour plots of computational water surface $w(x, y, 0.2)$ (first column) and density $\rho(x, y, 0.2)$ (second column) of the IVP (47)-(48) with the corresponding meshes (third column).

$$(w, u, v, \rho)^T(x, y, 0) = \begin{cases} (3, 0, 0, \frac{4}{3}\rho_0), & \text{if} \quad x^2 + y^2 < 0.25, \\ (2, 0, 0, 3\rho_0), & \text{otherwise.} \end{cases} \tag{49}$$

Table 5.3. Example 1: L^1-errors of the water surface w of the IVP (47)-(48) at $t = 0.2$, and the convergence rates of the central-upwind scheme without adaptivity (uniform mesh $2 \times N \times N, N = 100, 200, 400$) and the adaptive scheme (the corresponding adaptive mesh is reconstructed from the uniform mesh $2 \times N/2^{\mathcal{M}} \times N/2^{\mathcal{M}}$).

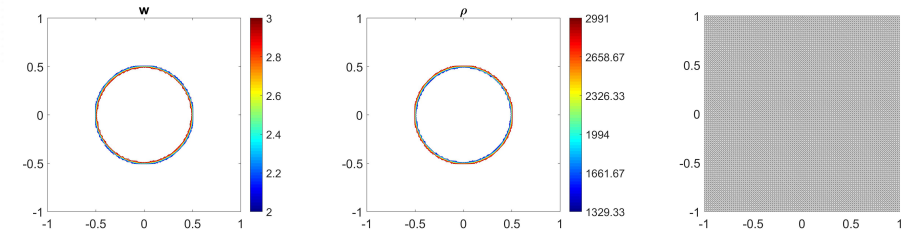
algorithm without adaptivity			adaptive algorithm					
			one level $\mathcal{M} = 1$			two levels $\mathcal{M} = 2$		
cells	L^1-error	rate	cells	L^1-error	rate	cells	L^1-error	rate
$2 \times 100 \times 100$	0.0271		16,252	0.0273		16,006	0.0282	
$2 \times 200 \times 200$	0.0134	1.01	57,300	0.0136	1.00	42,602	0.0140	0.99
$2 \times 400 \times 400$	0.0050	1.43	213,268	0.0054	1.40	120,654	0.0055	1.41

Table 5.4. Example 1: $\mathcal{R}_{CPU}$ ratio for the IVP (47)-(48) at $t = 0.2$, where for adaptive central-upwind scheme, we consider the total CPU times and CPU times without the grid generation.

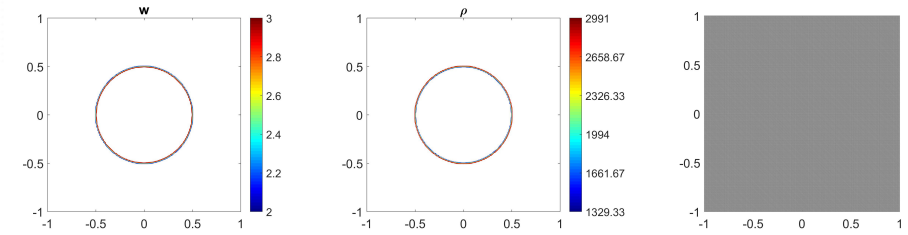
uniform mesh	adaptive mesh $\mathcal{M} = 1$	$\mathcal{R}_{CPU}$ with $\mathcal{M} = 1$		adaptive mesh $\mathcal{M} = 2$	$\mathcal{R}_{CPU}$ with $\mathcal{M} = 2$	
(cells)	(cells)	total	without grid generation	(cells)	total	without grid generation
$2 \times 100 \times 100$	16,252	1.69	1.89	16,006	1.84	1.94
$2 \times 200 \times 200$	57,300	2.57	2.95	42,602	3.41	3.68
$2 \times 400 \times 400$	213,268	2.24	2.60	120,654	2.63	2.84
$\mathcal{R}_{CPU}$ average:		2.17	2.48		2.63	2.82

In this example, we consider the bottom topography (48) on a computational domain $[-1, 1] \times [-1, 1]$. To reconstruct the adaptive meshes, the threshold is set, $\omega = 0.1 \max_j(e_j)$. In Fig. 5.11, we present the plots of the computed water surface (first column) and density (second column) at $t = 0.15$ obtained by using the central-upwind scheme, but without adaptivity, Fig. 5.11 (a, b) and by using the adaptive algorithm Fig. 5.11 (c, d). The adaptive grids plotted on Fig. 5.11 (third column) are generated from the uniform mesh $2 \times 100 \times 100$ with one level of refinement $\mathcal{M} = 1$, Fig. 5.11 (c), and two levels of refinement $\mathcal{M} = 2$, Fig. 5.11 (d). Fig. 5.12 shows the 3D plots of the numerical solution computed by the two methods. As expected, in Fig. 5.11 and Fig. 5.12, the adaptive scheme with interface tracking exactly preserves the steady state. Hence, in Fig. 5.11 (third column), the WLR error only marks cells surrounding the circle of density jump for refinement. In addition, no pressure oscillations are observed at the interface.

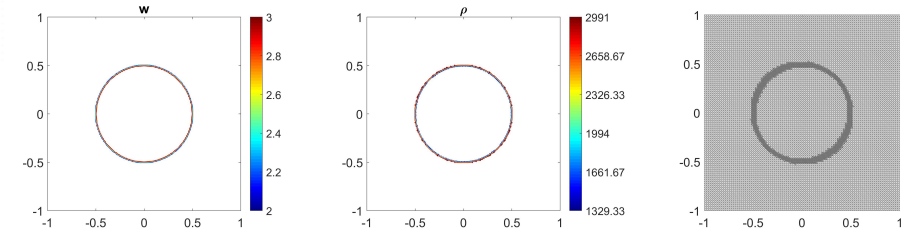
Next, we will illustrate the advantages of the adaptive central-upwind scheme. We compute the L^1-error, Table 5.5, and the CPU ratio, Table 5.6, by using the central-upwind method without adaptivity and using the adaptive algorithm presented in our work. In order to compare the computational costs and calculate $\mathcal{R}_{CPU}$, we consider uniform and adaptive meshes with the same size of the smallest



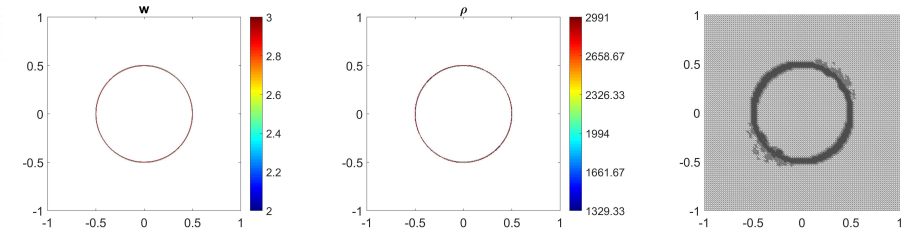
(a) Uniform mesh $2 \times 100 \times 100$.



(b) Uniform mesh $2 \times 200 \times 200$.



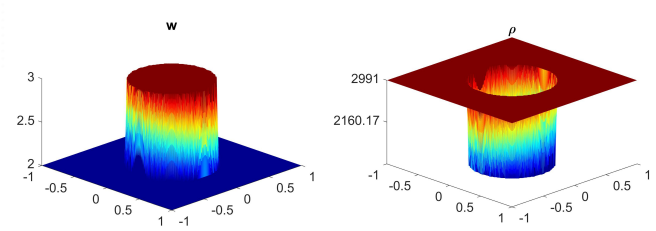
(c) Adaptive mesh with $\mathcal{M} = 1$.



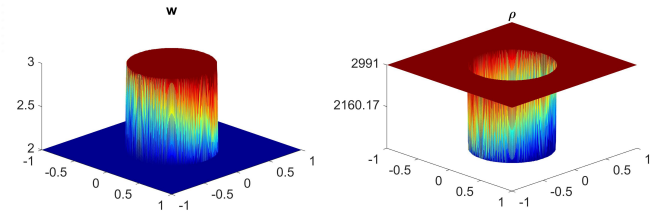
(d) Adaptive mesh with $\mathcal{M} = 2$.

Figure 5.11. Example 2: Contour plots of computational water surface $w(x, y, 0.15)$ (first column) and density $\rho(x, y, 0.15)$ with the corresponding meshes (right column).

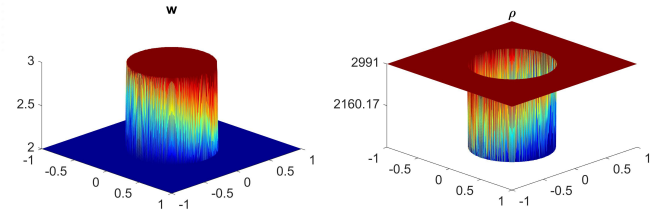
cells. Table 5.5 shows that in the adaptive meshes, we achieve L^1-errors as small as the errors obtained in the corresponding uniform meshes. However, the adaptive algorithm uses fewer cells and reduces the CPU times up to eight times. The computational cost is remarkably cut down since as illustrated in Fig. 5.11 and Fig. 5.12, only a few cells in the neighborhood of the density discontinuity have large WLR errors and are therefore marked for refinement. This example has clearly show the



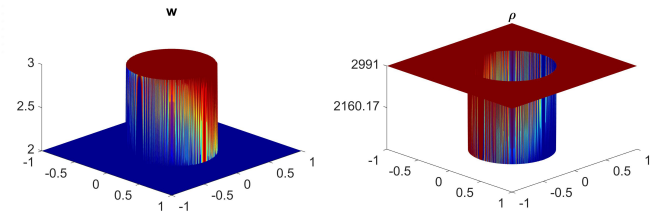
(a) Uniform mesh $2 \times 100 \times 100$.



(b) Uniform mesh $2 \times 200 \times 200$.



(c) Adaptive mesh with $\mathcal{M} = 1$.



(d) Adaptive mesh with $\mathcal{M} = 2$.

Figure 5.12. Example 2: 3-D plots of computational water surface $w(x, y, 0.15)$ (left column) and density $\rho(x, y, 0.15)$ (right column).

efficiency of the proposed scheme for numerically solving the system of multi-fluid flow.

5.3. Example 3. The last example is designed to illustrate the capability of the proposed adaptive algorithm to handle irregular density interfaces. Hence, in a domain $[-1, 1] \times [-1, 1]$, the density jump at $t = 0$ is given by a curve which consists

Table 5.5. Example 2: L^1-errors of the water surface w at $t = 0.15$, and the convergence rates of the central-upwind scheme without adaptivity (uniform mesh $2 \times N \times N, N = 100, 200, 400$) and the adaptive scheme (the corresponding adaptive mesh is reconstructed from the uniform mesh $2 \times N/2^{\mathcal{M}} \times N/2^{\mathcal{M}}$).

algorithm without adaptivity			adaptive algorithm					
			one level $\mathcal{M} = 1$			two levels $\mathcal{M} = 2$		
cells	L^1-error	rate	cells	L^1-error	rate	cells	L^1-error	rate
20,000	0.0045		6,284	0.0045		5,928	0.0045	
80,000	0.0021	1.10	22,546	0.0021	1.10	11,260	0.0021	1.10
320,000	7.9252e-04	1.41	85,132	8.5079e-04	1.30	33,904	8.9818e-04	1.23

Table 5.6. Example 2: $\mathcal{R}_{CPU}$ ratio at $t = 0.15$, where for adaptive central-upwind scheme, we consider the total CPU times and CPU times without the grid generation.

uniform mesh	adaptive mesh $\mathcal{M} = 1$	$\mathcal{R}_{CPU}$ with $\mathcal{M} = 1$		adaptive mesh $\mathcal{M} = 2$	$\mathcal{R}_{CPU}$ with $\mathcal{M} = 2$	
(cells)	(cells)	total	without grid generation	(cells)	total	without grid generation
$2 \times 100 \times 100$	6,284	3.01	3.54	5,928	3.48	3.75
$2 \times 200 \times 200$	22,546	3.78	4.76	11,260	7.76	8.66
$2 \times 400 \times 400$	85,132	4.46	5.57	33,904	11.13	12.86
$\mathcal{R}_{CPU}$ average:		3.75	4.62		7.46	8.42

of a horizontal segment, a vertical segment, and a quarter of a circle connected at their endpoints as illustrated in Fig. 5.13.

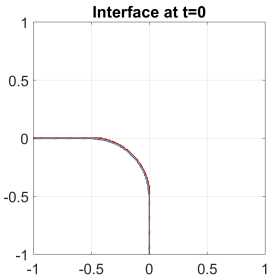


Figure 5.13. Example 3: The interface at initial time $t = 0$.

The initial condition is described by

$$(w,u,v,\rho)^T(x,y,0)=\begin{cases}(2,0,0,\rho_0), & \text{if} \quad (x,y)\in\Omega,\\ (1,0,0,1.5\rho_0), & \text{otherwise.}\end{cases} \tag{50}$$

where

$$\Omega:=\{x<-0.5,y<0\}\cup\{(x+0.5)^2+(y+0.5)^2<0.25\}\cup\{x<0,y<-0.5\}.$$

. We consider a bottom topography with a surbmerged hump as

$$B(x,y,t)=0.5e^{-100(x^2+y^2)}.$$

In this example, we will also perform the same numerical tests which are done in previous examples. Namely, we first calculate the water surface and density at $t=0.15$ using central-upwind scheme without adaptivity and present the results in Fig. 5.14 (a, b) and Fig. 5.15 (a, b). In 5.14 (c, d) and 5.15 (c, d), we plot the results for w (first column) and ρ (second column) obtained by the adaptive scheme. The adaptive grids in 5.14 (third column) are generated from the uniform grid $2\times100\times100$ for one level of refinement $\mathcal{M}=1$ and $\mathcal{M}=2$ using the threshold $\omega=0.01\max_j(e_j)$. As expected, the solutions obtained in the adaptive meshes with high levels of refinement are much sharper than the solutions computed in fixed uniform meshes. The density jump moves Northeast and does not diffuse. There is no non-physical spurious waves generated at the interface. Also, as can be seen in the adaptive meshes, the WLR error indicator captures subtle features of the solution.

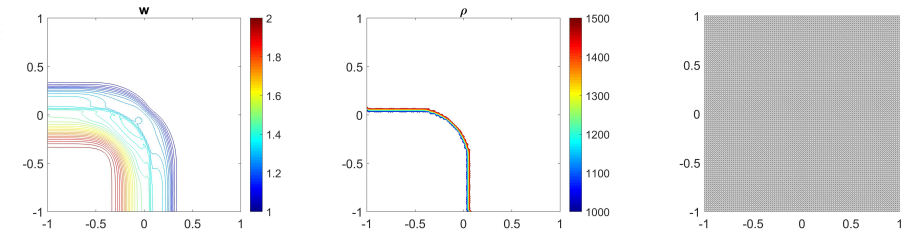
Finally, we compute the L^1-errors in Table 5.7 and the CPU ratio in Table 5.8 by using the adaptive scheme and using the central-upwind method without the mesh generation. By comparing the results presented in Tables 5.7 and 5.8, one can easily see that at a reduced computational cost, the proposed adaptive central-upwind method is still able to obtain the accurate solutions for this example. In our experiments, we considered only $\mathcal{M}=1$ and $\mathcal{M}=2$, but to further enhance the accuracy of the numerical solution at the lower computational cost, one can consider higher levels of refinement.

Table 5.7. Example 3: L^1-errors of the water surface w at $t=0.15$, and the convergence rates of the central-upwind scheme without adaptivity (uniform mesh $2\times N\times N, N=100,200,400$) and the adaptive scheme (the corresponding adaptive mesh is reconstructed from the uniform mesh $2\times N/2^{\mathcal{M}}\times N/2^{\mathcal{M}}$).

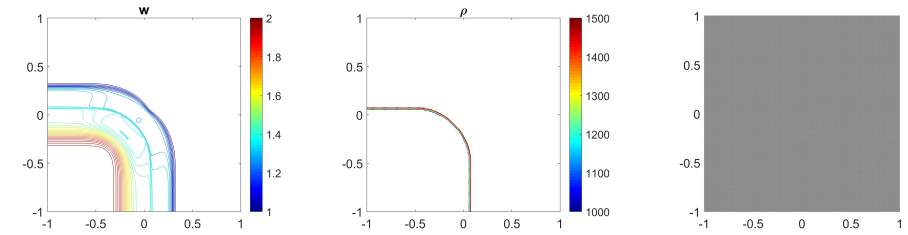
algorithm without adaptivity			adaptive algorithm					
			one level $\mathcal{M}=1$			two levels $\mathcal{M}=2$		
cells	L^1-error	rate	cells	L^1-error	rate	cells	L^1-error	rate
$2\times100\times100$	0.0155		12,164	0.0145		8,994	0.0166	
$2\times200\times200$	0.0074	1.07	40,814	0.0074	0.97	30,263	0.0071	1.23
$2\times400\times400$	0.0027	1.45	148,473	0.0028	1.40	87,214	0.0028	1.34

6. Conclusion

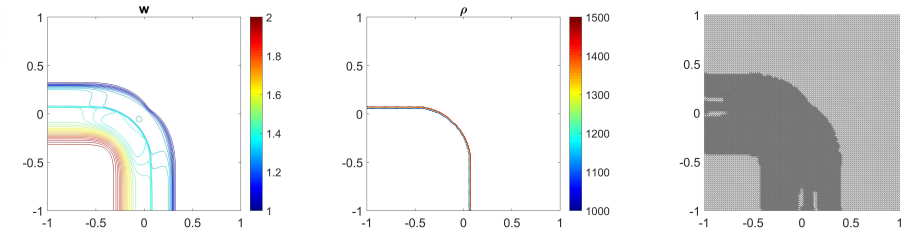
We have developed a new adaptive well-balanced and positivity preserving central-upwind scheme on unstructured traingular meshes for shallow water equations with



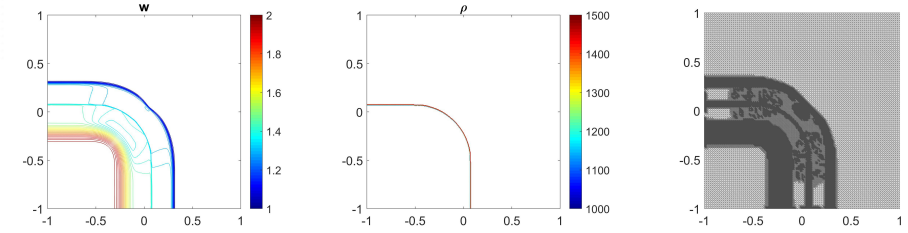
(a) Uniform mesh $2 \times 100 \times 100$.



(b) Uniform mesh $2 \times 200 \times 200$.



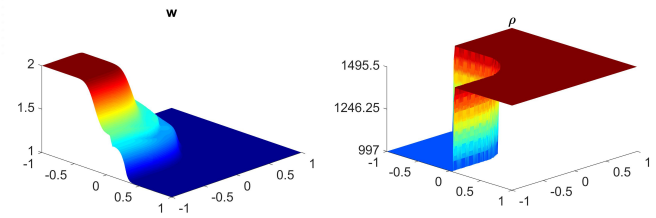
(c) Adaptive mesh with $\mathcal{M} = 1$.



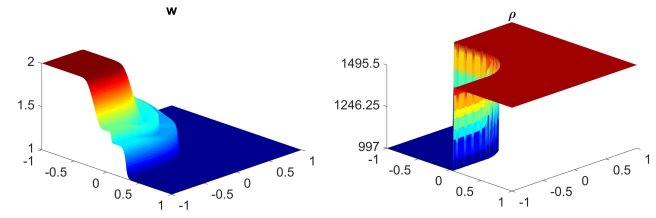
(d) Adaptive mesh with $\mathcal{M} = 2$.

Figure 5.14. Example 3: Contour plots of computational water surface $w(x, y, 0.15)$ (first column) and density $\rho(x, y, 0.15)$ (second column) with the corresponding meshes (third column).

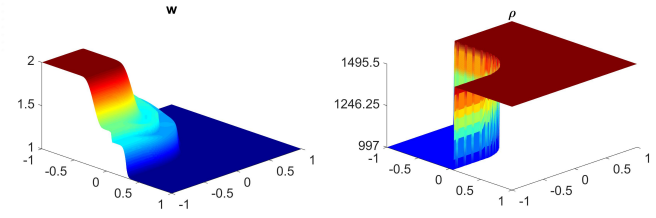
variable density. The scheme is designed as an extension the scheme in [13] by utilizing the interface tracking method in [9] and the interface reconstruction in [15]. The proposed scheme is capable to preserve the steady state solutions (2) and (3) and prevent the oscillation at the density jumps. In addition, to achieve an efficient strategy for the adaptive mesh reconstruction, we also obtain a robust local error indicator. We performed several challenging numerical tests for multi-fluid models and we demonstrated that the new adaptive central-upwind scheme maintains



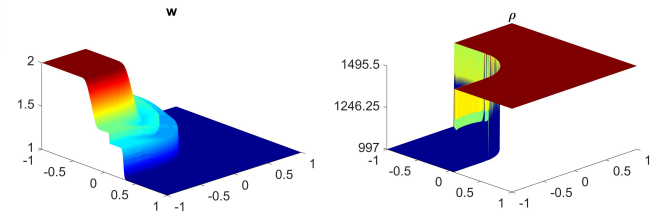
(a) Uniform mesh $2 \times 100 \times 100$.



(b) Uniform mesh $2 \times 200 \times 200$.



(c) Adaptive mesh with $\mathcal{M} = 1$.



(d) Adaptive mesh with $\mathcal{M} = 2$.

Figure 5.15. Example 3: 3-D plots of computational water surface $w(x, y, 0.15)$ (left column) and density $\rho(x, y, 0.15)$ (right column).

well-balanced and positivity-preserving properties and obtains high-accuracy at a reduced computational cost.

Table 5.8. Example 3: $\mathcal{R}_{CPU}$ ratio at $t = 0.15$, where for adaptive central-upwind scheme, we consider the total CPU times and CPU times without the grid generation.

uniform mesh	adaptive mesh $\mathcal{M}=1$	$\mathcal{R}_{CPU}$ with $\mathcal{M}=1$		adaptive mesh $\mathcal{M}=2$	$\mathcal{R}_{CPU}$ with $\mathcal{M}=2$	
(cells)	(cells)	total	without grid generation	(cells)	total	without grid generation
$2 \times 100 \times 100$	12,164	1.75	1.98	8,994	2.69	2.87
$2 \times 200 \times 200$	40,814	2.71	3.14	30,263	3.57	3.86
$2 \times 400 \times 400$	148,473	2.86	3.38	87,214	4.60	5.03
$\mathcal{R}_{CPU}$ average:		2.44	2.83		3.62	3.92

Acknowledgements

I would like to sincerely thank my advisor, Prof. Yekaterina Epshteyn, for her suggestion of the problem to investigate, her encouragement and help. The proposed approach in this paper is an extension of the adaptive method developed in [13] which is my joint work with her. Without her support, this work would not have started, progressed, or ended.

References

[1] R. Abgrall and S. Karni, Computations of compressible multifluids, Journal of computational physics, 169 (2001), pp. 594–623.
[2] D. S. Balsara, M. Dumbser, and R. Abgrall, Multidimensional hllc riemann solver for unstructured meshes–with application to euler and mhd flows, Journal of Computational Physics, 261 (2014), pp. 172–208.
[3] A. Bollermann, G. Chen, A. Kurganov, and S. Noelle, A well-balanced reconstruction of wet/dry fronts for the shallow water equations, J. Sci. Comput., 56 (2013), pp. 267–290.
[4] S. Bryson, Y. Epshteyn, A. Kurganov, and G. Petrova, Central Upwind Scheme on Triangular Grids for the Saint Venant System of Shallow Water Equations, AIP Conference Proceedings, 1389 (2011), pp. 686–689.
[5] S. Bryson, Y. Epshteyn, A. Kurganov, and G. Petrova, Well-balanced positivity preserving central-upwind scheme on triangular grids for the Saint-Venant system, ESAIM Math. Model. Numer. Anal., 45 (2011), pp. 423–446.
[6] S. Bryson and D. Levy, Balanced central schemes for the shallow water equations on unstructured grids, SIAM J. Sci. Comput., 27 (2005), pp. 532–552 (electronic).
[7] A. Chertock, Y. Epshteyn, H. Hu, and A. Kurganov, High-order positivity-preserving hybrid finite-volume-finite-difference methods for chemotaxis systems, Adv. Comput. Math., 44 (2018), pp. 327–350.
[8] A. Chertock, K. Fellner, A. Kurganov, A. Lorz, and P. A. Markowich, Sinking, merging and stationary plumes in a coupled chemotaxis-fluid model: a high-resolution numerical approach, Journal of Fluid Mechanics, 694 (2012), pp. 155–190.
[9] A. Chertock, A. Kurganov, and Y. Liu, Central-upwind schemes for the system of shallow water equations with horizontal temperature gradients, Numer. Math., 127 (2014), pp. 595–639.

[10] P. J. Dellar, Common hamiltonian structure of the shallow water equations with horizontal temperature gradients and magnetic fields, Physics of Fluids, 15 (2003), pp. 292–297.
[11] M. O. Domingues, S. M. Gomes, O. Roussel, and K. Schneider, An adaptive multiresolution scheme with local time stepping for evolutionary pdes, Journal of Computational Physics, 227 (2008), pp. 3758–3780.
[12] R. Donat, M. Martí, A. Martínez-Gavara, and P. Mulet, Well-balanced adaptive mesh refinement for shallow water flows, Journal of Computational Physics, 257 (2014), pp. 937–953.
[13] Y. Epshteyn and T. Nguyen, Adaptive central-upwind scheme on triangular grids for the saint-venant system, arXiv preprint arXiv:2011.06143, (2020).
[14] M. A. Ghazizadeh and A. Mohammadian, An adaptive central-upwind scheme on quadtree grids for variable density shallow water equations, arXiv preprint arXiv:2008.02111, (2020).
[15] M. A. Ghazizadeh, A. Mohammadian, and A. Kurganov, An adaptive well-balanced positivity preserving central-upwind scheme on quadtree grids for shallow water equations, Computers & Fluids, 208 (2020), p. 104633.
[16] S. Gottlieb, C.-W. Shu, and E. Tadmor, Strong stability-preserving high-order time discretization methods, SIAM Rev., 43 (2001), pp. 89–112.
[17] C. W. Hirt and B. D. Nichols, Volume of fluid (vof) method for the dynamics of free boundaries, Journal of computational physics, 39 (1981), pp. 201–225.
[18] X. Y. Hu, B. Khoo, N. A. Adams, and F. Huang, A conservative interface method for compressible flows, Journal of Computational Physics, 219 (2006), pp. 553–578.
[19] S. Karni and A. Kurganov, Local error analysis for approximate solutions of hyperbolic conservation laws, Adv. Comput. Math., 22 (2005), pp. 79–99.
[20] A. Kurganov, Finite-volume schemes for shallow-water equations, Acta Numerica, 27 (2018), pp. 289–351.
[21] A. Kurganov and D. Levy, Central-upwind schemes for the Saint-Venant system, M2AN Math. Model. Numer. Anal., 36 (2002), pp. 397–425.
[22] A. Kurganov and C.-T. Lin, On the reduction of numerical dissipation in central-upwind schemes, Commun. Comput. Phys, 2 (2007), pp. 141–163.
[23] A. Kurganov and Y. Liu, New adaptive artificial viscosity method for hyperbolic systems of conservation laws, J. Comput. Phys., 231 (2012), pp. 8114–8132.
[24] A. Kurganov and J. Miller, Central-upwind scheme for Savage-Hutter type model of submarine landslides and generated tsunami waves, Comput. Methods Appl. Math., 14 (2014), pp. 177–201.
[25] A. Kurganov, S. Noelle, and G. Petrova, Semi-discrete central-upwind schemes for hyperbolic conservation laws and Hamilton-Jacobi equations, SIAM J. Sci. Comput., 23 (2001), pp. 707–740.
[26] A. Kurganov and G. Petrova, Central-upwind schemes on triangular grids for hyperbolic systems of conservation laws, Numer. Methods Partial Differential Equations, 21 (2005), pp. 536–552.
[27] A. Kurganov and G. Petrova, A second-order well-balanced positivity preserving scheme for the Saint-Venant system, Commun. Math. Sci., 5 (2007), pp. 133–160.
[28] A. Kurganov and E. Tadmor, New high-resolution semi-discrete central schemes for Hamilton-Jacobi equations, J. Comput. Phys., 160 (2000), pp. 720–742.
[29] X. Liu, J. Albright, Y. Epshteyn, and A. Kurganov, Well-balanced positivity preserving central-upwind scheme with a novel wet/dry reconstruction on triangular grids for the Saint-Venant system, Journal of Computational Physics, 374 (2018), pp. 213 – 236.
[30] M. Moreira Lopes, M. O. Domingues, K. Schneider, and O. Mendes, Local time-stepping for adaptive multiresolution using natural extension of rungeÍCkutta methods, Journal of Computational Physics, 382 (2019), pp. 291 – 318.

[31] S. Ortleb, A. Meister, and T. Sonar, Adaptive spectral and dtv filtering on triangular grids for the dg method applied to compressible fluid flow, in AIP Conference Proceedings, vol. 1389, American Institute of Physics, 2011, pp. 911–914.
[32] S. Osher and R. P. Fedkiw, Level set methods: an overview and some recent results, Journal of Computational physics, 169 (2001), pp. 463–502.
[33] T. Qin, C.-W. Shu, and Y. Yang, Bound-preserving discontinuous galerkin methods for relativistic hydrodynamics, Journal of Computational Physics, 315 (2016), pp. 323–347.
[34] W. Rider and D. Kothe, Stretching and tearing interface tracking methods, in 12th computational fluid dynamics conference, 1995, p. 1717.
[35] W. J. Rider and D. B. Kothe, Reconstructing volume tracking, Journal of computational physics, 141 (1998), pp. 112–152.
[36] P. Ripa, Conservation laws for primitive equations models with inhomogeneous layers, Geophysical & Astrophysical Fluid Dynamics, 70 (1993), pp. 85–111.
[37] ——, On improving a one-layer ocean model with thermodynamics, Journal of Fluid Mechanics, 303 (1995), pp. 169–201.
[38] M. L. Sæ tra, A. R. Brodtkorb, and K.-A. Lie, Efficient GPU-implementation of adaptive mesh refinement for the shallow-water equations, J. Sci. Comput., 63 (2015), pp. 23–48.
[39] J. A. Sethian and P. Smereka, Level set methods for fluid interfaces, Annual review of fluid mechanics, 35 (2003), pp. 341–372.
[40] H. Shirkhani, A. Mohammadian, O. Seidou, and A. Kurganov, A well-balanced positivity-preserving central-upwind scheme for shallow water equations on unstructured quadrilateral grids, Comput. & Fluids, 126 (2016), pp. 25–40.
[41] M. Sussman, A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles, Journal of Computational Physics, 187 (2003), pp. 110–136.
[42] M. Sussman and E. G. Puckett, A coupled level set and volume-of-fluid method for computing 3d and axisymmetric incompressible two-phase flows, Journal of computational physics, 162 (2000), pp. 301–337.
[43] E. Tadmor, Local error estimates for discontinuous solutions of nonlinear hyperbolic equations, SIAM J. Numer. Anal., 28 (1991), pp. 891–906.
[44] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y.-J. Jan, A front-tracking method for the computations of multiphase flow, Journal of computational physics, 169 (2001), pp. 708–759.
[45] S. O. Unverdi and G. Tryggvason, A front-tracking method for viscous, incompressible, multi-fluid flows, Journal of computational physics, 100 (1992), pp. 25–37.
[46] I. Wenneker, G. Segal, and P. Wesseling, Computation of compressible flows on unstructured staggered grids, in Proceedings European Congress on Computational Methods in Applied Sciences and Engineering, ECCOMAS, vol. 2, 2000.
[47] Y. Xing and X. Zhang, Positivity-preserving well-balanced discontinuous galerkin methods for the shallow water equations on unstructured triangular meshes, Journal of Scientific Computing, 57 (2013), pp. 19–41.
[48] X. Yang and A. J. James, Analytic relations for reconstructing piecewise linear interfaces in triangular and tetrahedral grids, Journal of computational physics, 214 (2006), pp. 41–54.
[49] X. Yang, A. J. James, J. Lowengrub, X. Zheng, and V. Cristini, An adaptive coupled level-set/volume-of-fluid interface capturing method for unstructured triangular grids, Journal of Computational Physics, 217 (2006), pp. 364–394.

Department of Mathematics, The University of Utah, Salt Lake City, UT 84112, USA
E-mail: tnguyen@math.utah.edu