Journal of Scientific Computing (2022) 92:86
https://doi.org/10.1007/s10915-022-01902-y

A High-Order Velocity-Based Discontinuous Galerkin Scheme for the Shallow Water Equations: Local Conservation, Entropy Stability, Well-Balanced Property, and Positivity Preservation

Guosheng Fu[1]

Received: 31 January 2022 / Revised: 3 May 2022 / Accepted: 9 June 2022 /
Published online: 25 July 2022

Abstract
The nonlinear shallow water equations (SWEs) are widely used to model the unsteady water flows in rivers and coastal areas. In this work, we present a novel class of locally conservative, entropy stable and well-balanced discontinuous Galerkin (DG) methods for the nonlinear shallow water equation with a non-flat bottom topography. The major novelty of our work is the use of velocity field as an independent solution unknown in the DG scheme, which is closely related to the entropy variable approach to entropy stable schemes for system of conservation laws proposed by Tadmor (in: Tezduyar, Hughes T (eds) Proceedings of the winter annual meeting of the American Society of Mechanical Engineering 1986) back in 1986, where recall that velocity is part of the entropy variable for the shallow water equations. Due to the use of velocity as an independent solution unknown, no specific numerical quadrature rules are needed to achieve entropy stability of our scheme on general unstructured meshes in two dimensions. The proposed DG semi-discretization is then carefully combined with the classical explicit strong stability preserving Runge–Kutta (SSP–RK) time integrators (Gottlieb et al. in SIAM Rev. 43, 89–112, 2001) to yield a locally conservative, well-balanced, and positivity preserving fully discrete scheme. Here the positivity preservation property is enforced with the help of a simple scaling limiter. In the fully discrete scheme, we reintroduce discharge as an auxiliary unknown variable. In doing so, standard slope limiting procedures can be applied on the conservative variables (water height and discharge) without violating the local conservation property. Here we apply a characteristic-wise TVB limiter (Cockburn and Shu in J Comput Phys 141:199–224, 1998) on the conservative variables using the Fu-Shu troubled cell indicator (Fu and Shu in J Comput Phys 347:305–327, 2017) in each inner stage of the Runge–Kutta time stepping to suppress numerical oscillations. This fully discrete can be readily applied to various SWEs simulations without dry areas where the water height is close to *zero*. The case with dry areas need further special attention, where the velocity approximation can be unphysically large near cells with a small water height, which may eventually crashes the simulation if no special treatment is used near these cells.

✉ Guosheng Fu
gfu@nd.edu

[1] Department of Applied and Computational Mathematics and Statistics, University of Notre Dame, Notre Dame, USA

Here we propose a simple wetting/drying treatment for the velocity update without violating the local conservation property to enhance the robustness of the overall scheme. One- and two-dimensional numerical experiments are presented to demonstrate the performance of the proposed methods.

Keywords Discontinuous Galerkin methods · Shallow water equations · Entropy stable · Entropy variable · Well-balanced property · Positivity-preserving limiter

Mathematics Subject Classification 65N30 · 65N12 · 76S05 · 76D07

1 Introduction

The system of nonlinear shallow water equations (SWEs) is a mathematical model for the fluid movement in various shallow water environments, where the horizontal scales of motion are much greater than the vertical scale. The SWEs have been widely used to model flow in the river, near-shore ocean, and Earth's atmosphere, etc. In two dimensions, the inviscid SWEs take the following form:

$$h_t + \nabla \cdot (h\boldsymbol{u}) = 0, \tag{1a}$$

$$(h\boldsymbol{u})_t + \nabla \cdot (h\boldsymbol{u} \otimes \boldsymbol{u}) + \frac{1}{2} g \nabla (h^2) = -gh\nabla b, \tag{1b}$$

where h is the water height, $\boldsymbol{u} = (u, v)$ is the velocity field, $b(x, y)$ represents the bottom topography and g is the gravitational constant.

Below we review the four important properties that the SWEs satisfy, namely the entropy condition, the *lake-at-rest* well-balanced property, the positivity of the water height h, and the conservation property. The system (1) is a system of *balance laws*,

$$U_t + F(U)_x + G(U)_y = -s(x, y, U), \tag{2}$$

where $U = [h, hu, hv]^T$ is the vector of unknowns, $F = [hu, hu^2 + \frac{1}{2}gh^2, huv]^T$ and $G = [hv, huv, hv^2 + \frac{1}{2}gh^2]^T$ are flux vectors, and $s = [0, ghb_x, ghb_y]^T$ is the source vector. It is well-known that solutions of the balance law (2) can develop shock discontinuities in a finite time, independent of whether the initial data is smooth or not. Hence, the solution of (1) are considered in the weak sense [17], which are in general not unique.

(i) The entropy condition. To address the issue of non-uniqueness for weak solutions, an additional admissibility criterion based on the so-called *entropy condition* is imposed. For the SWEs, the total energy

$$E(U) := \frac{1}{2} h(u^2 + v^2) + \frac{1}{2} gh^2 + ghb$$

serves as an entropy function, which contains the kinetic energy $\frac{1}{2}h(u^2 + v^2)$ and the gravitational potential energy $\frac{1}{2}gh^2 + ghb$. If the bottom topography b and the solution of (1) is smooth, a straightforward calculation [21] reveals that

$$E(U)_t + \left(\frac{1}{2}(hu^3 + huv^2) + ghu(h + b)\right)_x + \left(\frac{1}{2}(hu^2 v + hv^3) + ghv(h + b)\right)_y = 0, \tag{3}$$

which is obtained by taking the inner product of the SWEs (1) with the *entropy variables*

$$V := \partial_U E = [g(h+b) - \frac{1}{2}(u^2+v^2), u, v]^T \tag{4}$$

and applying the chain rule. Since entropy should be dissipated across shock discontinuities, the entropy conservation equation (3) needs to be replaced by the following entropy dissipation postulate

$$E(U)_t + \left(\frac{1}{2}(hu^3 + huv^2) + ghu(h+b)\right)_x + \left(\frac{1}{2}(hu^2v + hv^3) + ghv(h+b)\right)_y \leq 0, \tag{5}$$

(ii) Steady states and well-balanced property. Another important issue which arises in SWEs (1) is the simulation of their steady states, which are solutions that are independent of the time variables. The most import example of a steady state solution to (1) is the so-called *lake at rest*, given by

$$u = v \equiv 0, \quad h + b \equiv Const. \tag{6}$$

Many interesting applications, such as waves on a lake or tsunami waves in deep ocean, involve computing perturbations of the lake at rest. A numerical scheme which preserves a discrete version of the steady state (6) is termed *well-balanced* with respect to the steady state.

(iii) Positivity of the water height. The water height h in the SWEs needs to remain positive (non-negative) for the system (1) to remain well-posed. If the water height becomes negative, the system (1) will be non-hyperbolic and non-physical, and the problem will be ill-posed.

(iv) The conservation property. For constant topography $b \equiv Const$, the balance law (2) reduces to a hyperbolic system of conservation laws

$$U_t + F(U)_x + G(U)_y = 0.$$

Integrating the above equation over any control volume $D \subset \mathbb{R}^2$ and applying the Gauss law, there holds the following conservation property:

$$\frac{d}{dt}\int_D U \, \mathrm{dx} = -\int_{\partial D} [F(U), G(U)]^T \cdot \boldsymbol{n} \, \mathrm{ds},$$

where $\boldsymbol{n}$ is the outward unit normal direction on the boundary ∂D.

The main focus of this work is to construct high-order numerical schemes for the SWEs (1) on unstructured meshes that respect the above mentioned four properties. All these properties are important in practice:

- The Lax-Wendroff theorem [29] ensures that if a conservative scheme produces a discrete solution $U_h(x,t)$ that uniformly converges to $\bar{U}(x,t)$, then $\bar{U}(x,t)$ is a weak solution to the continuous equation. Non-conservative schemes may converge to wrong solutions.
- Many shallow water applications involve rapidly moving interfaces between wet and dry areas, where no water is present. If no special attention is paid to maintain the positivity (non-negativity) of the water height, standard numerical methods may produce unacceptable negative water height near the dry/wet front, which crashes the numerical simulation.
- Well-balanced schemes are essential for computing perturbations of steady states.

- Entropy stability (5) provides additional stabilization mechanism to the scheme which further enhance its robustness.

Various numerical schemes satisfying (part of) these properties for hyperbolic conservation laws or balance laws have been proposed in the literature. We refer to the review articles [48, 50] for a survey of numerical schemes for the SWEs, in particular high-order well-balanced and positivity-preserving schemes; to the review articles [53, 54] for a survey of maximum-principle-satisfying and positivity-preserving high-order schemes for conservation laws; and to the review article [40] for entropy stable schemes. In particular, the discontinuos Galerkin (DG) method [13] is one of the most popular high-order methods for SWEs and related models; see, e.g., [5, 7, 8, 18, 25, 35, 42, 49].

Of particular relevance to the current work is the class of entropy stable schemes for the SWEs, which respect the entropy dissipation postulate (5). First-order entropy stable finite volume (FV) schemes for the SWEs were proposed in [20, 21, 41] where the key concepts of entropy variable, entropy conservative/stable numerical fluxes were discussed. Similar entropy conservative/stable numerical fluxes were adopted in the high-order *nodal* DG literature, which, in combination with discrete derivative operators using Gauss-Lobatto quadrature points that satisfy the summation-by-parts (SBP) property, yield entropy conservative/stable DG discretizations [24, 45, 46]. High-order entropy stable DG schemes were more recently extended to *modal* formulations [44, 47], following the work of Chan [10]. All these works are also well-balanced and conservative. These entropy stable spatial discretizations were then combined with *explicit* strong stability preserving Runge-Kutta (SSP-RK) time integrators [26] to yield fully discrete conservative and well-balanced schemes. We note that, in all these works, the entropy stability was proven in the semi-discrete level where only spatial discretization was involved, which does not hold theoretically for the fully explicit discretizations. We further note that the schemes [44, 46] can preserve the positivity of the water height with the aid of a positivity preserving scaling limiter [54]. Moreover, most of the above cited works use structured/rectangular meshes, with the exceptions of [46] which uses unstructured quadrilateral meshes and [44] which works on unstructured triangular meshes.

In this work, we construct high-order locally conservative, positivity preserving, well-balanced, and entropy stable DG schemes for the SWEs on general triangular meshes. Our approach is very different from the above cited entropy stable DG schemes which relies on the SBP property of the underling difference operators. In our semi-discrete scheme, instead of directly approximating the conservative variables, we use the water height and velocity as the solution unknowns. As a result, entropy stability is achieved naturally within the weak formulation without the need to convert to the strong form or work with difference operators/matrices. Our approach is more closely related to the entropy variable approach to entropy stable schemes proposed by Tadmor [38] back in 1986, as the velocity is part of the entropy variable (4). For the purpose of efficient explicit time integration, we still keep the water height as the solution unknown. As a result, we need to use the skew-symmetric formulation of the momentum equation [24] to guarantee entropy stability of the semi-discrete scheme. Similar to the works [44, 47], our proposed scheme achieves entropy stability regardless of the choice of underlying numerical integration rules, although the proofs are very different. Actually, all integrals in our scheme involve polynomials only, which can be easily computed exactly if one wishes.

The proposed DG semi-discretization is then carefully combined with a classical SSP-RK time integrator [26], in combination with a positivity-preserving scaling limiter to ensure positivity of the water height. Here special attention is paid to the Runge-Kutta inner stage reconstructions to maintain local conservation of the fully discrete scheme. To do so, we re-

introduce the discharge (momentum) as an auxiliary solution unknown and reconstruct inner stage values based on the the conservative variables, i.e., water height and discharge. We prove that this water height-velocity-discharge three-field formulation is mathematically equivalent to the water height-velocity two-field formulation in the semi-discrete level. The advantage of this three-field formulation over the two-field formulation is that standard slope limiting procedures can now be applied on the conservative variables (water height and discharge) to suppress numerical oscillations near discontinuities without violating the local conservation property. Here we apply the characteristic-wise TVB limiter [15] with the Fu-Shu troubled cell indicator [22] using the total height as the indicating function.

The last ingredient of our fully discrete scheme is a proper wetting/drying treatment for problems with (moving) dry areas. The above mentioned positivity preserving limiter and TVB limiter do not directly work on the velocity approximation. As a result, the scheme may produce arbitrarily large velocity approximations near dry cells where the water height is very small. Without any special treatment near these regions, the large velocity near dry areas will dictate the time step size, and may even crash the code due to too large velocity values. We didn't find existing wetting/drying treatments in the literature that work for our scheme. Hence we introduce a new wetting/drying treatment that at least works for our numerical examples; see details in Remark 3.5 below.

The rest of the paper is organized as follows. In Sect. 2, we introduce the reformation of SWEs (1), and used it to design a conservative, well-balanced and entropy stable DG spatial discretization. In Sect. 3, we present the explicit temporal discretization, and then prove the posivitity preservation property. The implementation of a characteristic-wise TVD slope limiter with an efficient troubled cell indicator is then discussed. We further remark on the proper wetting/drying treatment in the velocity calculation. Numerical results in one- and two-dimensions are then reported in Sect. 4. We draw concluding remarks in Sect. 5.

2 Reformulation of SWEs and the DG Semi-Discretization

In this section, we first reformulate the SWEs (1) into an equivalent skew-symmetric form, c.f. [24], and then introduce the associated conservative, entropy stable and well-balanced DG semi-discretization. Although using the same skew-symmetric form, we emphasis that our entropy stable DG discretization is completely different from the work [24], where we use velocity as independent solution unknowns.

2.1 The Skew-Symmetric form of the SWEs

Multiplying the mass conservation equation (1a) by $\frac{1}{2}\boldsymbol{u}$, and subtract it from the momentum balance equation (1b), we get the following equivalent form of the SWEs:

$$h_t + \nabla \cdot (h\boldsymbol{u}) = 0, \tag{7a}$$

$$(h\boldsymbol{u})_t + \nabla \cdot (h\boldsymbol{u} \otimes \boldsymbol{u}) + gh\nabla(h+b) - \frac{1}{2}h_t\boldsymbol{u} - \frac{1}{2}\nabla \cdot (h\boldsymbol{u})\boldsymbol{u} = 0. \tag{7b}$$

Here (7b) is referred to as the skew-symmetric form of the momentum balance equation (1b), c.f. [24]. Multiplying (7a) with $g(h+b)$ and (7b) with $\boldsymbol{u}$ and adding, we immediate get the entropy conservation equality (3). This suggests to use finite elements to directly approximate the quantities $g(h+b)$ and $\boldsymbol{u}$ in order to design a Galerkin method that respect the entropy conservation property (3), which is the approach we take in this article. In practice, we use a

discontinuous finite element space to directly approximate the water height $h(x,t)$ and use the same finite element space to approximate the bottom topography $b(x)$, so that $g(h+b)$ can be taken as a test function in the Galerkin formulation.

2.2 The Conservative, Entropy-Stable, and Well-Balanced DG Spatial Discretization

Without loss of generality, we formulate the DG spatial discretization for the SWEs (7) on a periodic domain $\Omega \subset \mathbb{R}^2$. Other standard boundary conditions will be used in the numerical experiments. Here we formulate the scheme on a general unstructured triangular mesh, while noting that the proposed method also works on structured/unstructured quadrilateral meshes.

To this end, let $\Omega_h := \{K\}$ be a conforming triangular discretization of the domain Ω. Denote $\partial\Omega_h := \{\partial K\}$ as the collection of element boundaries with $\boldsymbol{n}_K$ the associated outward unit normal direction. Let $\mathcal{E}_h := \{F\}$ be the collection of edges of the triangulation Ω_h. For any polynomial degree $k \geq 0$, let

$$V_h^k := \{v \in L^2(\Omega) : \quad v|_K \in \mathcal{P}_k(K), \quad \forall K \in \Omega_h\}, \tag{8}$$

where $\mathcal{P}_k(K)$ is the space of polynomials of degree at most k on the element K. Furthermore, let $\boldsymbol{V}_h^k$ be the vectorial version of the space V_h^k. Given an edge $F = K^+ \cap K^- \in \mathcal{E}_h$ which is shared by two elements K^+ and K^-, we denote $\boldsymbol{n}$ as the unit normal direction on F pointing towards K^-, and denote $[\![\phi]\!]|_F := \phi^+ - \phi^-$ and $\{\phi\}|_F := \frac{1}{2}(\phi^+ + \phi^-)$ as the standard jump and average on F for any function $\phi \in V_h^k$, where $\phi^\pm := \phi|_{K^\pm}$.

The proposed DG spatial discretization of (7) on the periodic domain Ω reads as follows: find $(h_h, \boldsymbol{u}_h) \in V_h^k \times \boldsymbol{V}_h^k$ such that

$$M_h\left((h_h)_t, e_h\right) + A_h((h_h, \boldsymbol{u}_h), e_h) \; = 0, \tag{9a}$$

$$\begin{aligned} &\boldsymbol{M}_h\left((h_h\boldsymbol{u}_h)_t, \boldsymbol{v}_h\right) + B_h((h_h, \boldsymbol{u}_h), \boldsymbol{v}_h) + C_h((h_h, \boldsymbol{u}_h), \boldsymbol{v}_h) \\ &\quad - M_h\left((h_h)_t, \frac{1}{2}\boldsymbol{u}_h \cdot \boldsymbol{v}_h\right) - A_h\left((h_h, \boldsymbol{u}_h), \frac{1}{2}\boldsymbol{u}_h \cdot \boldsymbol{v}_h\right) \; = 0, \end{aligned} \tag{9b}$$

for all $(e_h, \boldsymbol{v}_h) \in V_h^k \times \boldsymbol{V}_h^k$, where $b_h \in V_h^k$ is a proper approximation of the bottom topography, and the associated operators are given below:

$$M_h((h_h)_t, e_h) := \sum_{K\in\Omega_h} \int_K (h_h)_t e_h \,\mathrm{dx}, \tag{9c}$$

$$\begin{aligned} A_h((h_h, \boldsymbol{u}_h), e_h) &:= \sum_{K\in\Omega_h} \left(-\int_K h_h\boldsymbol{u}_h \cdot \nabla e_h \,\mathrm{dx} + \int_{\partial K} \widehat{h_h\boldsymbol{u}_h} \cdot \boldsymbol{n}_K e_h \,\mathrm{ds} \right), \\ &= -\sum_{K\in\Omega_h} \int_K h_h\boldsymbol{u}_h \cdot \nabla e_h \,\mathrm{dx} + \sum_{F\in\mathcal{E}_h} \int_F \widehat{h_h\boldsymbol{u}_h} \cdot \boldsymbol{n} [\![e_h]\!] \,\mathrm{ds}, \end{aligned} \tag{9d}$$

$$\boldsymbol{M}_h\Big((h_h\boldsymbol{u}_h)_t, \boldsymbol{v}_h\Big) := \sum_{K\in\Omega_h} \int_K (h_h\boldsymbol{u}_h)_t \cdot \boldsymbol{v}_h \,\mathrm{dx}, \tag{9e}$$

$$B_h((h_h, \boldsymbol{u}_h), \boldsymbol{v}_h) := -\sum_{K\in\Omega_h} \int_K h_h(\boldsymbol{u}_h \otimes \boldsymbol{u}_h) : \nabla \boldsymbol{v}_h \,\mathrm{dx} + \sum_{F\in\mathcal{E}_h} \int_F (\widehat{h_h\boldsymbol{u}_h\boldsymbol{u}_h})\boldsymbol{n} \cdot [\![\boldsymbol{v}_h]\!] \,\mathrm{ds}, \tag{9f}$$

$$C_h((h_h, \boldsymbol{u}_h), \boldsymbol{v}_h) := \sum_{K\in\Omega_h}\int_K gh_h\nabla(h_h+b_h)\cdot\boldsymbol{v}_h\,\mathrm{dx} - \sum_{F\in\mathcal{E}_h}\int_F g[\![h_h+b_h]\!]\{h_h\boldsymbol{v}_h\}\cdot\boldsymbol{n}\,\mathrm{dx}, \tag{9g}$$

where the numerical fluxes $\widehat{h_h\boldsymbol{u}_h}\cdot\boldsymbol{n}$ and $(\widehat{h_h\boldsymbol{u}_h\boldsymbol{u}_h})\boldsymbol{n}$ in the operators (9d) and (9f) are defined as follows:

$$\widehat{h_h\boldsymbol{u}_h}\cdot\boldsymbol{n} := \{h_h\boldsymbol{u}_h\}\cdot\boldsymbol{n} + \frac{1}{2}\alpha_h[\![h_h+b_h]\!], \tag{9h}$$

$$(\widehat{h_h\boldsymbol{u}_h\boldsymbol{u}_h})\boldsymbol{n} := \{h_h\boldsymbol{u}_h\}\cdot\boldsymbol{n}\,\{\boldsymbol{u}_h\} + \frac{1}{2}\alpha_h[\![(h_h+b_h)\boldsymbol{u}_h]\!], \tag{9i}$$

with estimated maximum speed

$$\alpha_h|_F := \max\left\{\sqrt{gh_h^+} + |\boldsymbol{u}_h^+\cdot\boldsymbol{n}|, \sqrt{gh_h^-} + |\boldsymbol{u}_h^-\cdot\boldsymbol{n}|\right\}. \tag{9j}$$

We will show below that these local Lax-Friedrichs type numerical fluxes are entropy-stable. We note that the above operators are very natural DG scretizations of the corresponding PDE operators in (7), in particular,

- The operators M_h in (9c) and $\boldsymbol{M}_h$ in (9e) are the weak forms associated with the time derivative term h_t in (7a), and $(h_h\boldsymbol{u}_h)_t$ in (7b), respectively;
- The operator A_h in (9d) is the DG discretization of the convection term $\nabla\cdot(h\boldsymbol{u})$ in (7a), with the numerical flux (9h), and The operator B_h in (9f) is the DG discretization of the convection term $\nabla\cdot(h\boldsymbol{u}\otimes\boldsymbol{u})$ in (7b), with the numerical flux (9i). Here the particular choice of the numerical fluxes (9h) and (9i) is crucial for the *entropy stability* of the semi-discrete scheme (9). Similar numerical fluxes have been used in the literature, c.f. [21, 24];
- The operators M_h and A_h in (9b) are the operators associated with the skew-symmetric terms $\frac{1}{2}h_t\boldsymbol{u}$ and $\frac{1}{2}\nabla\cdot(h\boldsymbol{u})\boldsymbol{u}$ in (7b), respectively;
- The operator C_h in (9g) is a DG discretization of the gravitational term $gh\nabla(h+b)$ in (7b) using a central numerical flux. To see this, we note that the DG discretization with central numerical flux for this operator reads as follows:

$$-\sum_{K\in\Omega_h}\int_K g(h_h+b_h)\nabla\cdot(h_h\boldsymbol{v}_h)\,\mathrm{dx} + \sum_{F\in\mathcal{E}_h}\int_F g\{h_h+b_h\}[\![h_h\boldsymbol{v}_h]\!]\cdot\boldsymbol{n}\,\mathrm{dx},$$

  which is equivalent to C_h by integration by parts. We mention that the two gravitational terms in momentum balance (1b) are combined into a single non-conservative product, which is another key to the entropy stability and well-balanced property of our scheme. We note that such non-conservative product has been explored in the literature, e.g., [33], to design well-balanced DG schemes.

We conclude this section with the main properties of our proposed DG discretization (9), namely, local conservation, entropy stability and the well-balanced property.

Theorem 2.1 *The semi-discrete scheme (9) is*

- *locally conservative in the sense of the following equalities:*

$$\frac{d}{dt}\int_K h_h\,\mathrm{dx} = -\int_{\partial K}\widehat{h_h\boldsymbol{u}_h}\cdot\boldsymbol{n}_K\,\mathrm{ds}, \tag{10a}$$

$$\frac{d}{dt}\int_K h_h\boldsymbol{u}_h\,\mathrm{dx} = -\int_{\partial K}\left(\widehat{h_h\boldsymbol{u}_h\boldsymbol{u}_h} + \frac{1}{2}gh_h^+h_h^-\right)\boldsymbol{n}_K\,\mathrm{ds}$$

$$-\int_K g h_h \nabla(b_h)\,\mathrm{dx} + \int_{\partial K} \frac{1}{2} g h_h [\![b_h]\!] \boldsymbol{n}_K\,\mathrm{ds} \tag{10b}$$

- *entropy stable in the sense of the following equality:*

$$\frac{d}{dt} E_h = -\sum_{F\in\mathcal{E}_h} \int_F \frac{1}{2}\alpha_h \left(g[\![h_h + b_h]\!]^2 + \{h_h + b_h\}[\![\boldsymbol{u}_h]\!] \cdot [\![\boldsymbol{u}_h]\!] \right) \mathrm{ds} \le 0, \tag{10c}$$

where the discrete entropy (total energy) is

$$E_h := \sum_{K\in\Omega_h} \int_K (\frac{1}{2}\boldsymbol{h}_h |\boldsymbol{u}_h|^2 + \frac{1}{2} g h_h^2 + g h_h b_h)\mathrm{dx},$$

- *well-balanced in the sense that it preserve the lake-at-rest steady state: if the initial condition satisfies*

$$\boldsymbol{u}_h(0) = 0, \textit{ and } \quad h_h(0) + b_h = C, \tag{10d}$$

where the constant $C > \max b_h$*, then the solution to the semi-discrete scheme (9) satisfies*

$$\boldsymbol{u}_h(t) = 0, \textit{ and } \quad h_h(t) + b_h = C. \tag{10e}$$

Proof Taking test function $e_h := 1$ on element K and *zero* elsewhere in (9a), we get mass conservation in (10a). Denote u_h and v_h as the two components of the velocity approximation $\boldsymbol{u}_h$. Taking test function $e_h := \frac{1}{2} u_h$ on element K and *zero* elsewhere in (9a), and $\boldsymbol{v}_h := (1, 0)$ on element K and *zero* elsewhere in (9b) and adding, we get the following:

$$\begin{aligned}\frac{d}{dt}\int_K h_h u_h\,\mathrm{dx} = &-\int_{\partial K} (\widehat{h_h \boldsymbol{u}_h \boldsymbol{u}_h}) n_x\,\mathrm{ds} - \int_K g h_h \partial_x (h_h + b_h)\,\mathrm{dx} \\ &+ \int_{\partial K} \frac{1}{2} g h_h [\![(h_h + b_h)]\!] n_x\,\mathrm{ds},\end{aligned}$$

where n_x is the first component of the normal direction $\boldsymbol{n}_K$. Combining the above identity with the fact that

$$\int_K g h_h \partial_x (h_h)\,\mathrm{dx} - \int_{\partial K} \frac{1}{2} g h_h^2 n_x\,\mathrm{ds} = 0,$$

we get the first component of the momentum balance identity (10b). We can apply the same argument to obtain the second component of the momentum balance identity (10b).

Taking test function $e_h = g(h_h + b_h)$ in (9a) and $\boldsymbol{v}_h = \boldsymbol{u}_h$ in (9b) and adding, we get

$$\begin{aligned}&M_h((h_h)_t, g(h_h + b_h)) + \boldsymbol{M}_h((h_h \boldsymbol{u}_h)_t, \boldsymbol{u}_h) - \frac{1}{2} M_h((h_h)_t, \boldsymbol{u}_h \cdot \boldsymbol{u}_h) \\ &= \underbrace{-B_h((h_h, \boldsymbol{u}_h), \boldsymbol{u}_h) + A_h\Big((h_h, \boldsymbol{u}_h), \frac{1}{2}\boldsymbol{u}_h \cdot \boldsymbol{u}_h\Big)}_{:=I_1} \\ &\quad \underbrace{-C_h((h_h, \boldsymbol{u}_h), \boldsymbol{u}_h) - A_h\Big((h_h, \boldsymbol{u}_h), g(h_h + b_h)\Big)}_{:=I_2}.\end{aligned}$$

Simplifying the above equality, we yield the entropy dissipation equality (10c). More specifically, it is easy to show that the left hand side of the above equality is the entropy dissipation

rate $\frac{d}{dt}E_h$, and the first term in the above right hand side

$$I_1 = -\sum_{F\in\mathcal{E}_h}\int_F \frac{1}{2}\alpha_h\{h_h+b_h\}[\![\boldsymbol{u}_h]\!]\cdot[\![\boldsymbol{u}_h]\!]\,\mathrm{ds}, \tag{11}$$

and the second term

$$I_2 = -\sum_{F\in\mathcal{E}_h}\int_F \frac{1}{2}\alpha_h g[\![h_h+b_h]\!]^2\,\mathrm{ds},$$

Below we only give detailed proof of the identity for I_1. We have

$$\begin{aligned} I_1 =& -B_h((h_h,\boldsymbol{u}_h),\boldsymbol{u}_h)+A_h\Big((h_h,\boldsymbol{u}_h),\frac{1}{2}\boldsymbol{u}_h\cdot\boldsymbol{u}_h\Big)\\ =& \underbrace{\sum_{K\in\Omega_h}\int_K\left(h_h(\boldsymbol{u}_h\otimes\boldsymbol{u}_h):\nabla\boldsymbol{u}_h-\frac{1}{2}h_h\boldsymbol{u}_h\cdot\nabla(\boldsymbol{u}_h\cdot\boldsymbol{u}_h)\right)\mathrm{dx}}_{\equiv 0}\\ &-\sum_{F\in\mathcal{E}_h}\int_F\left(\widehat{(h_h\boldsymbol{u}_h\boldsymbol{u}_h)}\boldsymbol{n}\cdot[\![\boldsymbol{u}_h]\!]-\widehat{h_h\boldsymbol{u}_h}\cdot\boldsymbol{n}[\![\frac{1}{2}\boldsymbol{u}_h\cdot\boldsymbol{u}_h]\!]\right)\mathrm{ds}. \end{aligned}$$

Now by the definition of the numerical fluxes in (9h)–(9i) and the simple fact that $[\![ab]\!]=\{a\}[\![b]\!]+[\![a]\!]\{b\}$, we have

$$\widehat{(h_h\boldsymbol{u}_h\boldsymbol{u}_h)}\boldsymbol{n}\cdot[\![\boldsymbol{u}_h]\!]-\widehat{h_h\boldsymbol{u}_h}\cdot\boldsymbol{n}[\![\frac{1}{2}\boldsymbol{u}_h\cdot\boldsymbol{u}_h]\!]=\frac{1}{2}\alpha_h\{h_h+b_h\}[\![\boldsymbol{u}_h]\!]\cdot[\![\boldsymbol{u}_h]\!],$$

which proves the identity for I_1 in (11).

Finally, under the assumption (10d), it is trivial to show that the spatial operators A_h, B_h, C_h in (9) all stay *zero*. Hence, $(h_h)_t\equiv 0$ from equation (9a), and $(h_h\boldsymbol{u}_h)_t=\frac{1}{2}(h_h)_t\boldsymbol{u}_h\equiv 0$ from (9b). This implies the well-balanced property (10e). □

Remark 2.1 (Comparison with other entropy stable DG schemes) Our first order scheme with polynomial degree $k=0$ is closely related to the first order finite volume entropy stable schemes [20, 21], as both approaches use the concept of entropy conservative/stable fluxes, cf. (9h)–(9i). The work [20, 21] promote to use a Roe-type dissipation operator in the numerical flux, while our numerical dissipation is of the more dissipative Lax-Friedrichs type.

For our high-order DG scheme with $k\geq 1$, we were not able to find similar work in the literature. There are two main approaches to construct high-order entropy stable schemes for nonlinear conservation laws, both stem from Tadmor's pioneer work on entropy variables and entropy conservative/stable fluxes [38, 39]. The first approach directly discretizes the conservation equations using the entropy variables [38], see also [27]. The major drawback of this approach is that explicit time stepping is usually not applicable to these schemes due to the highly nonlinear mapping between the entropy variables and the conservative variables. Hence they are generally more expensive than other explicit schemes. The second approach is based on the (quadrature-based) SBP operator concept, and has undergone a major development in the past few years, see, e.g., the entropy stable DG schemes [24, 44–47] for SWEs. These quadrature-based approaches may lead to accuracy loss, and they may be more cumbersome to implement on unstructured triangular meshes than classical DG schemes; see the more discussion in the recent review work [12].

Our scheme (9) combines both advantages of the above mentioned approaches:

- the proof of entropy stability can be performed directly on the variational formation (9) without converting to any matrix-vector form. This is made possible due to the use of velocity approximation and the skew-symmetrization of the momentum balance equation. In particular, we have the entropy stability result (10c) for the DG scheme (9) with any choice of numerical quadrature rule. This is possible because the proof of (10c) does not rely on integration by parts as the operators have already been properly skew-symmetrized. Hence, the quadrature rule can be chosen only for accuracy considerations. In our numerical experiments, we simply use Gauss quadrature rules that are exact for integrating polynomials of degree $2k$.
- the semi-discrete scheme (9) can be discretized in time using classical explicit time stepping schemes. This is possible because we discretize the water height h_h as the solution unknown.

Remark 2.2 (Comment on the well-balanced property) The well-balanced property (10e) of the proposed scheme preserves a steady constant surface elevation level C that is above the maximal bottom topography, i.e. $C > \max b_h$. The more challenging case with $C < \max b_h$ is not considered in this work and will be investigated in the near future. We note that when $C < \max b_h$, the "lake-at-rest" solution became

$$u = v = 0, h + b = \max\{C, b\},$$

where the domain contains dry regions with zero water height. The above steady state solution is particularly difficult to preserve in the case when the mesh is not aligned with the wet/dry interface, which typically requires special numerical flux treatments; see [3, Section 4]. However, the approach proposed in [3, Section 4] is not directly applicable to our scheme on unstructured triangular meshes as the technique relies on a strong nodal DG formulation on quadrilateral meshes.

3 Fully Discrete Scheme: Local Conservation, Well-Balanced Property, Positivity Preservation, and Slope Limiting

In this section, we discrete the semi-discrete scheme (9) in time using explicit SSP-RK time integrators. Special attention is paid to maintain the local conservation, well-balanceness, and positivity preservation properties. We also discuss the use of a characteristic-wise TVB slope limiter [15] in combination with the recent troubled-cell indicator proposed in [22] to improve its efficiency. The slope limiter, which suppress numerical oscillations near shock discontinuities, is a crucial component for the accuracy and robustness of the overall scheme for polynomial degree $k \geq 1$, c.f. [16]. Moreover, we propose a simple wetting/drying treatment for the velocity calculation near dry cells where water height is small.

3.1 A Three-Field Reformulation of the Semi-Discrete Scheme (9)

Here we introduce a three-field reformation of the semi-discrete DG scheme (9) by using the discharge $\boldsymbol{m} := h\boldsymbol{u}$ as an additional independent unknown, which is then discretized in time using the explicit SSP-RK method. The three-field DG scheme reads as follows: find $(h_h, \boldsymbol{u}_h, \boldsymbol{m}_h) \in V_h^k \times \boldsymbol{V}_h^k \times \boldsymbol{V}_h^k$ such that

$$\begin{aligned} &M_h\left((h_h)_t, e_h\right) + A_h((h_h, \boldsymbol{u}_h), e_h) = 0, \\ &\boldsymbol{M}_h\left((\boldsymbol{m}_h)_t, \boldsymbol{v}_h\right) + B_h((h_h, \boldsymbol{u}_h), \boldsymbol{v}_h) + C_h((h_h, \boldsymbol{u}_h), \boldsymbol{v}_h) \end{aligned} \tag{12a}$$

$$-M_h\left((h_h)_t, \frac{1}{2}\boldsymbol{u}_h \cdot \boldsymbol{v}_h\right) - A_h\left((h_h, \boldsymbol{u}_h), \frac{1}{2}\boldsymbol{u}_h \cdot \boldsymbol{v}_h\right) = 0, \tag{12b}$$

$$\boldsymbol{M}_h\left(h_h\boldsymbol{u}_h - \boldsymbol{m}_h, \boldsymbol{w}_h\right) = 0, \tag{12c}$$

for all $(e_h, \boldsymbol{v}_h, \boldsymbol{w}_h) \in V_h^k \times \boldsymbol{V}_h^k \times \boldsymbol{V}_h^k$. We have the following equivalence of the two formulations (9) and (12).

Theorem 3.1 *Let $(h_h, \boldsymbol{u}_h, \boldsymbol{m}_h)$ be the solution to the three-field DG formulation (12). Then $(h_h, \boldsymbol{u}_h)$ is the solution to the two-field DG formulation (9).*

Proof Let $(h_h, \boldsymbol{u}_h, \boldsymbol{m}_h)$ be the solution to the scheme (12). Equation (12c) simply implies that $\boldsymbol{m}_h$ is the L^2-projection of $h_h\boldsymbol{u}_h$ onto the DG space $\boldsymbol{V}_h^k$. Since this projection commutes with the time derivative term, we have $\boldsymbol{M}_h\left((\boldsymbol{m}_h)_t, \boldsymbol{v}_h\right) = \boldsymbol{M}_h\left((h_h\boldsymbol{u}_h)_t, \boldsymbol{v}_h\right)$, for all $\boldsymbol{v}_h \in \boldsymbol{V}_h^k$. Combining this equality with (12b), we have that $(h_h, \boldsymbol{u}_h)$ satisfies equation (9b). Since $(h_h, \boldsymbol{u}_h)$ also satisfies equation (12a), which is identical to (9a), we conclude that $(h_h, \boldsymbol{u}_h)$ is the solution to the system (9). □

The advantage of this reformulation will be clear next when we discuss SSP-RK time discretizations and slope limiting.

3.2 High Order SSP-RK Discretization and Inner Stage Reconstruction

The semi-discrete scheme (12) is not a standard ODE system $U_t + \mathcal{F}(U) = 0$, with U being the solution vector and $\mathcal{F}(U)$ the spatial operator, as the time derivative terms in (12b) involve the nonlinear product $h_h\boldsymbol{u}_h$, and (12c) is an algebraic equation. As a result, special care is need in design locally conservative high-order time discretizations. Here we apply the third order SSP-RK3 scheme to (12) which preserves the local conservation property. It is built on top of a plain forward Euler discretization in Algorithm 1, a velocity update in Algorithm 2, and a convex combination step in Algorithm 3. The full plain SSP-RK3 algorithm without slope limiting is given in Algorithm 4. Here the weights w_1 and w_2 in Step 4/7 of Algorithm 4 are the RK3 weights from Shu and Osher [37, eqn (2.18)].

Algorithm 1 Plain Forward Euler + DG

Input: $h_h^{old} \in V_h^k$, $\boldsymbol{u}_h^{old}, \boldsymbol{m}_h^{old} \in \boldsymbol{V}_h^k$, and time step size $\Delta t > 0$.
Output: $h_h^{new} \in V_h^k$, and $\boldsymbol{m}_h^{new} \in \boldsymbol{V}_h^k$.
1: Compute h_h^{new} by the following equation:

$$M_h(h_h^{new}, e_h) = M_h(h_h^{old}, e_h) - \Delta t A_h((h_h^{old}, \boldsymbol{u}_h^{old}), e_h), \quad \forall e_h \in V_h^k,$$

2: Compute $\boldsymbol{m}_h^{new}$ by the following equation:

$$\begin{aligned}\boldsymbol{M}_h(\boldsymbol{m}_h^{new}, \boldsymbol{v}_h) = \boldsymbol{M}_h(\boldsymbol{m}_h^{old}, \boldsymbol{v}_h) - \Delta t B_h((h_h^{old}, \boldsymbol{u}_h^{old}), \boldsymbol{v}_h) - \Delta t C_h((h_h^{old}, \boldsymbol{u}_h^{old}), \boldsymbol{v}_h) \\ + \frac{1}{2}M_h\left(h_h^{new} - h_h^{old}, \boldsymbol{u}_h^{old} \cdot \boldsymbol{v}_h\right) + \frac{1}{2}\Delta t A_h\left((h_h^{old}, \boldsymbol{u}_h^{old}), \boldsymbol{u}_h^{old} \cdot \boldsymbol{v}_h\right), \quad \forall \boldsymbol{v}_h \in \boldsymbol{V}_h^k,\end{aligned}$$

Algorithm 2 Velocity update

Input: $h_h \in V_h^k, \boldsymbol{m}_h \in \boldsymbol{V}_h^k$.
Output: $\boldsymbol{u}_h \in \boldsymbol{V}_h^k$.
1: Compute $\boldsymbol{u}_h$ by the following equation:

$$\boldsymbol{M}_h(h_h \boldsymbol{u}_h, \boldsymbol{w}_h) = \boldsymbol{M}_h(\boldsymbol{m}_h, \boldsymbol{w}_h), \quad \forall \boldsymbol{w}_h \in \boldsymbol{V}_h^k.$$

Algorithm 3 Convex combination

Input: Data $h_h^1, h_h^2 \in V_h^k, \boldsymbol{m}_h^1, \boldsymbol{m}_h^2 \in \boldsymbol{V}_h^k$. Positive weights w_1, w_2 with $w_1 + w_2 = 1$.
Output: $h_h \in V_h^k$, and $\boldsymbol{m}_h \in \boldsymbol{V}_h^k$.
1: Compute h_h and $\boldsymbol{m}_h$ using convex combination:

$$h_h \leftarrow w_1 h_h^1 + w_2 h_h^2, \quad \boldsymbol{m}_h \leftarrow w_1 \boldsymbol{m}_h^1 + w_2 \boldsymbol{m}_h^2.$$

Algorithm 4 Plain SSP-RK3 + DG

Input: $h_h^n \in V_h^k, \boldsymbol{u}_h^n, \boldsymbol{m}_h^n \in \boldsymbol{V}_h^k$ at time level t^n, and time step size $\Delta t > 0$.
Output: $h_h^{n+1} \in V_h^k$, and $\boldsymbol{u}_h^{n+1}, \boldsymbol{m}_h^{n+1} \in \boldsymbol{V}_h^k$ at next time level $t^{n+1} := t^n + \Delta t$.
1: Apply Algorithm 1 with inputs $h_h^n, \boldsymbol{u}_h^n, \boldsymbol{m}_h^n$ and Δt. Denote the outputs as $h_h^{(1)}, \boldsymbol{m}_h^{(1)}$.
2: Apply Algorithm 2 with inputs $h_h^{(1)}, \boldsymbol{m}_h^{(1)}$. Denote the velocity output as $\boldsymbol{u}_h^{(1)}$.
3: Apply Algorithm 1 with inputs $h_h^{(1)}, \boldsymbol{u}_h^{(1)}, \boldsymbol{m}_h^{(1)}$ and Δt. Denote outputs as $h_h^{(2*)}, \boldsymbol{m}_h^{(2*)}$.
4: Apply Algorithm 3 with inputs $h_h^n, h_h^{(2*)}, \boldsymbol{m}_h^n, \boldsymbol{m}_h^{(2*)}$ and weights $w_1 = 0.75, w_2 = 0.25$. Denote outputs as $h_h^{(2)}, \boldsymbol{m}_h^{(2)}$.
5: Apply Algorithm 2 with inputs $h_h^{(2)}, \boldsymbol{m}_h^{(2)}$. Denote the velocity output as $\boldsymbol{u}_h^{(2)}$.
6: Apply Algorithm 1 with inputs $h_h^{(2)}, \boldsymbol{u}_h^{(2)}, \boldsymbol{m}_h^{(2)}$ and Δt. Denote outputs as $h_h^{(3*)}, \boldsymbol{m}_h^{(3*)}$.
7: Apply Algorithm 3 with inputs $h_h^n, h_h^{(3*)}, \boldsymbol{m}_h^n, \boldsymbol{m}_h^{(3*)}$ and weights $w_1 = 1/3, w_2 = 2/3$. Denote outputs as $h_h^{n+1}, \boldsymbol{m}_h^{n+1}$.
8: Apply Algorithm 2 with inputs $h_h^{n+1}, \boldsymbol{m}_h^{n+1}$. Denote the velocity output as $\boldsymbol{u}_h^{n+1}$.

Remark 3.1 (Local conservation and well-balanced property) Similar to the proof of local conservation in Theorem 2.1, the forward Euler algorithm Algorithm 1 is also locally conservative. Meanwhile, the local conservation property is not polluted by the convex combination step in Algorithm 3 due to the use of conservative variables in convex combination. If the velocity $\boldsymbol{u}_h$ were to be used in the convex combination step, local conservation for the discharge would be lost. This is the major reason that the discharge $\boldsymbol{m}_h$ is re-introduced as an independent unknown in the DG formulation. Hence the overall algorithm Algorithm 4 is locally conservative. Moreover, it is easy to see that Algorithm 4 preserve the steady state solution (6), hence it is also well-balanced.

Remark 3.2 (Computational cost and positivity of water height) The computational cost of Algorithm 1 involves the evaluation of the right hand side operators, and solving the associated linear system for the mass matrix for V_h^k in Step 1, and for $\boldsymbol{V}_h^k$ in Step 2. The mass matrices can be made diagonal if orthogonal L^2-basis is used in the computation. The computational cost of Algorithm 2 involves the linear system solve of a water height-weighted mass matrix, which is block diagonal and can be computed very efficiently. The Algorithm 3 is simply a vector update. Hence, the computational cost in Algorithm 4 is of linear complexity with

respect to the total number of elements, which is similar to, but slightly more expensive than (due to the velocity computation in Algorithm 2), the cost of a classical DG scheme with SSP-RK3 time stepping.

Note that in Algorithm 2, we need to invert the water height-weighted mass matrix to compute the velocity approximation $\boldsymbol{u}_h$. This weighted mass matrix may fail to be invertible if the water height approximation h_h become negative in parts of the domain. In practice, requiring positivity of water height on the volume integration points used to compute these mass matrices suffice to ensure its invertibility, which, however, is not guaranteed in the plain Algorithm 4. In the next subsection, we apply the positivity-preserving limiting approach used in [51] to guarantee such positivity requirement.

3.3 Hydrostatic Reconstruction and Posivitivity-Preserving Limiter

The key idea of the posivity-preserving limiter in [51] is to ensure the cell average of water height is positive after one step of forward Euler time stepping under a usual CFL condition. Given solution $h_h^n, \boldsymbol{u}_h^n$ at time t^n, and time step size Δt, the water height h_h^{n+1} at next time level for the forward Euler Algorithm 1 reads as follows:

$$M_h(h_h^{n+1}, e_h) = M_h(h_h^n, e_h) - \Delta t A_h((h_h^n, \boldsymbol{u}_h^n), e_h), \quad \forall e_h \in V_h^k.$$

Taking test function $e_h = 1$ on a single element $K \in \Omega_h$, we get the evolution equation for the water height cell average:

$$\bar{h}_K^{n+1} = \bar{h}_K^n - \frac{\Delta t}{|K|} \int_{\partial K} \widehat{h_h^n \boldsymbol{u}_h^n} \cdot \boldsymbol{n}_K \, \mathrm{ds} \tag{13}$$

where $\bar{h}_K^n$ stands for the cell average of h_h on the triangle K at time level t^n, and $|K|$ is the area of the element K. Due to the fact that the numerical flux (9h) contains the jump of bottom topography b_h, which can arbitrarily large, we can not prove positivity of $\bar{h}_K^{n+1}$ in the above equation (13) under the condition of positivity of $\bar{h}_K^n$ and a reasonable time step size restriction. To fix this, we slightly modify the numerical fluxes using the idea of hydrostatic reconstruction [2, 51]. In particular, introducing the following hydrostatic reconstructed version of the water height:

$$h_h^{*,+} := \max\left\{0, h^+ + \min\{0, [\![b_h]\!]\}\right\}, \tag{14a}$$

$$h_h^{*,-} := \max\left\{0, h^- - \max\{0, [\![b_h]\!]\}\right\}, \tag{14b}$$

we replace $h_h^{\pm}$ in the flux terms in the scheme (9) by $h_h^{*,\pm}$, and replace the associated jump term $[\![h_h + b_h]\!]$ by $[\![h_h^*]\!] = h_h^{*,+} - h_h^{*,-}$. For example, the flux (9h) is now replaced by the following one:

$$\widehat{h_h^* \boldsymbol{u}_h} \cdot \boldsymbol{n} := \{h_h^* \boldsymbol{u}_h\} \cdot \boldsymbol{n} + \frac{1}{2} \alpha_h^* [\![h_h^*]\!], \tag{15}$$

with

$$\alpha_h^*|_F := \max\left\{\sqrt{g h_h^{*,+}} + |\boldsymbol{u}_h^+ \cdot n|, \sqrt{g h_h^{*,-}} + |\boldsymbol{u}_h^- \cdot n|\right\}. \tag{16}$$

It is clear that if h_h satisfies $h_h + b_h = Const$ with $h_h > 0$, then

$$h_h^{*,+} = h^+ + \min\{0, [\![b_h]\!]\},$$
$$h_h^{*,-} = h^- - \max\{0, [\![b_h]\!]\},$$

and $[\![h_h^*]\!] \equiv [\![h_h + b_h]\!]$, which implies that the modified fluxes will not pollute the well-balanced property of the original fluxes.

With this modification, the forward Euler discretization lead to the following cell average evolution for water height:

$$\begin{aligned}\bar{h}_K^{n+1} &= \bar{h}_K^n - \frac{\Delta t}{|K|}\int_{\partial K} \widehat{h_h^{*,n}\boldsymbol{u}_h^n}\cdot \boldsymbol{n}_K \,\mathrm{ds} \\ &= \bar{h}_K^n - \frac{\Delta t}{|K|}\int_{\partial K}\left(\{h_h^{*,n}\boldsymbol{u}_h^n\}\cdot\boldsymbol{n}_K + \frac{1}{2}\alpha_h^{*,n}[\![h_h^{*,n}]\!]_K\right)\mathrm{ds},\end{aligned} \tag{17}$$

where $[\![h_h]\!]_K := h_h^{int(K)} - h_h^{ext(K)}$ is the jump, with $h_h^{int(K)}$ and $h_h^{ext(K)}$ being the approximations obtained from the interior and the exterior of K. Note that by definition, on any edge $F = K^+ \cap K^-$ shared by two elements, there holds

$$[\![h_h]\!] = h_h|_{K+} - h_h|_{K^-} = [\![h_h]\!]_{K^+} = -[\![h_h]\!]_{K^-}.$$

The cell average evolution equation (13) now has a similar form as [51, Equation 3.1]. Hence, we can follow the same analysis in [51, Section 3] to ensure positivity of the water height cell average at next time level. The following result is Theorem 3.2 in [51]. The proof is almost identical, hence we omit it for simplicity.

Theorem 3.2 (Theorem 3.2 in [51]) *For the scheme (17) to be positivity preserving, i.e., $\bar{h}_K^{n+1} \geq 0$, a sufficient condition is that $h_K(\boldsymbol{x}) \geq 0$, $\forall \boldsymbol{x} \in S_K$ for all K, under the CFL condition*

$$\alpha_{\partial K}^* \frac{\Delta t}{|K|}|\partial K| \leq \frac{2}{3}\widehat{w}_1. \tag{18}$$

Here $h_K(\boldsymbol{x})$ denotes the polynomial for water height at time level n, S_K is a set of (symmetric) quadrature points on K that includes $k+1$ Gauss quadrature points on each boundary edge, $\alpha_{\partial K}^* := \max_{F\subset\partial K}\alpha_h^*|_F$ is the maximum estimated speed on ∂K, $|\partial K|$ is the perimeter of element K, and $\widehat{w}_1$ is the quadrature weight of the $\lceil\frac{k+3}{2}\rceil$-point Gauss-Lobatto rule on $[-1/2, 1/2]$ for the first quadrature point.

At time level n, given the water height DG polynomial $h_K(\boldsymbol{x})$ with its cell average $\bar{h}_K^n \geq 0$, we use the simple scaling limiter [51, Section 3.4] to ensure the above sufficient condition $h_K(\boldsymbol{x}) \geq 0$ for all $\boldsymbol{x} \in S_K$, i.e., replacing $h_K(\boldsymbol{x})$ by a linear scaling around the cell average:

$$\widetilde{h}_K(\boldsymbol{x}) = \theta_K(h_K(\boldsymbol{x}) - \bar{h}_K^n) + \bar{h}_K^n, \tag{19}$$

where $\theta_K \in [0, 1]$ is determined by

$$\theta_K := \min_{\boldsymbol{x}\in S_K}\theta_{\boldsymbol{x}}, \quad \theta_{\boldsymbol{x}} = \min\left\{1, \frac{\bar{h}_K^n}{\bar{h}_K^n - h_K(\boldsymbol{x})}\right\}. \tag{20}$$

A slightly more efficient and less restrictive scaling parameter θ_K can be obtained using a reduced set of quadrature points, see [51, Section 3.4] for more details. In practice, the positivity preserving limiter (19) is applied in each inner stage of the RKDG algorithm 4. We notice that the well-balanced property is also not affected by this positivity preserving limiter.

Remark 3.3 (On hydrostatic reconstruction and bottom topography approximation) The proof of Theorem 3.2 requires the use of hydrostatic reconstruction (14), which is needed due to the lack of control of the bottom topography jump $[\![b_h]\!]$ across edges. When the polynomial

degree $k \geq 1$ in the DG scheme (9), one can approximate the bottom topography using a continuous approximation $b_h \in V_h^k \cap H^1(\Omega)$, which implies $[\![b_h]\!] = 0$. In this case, under the positivity assumption of Theorem 3.2, we have $h_h^{*,\pm} = h_h^{\pm}$, hence equivalence of the original scheme (13) and the reconstructed version (17). For this reason, we prefer to use a continuous bottom topography approximation for $k \geq 1$, where the hydrostatic reconstruction (14) is not necessary anymore. In the future, we plan to investigate more modern hydrostatic reconstruction techniques that are more robust/accurate for abruptly changing bottom topographies such as the ones proposed in [9, 11].

3.4 The Troubled-Cell Indicator and Slope Limiter

Another important ingredient of the DG methods is the slope limiter procedure which is needed to suppress spurious oscillations near solution discontinuities. We follow the standard slope limiting procedure in RKDG methods [16, 32]:

(1) First we identify the *troubled cells*, namely, those cells which might need the limiting procedure.
(2) Second we replace the solution polynomials in those troubled cells by reconstructed polynomials with limited slopes that maintain the original cell averages (conservation).

We use the Fu-Shu troubled-cell indicator proposed in [22] to identify the troubled cells, with a scaling modification to improve its performance and computational efficiency. Given a discontinuous function $p \in V_h^k$, the troubled-cell indicator [22] is given as follows:

$$I_K(p) = \frac{\sum_{T \in \omega(K)} |\bar{\bar{p}}_T - \bar{p}_K|}{\bar{p}_{\max} - \bar{p}_{\min}}, \tag{21}$$

where $\omega(K)$ is the union of cells that share a common edge with K, including K itself, and $\bar{\bar{p}}_T$ is the cell average of the polynomial $p|_T$ extended to the target cell K, and $\bar{p}_{\max}$ and $\bar{p}_{\min}$ are the global maximal and minimal cell average on the domain. Relying on the *assumption* [15, 16] that spurious oscillations are present in the solution p_h only if they are present in its linear part p_h^1, which is its L^2-projection into the space of piecewise linear functions V_h^1, we use use the linear L^2-projection of the total height $h_h + b_h$ as the indicating function in (21), which simplifies the implementation of the extended cell average $\bar{\bar{h}}_h$ for high-order case where the polynomial degree $k > 1$. The cell K is marked as a troubled cell if

$$I_K(h_h^1 + b_h^1) > tol, \tag{22}$$

where tol is a user defined parameter. Note that this indicator is of $\mathcal{O}(\tau_K^2)$ in smooth regions, where τ_K is the mesh size, and of $\mathcal{O}(1)$ near discontinuities, hence is expected to be effective to detect troubled cells near discontinuities with a proper choice of tol. Our numerical experiments suggest that the indicator is not too sensitive to the tolerance tol. The indicator with $tol \in (0.01, 0.1)$ performs similarly for most of the examples, where $tol = 0.01$ leads to a slightly larger number of detected troubled cells than $tol = 0.1$ as expected. In our implementation, we take $tol = 0.02$ for all the reported results.

Remark 3.4 (On scaling of the indicator (21)) The original indicator proposed in [22] use the local maximal cell average $\max_{T \in \omega(K)}\{\bar{p}_T\}$ as the scaling denominator. This scaling has the drawback of not able to detect any troubled cells for small perturbation tests where the total height is a very small perturbation of a constant state. In particular, the original indicator with a local maximum scaling will produce a completely different result when the

indicating function is perturbed by a global constant $p(x) \leftarrow p(x) + Const$. The new global difference scaling denominator $\bar{p}_{\max} - \bar{p}_{\min}$ now produce the same indicating value when the indicating function is perturbed by a global constant. It performs quite well for all the numerical examples reported here. We further mention that this global scaling is suggested to us by Prof. Chi-Wang Shu from Brown University in a private communication.

After the troubled cells have been detected, we apply the characteristic-wise TVB limiter [15, 16] on the conservative variables $(h_h + b_h, \boldsymbol{m}_h)$ with TVB parameter $M = 0$. To save space, we leave out the derivation of this limiter and refer to [15, 16] for details. We mention that while this TVB limiter is compatible with the well-balanced property of the DG scheme as $h_h + b_h$ is used in the limiting process. In practice, we first apply this TVB limiter then apply the positivity preserving limiter (19) for each inner Runge-Kutta stage values.

3.5 Velocity Computation and Dry Cell Treatment

We note that while Theorem 3.2 and the limiter (19) ensures non-negativity of the water height cell average $\bar{h}_K^{n+1}$, and water height on the quadrature points S_K at the next time level, this in general is not enough for the invertibility of the water height-weighted mass matrix, which is needed to compute the velocity approximation. The invertibility of this weighted mass matrix is not a big issue as one can compute the scaling factor in (20) such that it ensures positivity of water height on all volume integration points in each cell.

A more serious issue is the velocity computation on dry cells with nearly zero water height, this weighted mass matrix is nonsingular but now close to zero, and the computed velocity approximation may be unphysically large and not reliable anymore. Without a special velocity treatment on dry cells, the scheme (with TVB and posivity preserving limiters) may still fail to solve challenging problems with moving interface with wet and dry areas. There are various wetting/drying treatment available in the literature [6, 28]. However, our preliminary numerical experiments suggest that the most common approaches may not work well for our velocity based DG scheme. For example, the simple trick of setting zero velocity when the water height h_h is less than a given threshold, e.g. 10^{-6}, which worked in [52], or using a regularized water height

$$h^* := \frac{1}{2}h + \frac{1}{2}\max\{h, (tol)^2/h\},$$

with tol a given small tolerance, to compute the weighted mass matrix in Algorithm 2 as suggested in [28] were not enough for our scheme with polynomial degree $k = 2$ to solve a dam break problem on a dry bed.

After some initial testing, we came up with a relative simple velocity limiting approach that works for the numerical results reported in this manuscript. We apply the following two steps after an inner stage water height h_h and discharge $\boldsymbol{m}_h$ has been computed by Algorithm 1:

(1) Given a threshold percentage $0 < \epsilon_d \ll 1$, we first mark cells with cell average $\bar{h}_K \le \epsilon_d \times h^0_{\max}$ as *dry* cells, where $h^0_{\max}$ is the maximum water height at initial time. Then, we remove the *high order* information on these dry cells by reverting to piecewise constant approximation of water height and discharge:

$$h_K \leftarrow \bar{h}_K, \quad \boldsymbol{m}_K \leftarrow \bar{\boldsymbol{m}}_K, \quad \text{for all } K \in \Omega_h \text{ such that } \bar{h}_K \le \epsilon_d \times h^0_{\max}, \tag{23}$$

where h_K and $\boldsymbol{m}_K$ are the polynomial data in cell K, and $\bar{h}_K$ and $\bar{\boldsymbol{m}}_K$ are the cell averages. Note that this approach does not affect the local conservation property, but may lead to

accuracy loss. However, since there are only a small amount of water in dry cells, such loss of accuracy may not be too significant if ϵ_d is taken small enough. We note that similar treatment was used in [6].

(2) The above approach may not be enough to control the velocity magnitude for high order schemes when ϵ_d is taken to be too small. We further propose a velocity limiter to smooth out extreme velocity values. Given a user tunable value $V_{\max}$, which is an estimation of maximal allowed velocity approximation, we do the following two steps for each component of the velocity approximation:

(i) Identify the collection of *troubled velocity cells*, denoted as $\omega(u_h)$, for the velocity component u_h as the cells where the maximum of the absolute velocity is larger than $V_{\max}$. For polynomial degree $k = 2$ on triangles, the maximal value in the triangle is estimated as the maximal value on three vertices and three mid points of each edge:

$$\omega(u_h) := \{K \in \Omega_h : \max_{x \in v(K)} |u_h(x)| > V_{\max}, \} \tag{24}$$

where $v(K)$ is the collection of three vertices and three edge midpoints of cell K.

(ii) On each of these troubled cells, we remove the velocity data, and replace it by the average of cell averages of its immediate neighboring cells which are not marked as troubled cells. We repeat this procedure until all troubled cells have an updated (constant) velocity value:

$$\begin{aligned} &\text{While } \omega(u_h) \text{ is not empty, do the following:} \\ &\quad u_K \leftarrow \text{average of } \{\bar{u}_T\} \text{ for } T \in \omega(K) \text{ and } T \notin \omega(u_h). \\ &\quad \text{remove cell } K \text{ from } \omega(u_h) \text{ if its value has been updated.} \end{aligned} \tag{25}$$

We note that the above velocity limiting procedure does not affect the local conservation property as the water height and discharge cell averages were never changed. The above two approaches introduce two tunable parameters, namely ϵ_d in (23), and $V_{\max}$ in (24). They will be chosen accordingly for specific wetting/drying examples. For example, we can take $\epsilon_d = 5 \times 10^{-3}$, and take $V_{\max}$ based on the maximum velocity magnitude for the lowest order scheme with $k = 0$ for problems with moving wet/dry interfaces. With the above wetting/drying treatment, we are able to run simulation for the circular dam break problem with a dry bed, and the dam bream problem with three mounds on unstructured triangular grids. We mention that the above treatments are far away from perfect, as they need parameter tuning, and may lead to accuracy loss near dry cells. They only serves as initial approaches for a successful simulation of SWEs with moving wet/dry interfaces. We are planning to further investigate more robust and accurate wetting and drying treatments for our velocity based DG scheme in the near future.

For completeness, we list the final form of the fully discrete scheme below. This method is locally conservative, well-balanced, and positivity preserving provided the time step size Δt satisfy the CFL condition (18). In practice, we take the time step size to be

$$\Delta t = cfl \min_{K \in \Omega_h} \{\tau_K / \alpha_K^{\max}\}, \tag{26}$$

where cfl is the CFL number which depends on the polynomial degree k, τ_K is the mesh size, and $\alpha_K^{\max}$ is the estimated maximum speed on the cell K. If we detect a water height cell average $\bar{h}_K < \epsilon = 10^{-12}$ in the inner stages in Step 1/2/4 of Algorithm 5, which means the time step size does not satisfy the condition (18), we simply decrease Δt by a half and redo the whole computation.

Algorithm 5 Posivity-preserving SSP-RK3 + DG + TVB limiter + wetting/drying treatment

Input: $h_h^n \in V_h^k, \boldsymbol{u}_h^n, \boldsymbol{m}_h^n \in \boldsymbol{V}_h^k$ at time level t^n, and time step size $\Delta t > 0$. $tol > 0$ for TVB limiter indicator (21), $\epsilon_d > 0$ for dry cell indicator (23), and $V_{\max} > 0$ for troubled velocity cell indicator (24),

Output: $h_h^{n+1} \in V_h^k$, and $\boldsymbol{u}_h^{n+1}, \boldsymbol{m}_h^{n+1} \in \boldsymbol{V}_h^k$ at next time level $t^{n+1} := t^n + \Delta t$.

1: Apply Algorithm 1 with inputs $h_h^n, \boldsymbol{u}_h^n, \boldsymbol{m}_h^n$ and Δt. (If bottom topography b_h is discontinuous, apply the hydrostatic reconstruction (14) in flux evaluations.) Denote the outputs as $h_h^{(1)}, \boldsymbol{m}_h^{(1)}$.

2: Apply the dry cell limiter (23) for $h_h^{(1)}$ and $\boldsymbol{m}_h^{(1)}$; Apply the characteristic-wise TVB limiter for the variables $(h_h^{(1)} + b_h, \boldsymbol{m}_h^{(1)})$ using indicator (21) with indicating function $h_h^{(1)} + b_h$; Apply the positivity preserving limiter for $h_h^{(1)}$.

3: Apply Algorithm 2 with inputs $h_h^{(1)}, \boldsymbol{m}_h^{(1)}$. Denote the velocity output as $\boldsymbol{u}_h^{(1)}$. Then apply the velocity limiter in (25).

4: Apply Algorithm 1 with inputs $h_h^{(1)}, \boldsymbol{u}_h^{(1)}, \boldsymbol{m}_h^{(1)}$ and Δt. (If bottom topography b_h is discontinuous, apply the hydrostatic reconstruction (14) in flux evaluations.) Denote outputs as $h_h^{(2*)}, \boldsymbol{m}_h^{(2*)}$.

5: Apply Algorithm 3 with inputs $h_h^n, h_h^{(2*)}, \boldsymbol{m}_h^n, \boldsymbol{m}_h^{(2*)}$ and weights $w_1 = 0.75, w_2 = 0.25$. Denote outputs as $h_h^{(2)}, \boldsymbol{m}_h^{(2)}$.

6: Apply the dry cell limiter (23) for $h_h^{(2)}$ and $\boldsymbol{m}_h^{(2)}$; Apply the characteristic-wise TVB limiter for the variables $(h_h^{(2)} + b_h, \boldsymbol{m}_h^{(2)})$ using indicator (21) with indicating function $h_h^{(2)} + b_h$; Apply the positivity preserving limiter for $h_h^{(2)}$.

7: Apply Algorithm 2 with inputs $h_h^{(2)}, \boldsymbol{m}_h^{(2)}$. Denote the velocity output as $\boldsymbol{u}_h^{(2)}$. Then apply the velocity limiter in (25).

8: Apply Algorithm 1 with inputs $h_h^{(2)}, \boldsymbol{u}_h^{(2)}, \boldsymbol{m}_h^{(2)}$ and Δt. (If bottom topography b_h is discontinuous, apply the hydrostatic reconstruction (14) in flux evaluations.) Denote outputs as $h_h^{(3*)}, \boldsymbol{m}_h^{(3*)}$.

9: Apply Algorithm 3 with inputs $h_h^n, h_h^{(3*)}, \boldsymbol{m}_h^n, \boldsymbol{m}_h^{(3*)}$ and weights $w_1 = 1/3, w_2 = 2/3$. Denote outputs as $h_h^{n+1}, \boldsymbol{m}_h^{n+1}$.

10: Apply the dry cell limiter (23) for h_h^{n+1} and $\boldsymbol{m}_h^{n+1}$; Apply the characteristic-wise TVB limiter for the variables $(h_h^{n+1} + b_h, \boldsymbol{m}_h^{n+1})$ using indicator (21) with indicating function $h_h^{n+1} + b_h$; Apply the positivity preserving limiter for h_h^{n+1}.

11: Apply Algorithm 2 with inputs $h_h^{n+1}, \boldsymbol{m}_h^{n+1}$. Denote the velocity output as $\boldsymbol{u}_h^{n+1}$. Then apply the velocity limiter in (25).

4 Numerical Results

In this section we present numerical results of our velocity based DG scheme Algorithm (5). We report results using the third order DG method with $k = 2$. The CFL number is taken to be $cfl = 0.1$ for 1D examples, and $cfl = 0.05$ for 2D examples. The gravitation constant g is fixed as 9.812 except the test in Example 4.6, where it is taken to be $g = 10$. We take the tolerance $tol = 0.02$ in the Fu-Shu indicator (21) for all examples. Moreover, unless explicitly mentioned, we turn off the dry cell limiter (23) and the velocity limiter (24)–(25), which are only needed when the problem has a moving dry/wet interface.

Remark 4.1 (Implementation details and boundary conditions) The implementation is based on the python interface of the NGSolve software [34], https://ngsolve.org/. Source code for all the examples can be found in the git repository, https://github.com/gridfunction/SWE. Here we briefly comment on the implementation details of the third order DG scheme with $k = 2$ on a 2D triangular mesh. The Dubiner orthogonal bases [19] are used as the basis functions for the DG polynomial space V_h^2 of degree $k = 2$. A continuous interpolation in the space $V_h^2 \cap H^1(\Omega)$ is used for the bottom topography b_h so that hydrostatic reconstruction (14) is

not needed in the algorithm. We follow the suggestion in [14, Theorem 2.10] to use a 6-points Gauss-type quadrature rules of order $2k = 4$ for the numerical integration on triangles, and a 3-point Gauss-Legendre quadrature rule of order $2k + 1 = 5$ for the numerical integration on edges, which achieves formally $k + 1 = 3$rd order of accuracy. We specifically note that while these quadrature rules do not exactly integrate most of the terms in the DG scheme (9), the entropy stability result (10c) is not affected by the introduced numerical integration errors as the proof of Theorem 2.1 does not rely on exact integration.
Furthermore, numerical examples below require the implementation of wall/symmetry boundary conditions and outflow boundary conditions. We use the ghost cell technique [31] for numerical flux evaluations on edges that lie on the domain boundary, see also [1]. Specifically, let F be a boundary edge that belongs to cell K with polynomial data h_K and $\boldsymbol{u}_K$, we provide ghost cell data h_G and $\boldsymbol{u}_G$ exterior to F to evaluate the average and jump terms in (9h)–(9i), e.g., $\{h_h \boldsymbol{u}_h\}|_F := \frac{1}{2}(h_K \boldsymbol{u}_K + h_G \boldsymbol{u}_G)$. When F lies on a wall or symmetry boundary segment, we set $h_G^{(wall)} := h_K$, and $\boldsymbol{u}_G^{(wall)} := \boldsymbol{u}_K - 2(\boldsymbol{u}_K \cdot \boldsymbol{n})\boldsymbol{n}$, which implies that

$$\widehat{h_h \boldsymbol{u}_h} \cdot \boldsymbol{n}_F = 0, \quad (\widehat{h_h \boldsymbol{u}_h \boldsymbol{u}_h})\boldsymbol{n} = \alpha_K h_K (\boldsymbol{u}_K \cdot \boldsymbol{n}), \text{ where } \alpha_K = \sqrt{g h_K} + |\boldsymbol{u}_K \cdot \boldsymbol{n}|.$$

When F is an outflow edge, we simply use extrapolation: $h_G^{(out)} := h_K$, and $\boldsymbol{u}_G^{(out)} := \boldsymbol{u}_K$, which implies that

$$\widehat{h_h \boldsymbol{u}_h} \cdot \boldsymbol{n} = h_K \boldsymbol{u}_K \cdot \boldsymbol{n}, \quad (\widehat{h_h \boldsymbol{u}_h \boldsymbol{u}_h})\boldsymbol{n} = h_K \boldsymbol{u}_K (\boldsymbol{u}_K \cdot \boldsymbol{n}).$$

It is easy to show that the wall boundary condition is entropy stable. However, entropy stability of the simple outflow boundary requires $\boldsymbol{u}_K \cdot \boldsymbol{n} \geq 0$ (no back-propagation on outflow boundaries), which in general can not be verified a priori. Since this simple outflow boundary condition does not cause any numerical issues in our examples below, we didn't further investigate alternative outflow boundary treatments. We refer to [1] for more stable inflow/outflow boundary treatments based on characteristic decomposition.

Example 4.1: Accuracy Test in 1D

We start with an accuracy test to demonstrate the high order accuracy of our schemes for a smooth solution of the SWEs. Following the setup in [44], we take the following bottom topography and initial conditions:

$$b(x) = \sin^2(\pi x), \quad h(x, 0) = 5 + e^{\cos(2\pi x)}, \quad hu(x, 0) = \sin(\cos(2\pi x)).$$

The computation domain is a periodic unit interval $[0, 1]$, and final time is $t = 0.1$ where the solution is still smooth. We apply the plain Algorithm 4 without limiter, and compute the L^2-errors of water height h_h, velocity u_h, and discharge m_h on a sequence of uniform meshes with $N = 50 \times 2^l$ cells for $l = 0, 1, 2, 3, 4$. We take the solution on $N = 25 \times 2^5 = 1600$ cells as the *reference* solution when computing these L^2-errors. The results are recorded in Table 1. We clearly observe the expected third order convergence, and the error magnitude are comparable to the results reported in [44] for another third order entropy stable DG scheme.

Example 4.2: The Well-Balanced Test in 1D

In this example, we test the well-balanced property of our proposed methods to ensure that the still-water steady state is exactly preserved. We consider two different choices of the

Table 1 L^2 errors and convergence rate at time $t = 0.1$ for Example 4.1

N	L^2-err in h_h	rate	L^2-err in u_h	rate	L^2-err in m_h	rate
50	2.997e−04	–	3.583e−04	–	2.577e−03	–
100	2.730e−05	3.46	3.273e−05	3.45	2.352e−04	3.45
200	2.949e−06	3.21	3.538e−06	3.21	2.542e−05	3.21
400	3.600e−07	3.03	4.323e−07	3.03	3.103e−06	3.03
800	4.408e−08	3.03	5.296e−08	3.03	3.798e−07	3.03

Table 2 L^2 errors at time $t = 0.5$ for Example 4.2

N	smooth bot.			disc. bot.		
	L^2-err in h_h	L^2-err in u_h	L^2-err in m_h	L^2-err in h_h	L^2-err in u_h	L^2-err in m_h
100	9.819e−14	7.380e−14	5.025e−13	7.700e−14	7.627e−14	5.024e−13
200	1.747e−13	8.344e−14	6.331e−13	1.805e−13	1.042e−13	8.494e−13
400	3.740e−13	1.328e−13	1.029e−12	2.787e−13	2.153e−13	1.355e−12

bottom topography as used in [44]: a smooth bottom with

$$b(x) = 5\exp\left(-0.4(x-5)^2\right),$$

and a discontinuous bottom with

$$b(x) = \begin{cases} 4, & \text{if } 4 \le x \le 8, \\ 0, & \text{otherwise.} \end{cases}$$

The computational domain is [0, 10] with wall boundary conditions. The initial condition is taken as the stationary state

$$h + b = 10, u = 0.$$

We solve the problem until time $t = 0.5$ on three meshes with 100, 2000, and 400 uniform cells, and record the L^2-errors in Table 2. We observe all errors are at the level of round-off errors, which verifies the well-balanced property.

Example 4.3: A Small Perturbation Test in 1D

We test the following quasi-stationary test case proposed by LeVeque [30], which is a small perturbation of the steady state solution. The computational domain is [0, 2], and the bottom topography b(x) is given by

$$b(x) = \begin{cases} \frac{1}{4}(\cos(10\pi(x-1.5)) + 1), & \text{if } 1.4 \le x \le 1.6, \\ 0, & \text{otherwise.} \end{cases}$$

The initial conditions are

$$h(x,0) = \begin{cases} 1 - b(x) + \epsilon, & \text{if } 1.1 \le x \le 1.2, \\ 1 - b(x), & \text{otherwise.} \end{cases}, \quad u(x,0) = 0,$$

where ϵ is a given constant representing the size of the perturbation. Following [30], we consider a case with a big pulse ($\epsilon = 0.2$) and case with a small pulse ($\epsilon = 0.001$). The

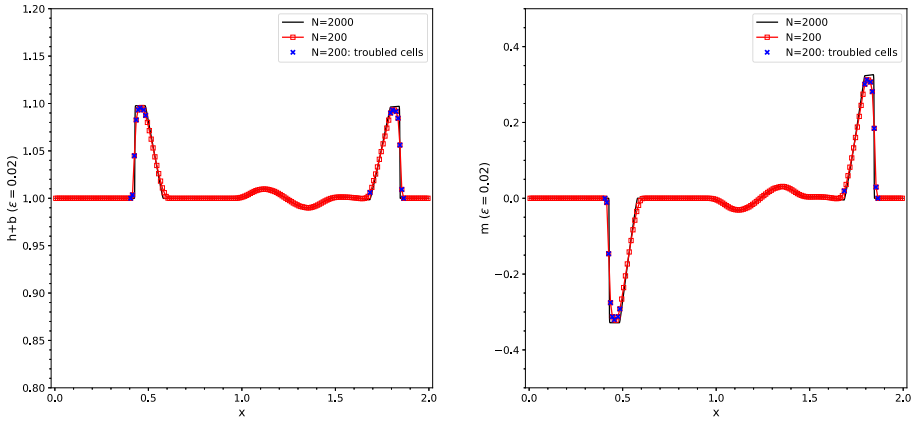

Fig. 1 Example 4.3 with with a big pulse $\epsilon = 0.2$ at time $t = 0.2$. Left: the water surface $h + b$; right: the discharge m. Blue squares indicate cells where TVB limiter are used at the final time

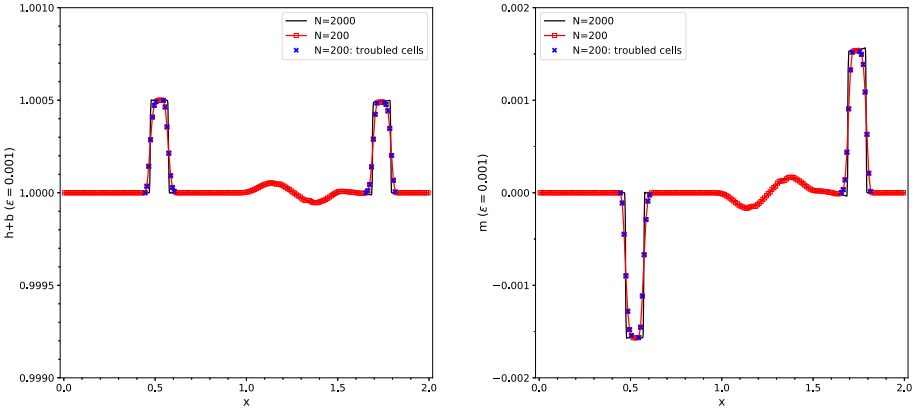

Fig. 2 Example 4.3 with with a small pulse $\epsilon = 0.001$ at time $t = 0.2$. Left: the water surface $h + b$; right: the discharge m. Blue squares indicate cells where TVB limiter are used at the final time

final time of simulation is $t = 0.2$. We compare our scheme on a uniform coarse mesh with $N = 200$ cells and a uniform fine mesh with $N = 2000$ cells. The results at final time for the total water surface $h + b$ and discharge $m = hu$ for the big pulse case are shown in Fig. 1, and those for the small pulse case are shown in Fig. 2. In these figures, blue squares indicate the troubled cells identified by our indicator (21). We observe good agreement of the results on coarse and fine meshes without spurious numerical oscillations, which also agrees well with results in the literature. Moreover, we observe that the indicator (21) with $tol = 0.02$ only activates cells close to the moving shocks for both case, with slightly more cells being identified as troubled cells for the small perturbation test.

Example 4.4: A Dam Breaking Problem over a Bump in 1D

We consider a one-dimensional dam breaking problem over a rectangular bump. It involves a rapidly varying flow over a discontinuous bottom topography. Following [44], we take the

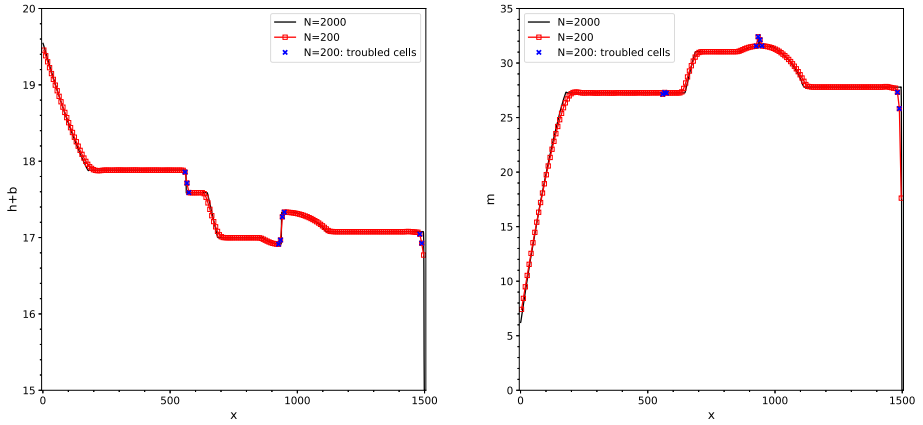

Fig. 3 Example 4.4 at time $t = 60$. Left: the water surface $h + b$; right: the discharge m. Blue squares indicate cells where TVB limiter are used at the final time

computational domain as [0, 1500], and use the following discontinuous bottom topography:

$$b(x) = \begin{cases} 8, & \text{if } |x - 750| < 1800/8, \\ 0, & \text{otherwise.} \end{cases}$$

We use outflow boundary conditions, and record the results at time $t = 60$ in Fig. 3, again using a uniform coarse mesh with $N = 200$ cells, and a uniform fine mesh with $N = 2000$ cells. We observe good agreement of the results on two meshes, which also agrees well with results reported in the literature. We note that the discharge has a small kink near $x = 937.5$, where the discontinuous of bottom topography happens. Also, the indicator is successful in identifying solution discontinuities.

Example 4.5: Entropy Glitch Test in 1D

We consider the Riemann problem with a flat bottom considered in [46]. The computational domain is $[-1, 1]$, the bottom topography $b(x) = 0$, and initial condition is

$$h(x, 0) = \begin{cases} 1, & \text{if } x < 0, \\ 0.1, & \text{otherwise.} \end{cases} \qquad u(x, 0) = 0.$$

The gravitational constant is taken to be $g = 10$, and final time of simulation is $t = 0.2$.

It was shown in [46, Fig. 8] that standard DG method with a local Lax-Friedrichs numerical flux develops an unphysical discontinuity, called an "entropy glitch", at $x = 0$, while the entropy stable DG method is able to capture the solution well on the coarse mesh. The results in a uniform mesh with 200 cells are shown in Fig. 4.

Example 4.6: A Dam Break Problem with a Dry Bed in 1D

We consider a Riemann Problem with a constant bottom used in [52]. Here a dried river bed is used to examine the performance of our scheme in case of moving wet/dry interface. The

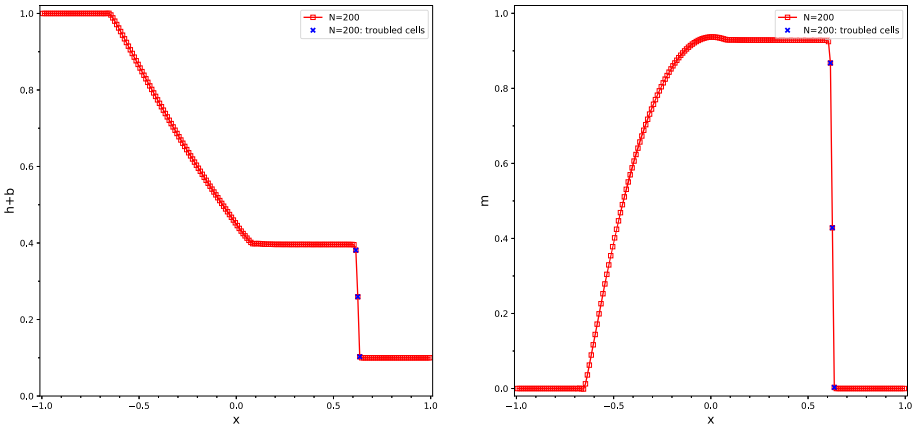

Fig. 4 Example 4.5 at time $t = 0.2$. Left: the water surface $h+b$; right: the discharge m. Blue squares indicate cells where TVB limiter are used at the final time

computation domain is taken to be $[-300, 300]$, and initial condition is

$$h(x,0) = \begin{cases} 10, & \text{if } x < 0, \\ 10^{-12}, & \text{otherwise.} \end{cases} \qquad u(x,0) = 0.$$

We use a tiny positive value 10^{-12} to indicate the dry bed to avoid division by zero in the velocity computation. This is a very challenging problem as our default algorithm without dry cell limiter or velocity limiter fails after a couple of time steps due to an excessive large velocity approximation. Here we activate the dry cell limiter (23) with $\epsilon_d = 5 \times 10^{-3}$, and do not apply the velocity limiter. We obtain the dry cell threshold constant $\epsilon_d = 5 \times 10^{-3}$ by test-and-trial, where the solution quality was observed to be quite sensitive to this value. Taking ϵ_d too large, say 0.05, leads to loss of accuracy as piecewise constant approximations are used in dry cells. On the other hand, taking ϵ_d too small, say 5×10^{-4}, leads to instability caused by large velocity approximations, where additional velocity limiter would be needed for a stable simulation. In our experience, taking $\epsilon_d = 5 \times 10^{-3}$ is a good comprise between stability and accuracy without using the velocity limiter.

The exact solution to the above Riemann problem (with dry bed $h = 0$ for $x > 0$) was given in [43, Section 6.3.1], which we cite below:

$$[h(x,t), u(x,t)] = \begin{cases} [h_L, 0] & \text{if } x/t \le -a_L, \\ \left[\frac{(2a_L - x/t)^2}{9g}, \frac{2}{3}(a_L + x/t)\right] & \text{if } -a_L \le x/t \le 2a_L, \\ [0, 0] & \text{if } 2a_L \le x/t, \end{cases}$$

where $h_L = 10$ and $a_L = \sqrt{h_L g}$ is the left state celerity. The jump in velocity at the dry-wet interface $x/t = 2a_L$ makes the accurate numerical computation of this interface very challenging. In Fig. 5 we present the results of our algorithm at times $T = 4$, $T = 8$, and $T = 12$ on a uniform mesh with $N = 200$ cells, along with the exact solution. We observe that the numerical results are in general in good agreements with the exact solution, but the location of the moving wet/dry interface was not accurately captured. Similar results were also observed for the DG scheme in [52, Example 6.3]. In Fig. 6 we plot the zoomed-in version of the solution near the wet/dry front at these times for our DG scheme on uniform meshes with $N = 200$ and $N = 800$ cells. It is clear that increasing mesh resolution improves the accuracy near the interface.

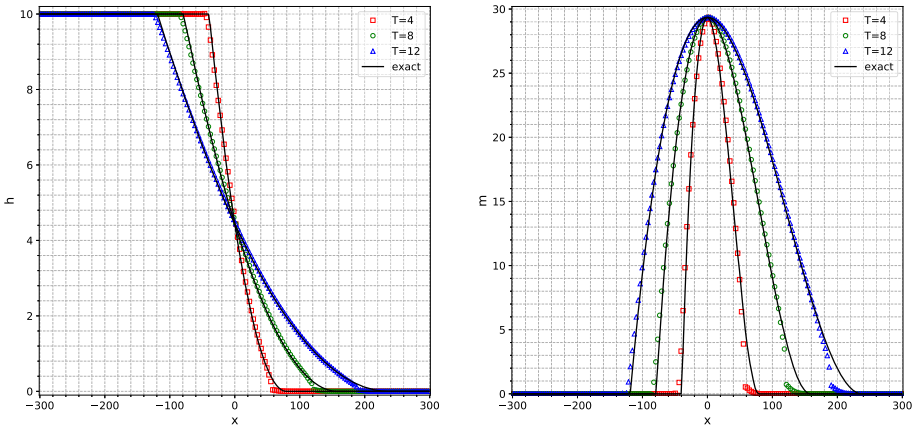

Fig. 5 Example 4.6 at times $T = 4$, $T = 8$, and $T = 12$. Left: the water surface h; right: the discharge m. Uniform mesh with $N = 200$ cells

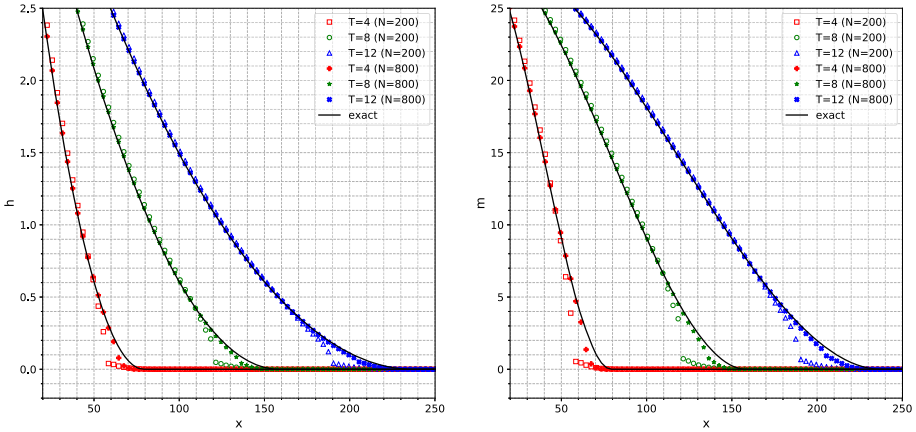

Fig. 6 Example 4.6 at times $T = 4$, $T = 8$, and $T = 12$. Zoomed-in of the wet/dry front. Left: the water surface h; right: the discharge m

Finally, we show in Fig. 7 the time evolution of the interface $h = 0.05$ and $h = 5 \times 10^{-5}$ for the scheme with $N = 200$ and $N = 800$, along with the exact interface. The general trend of a right-going moving interface is observed, and the results for $N = 800$ is quite good, especially in approximating the interface $h = 5 \times 10^{-5}$. We finally note that (i) the interface $h = 0.05$ split the computational domain into a wet region on the left where high-order DG approximations are used and a dry region on the right where piecewise constant approximations are used, (ii) and the interface $h = 5 \times 10^{-5}$ can be viewed as the numerical approximation of the actual wet/dry interface.

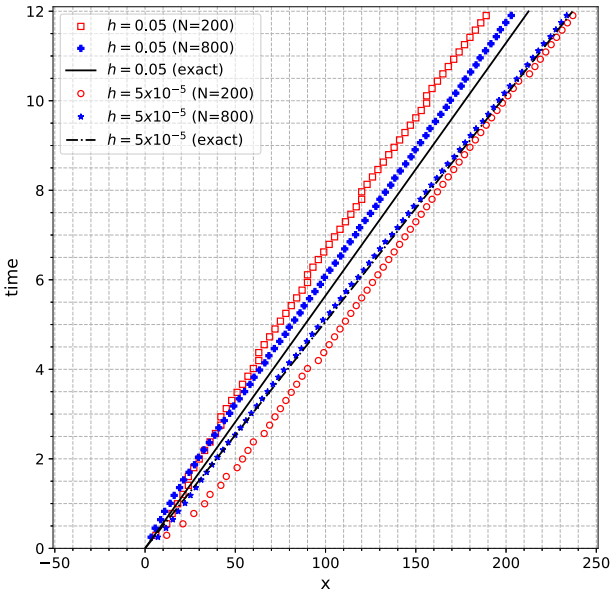

Fig. 7 Example 4.6. Time evolution of the interface $h = 0.05$ and $h = 5 \times 10^{-5}$

Example 4.7: Accuracy Tests in 2D

Now we turn to the performance of our scheme on 2D triangular meshes. We consider the following three test cases to demonstrate the high order spatial/temporal accuracy of the proposed schemes.

Example 4.7(a): Smooth Solution Evolution

This is a 2D version of the 1D accuracy test considered in Example 4.1 The domain is a periodic unit square $[0, 1] \times [0, 1]$. The bottom topography and the initial conditions are given as follows:

$$\begin{aligned} b(x, y) &= \sin(2\pi x) + \sin(2\pi y), \\ h(x, y, 0) &= 10 + e^{\sin(2\pi x)} \cos(2\pi y), \\ hu(x, y, 0) &= \sin(\cos(2\pi x)) \sin(2\pi y), \\ hv(x, y, 0) &= \cos(2\pi x) \cos(\sin(2\pi y)). \end{aligned}$$

We take the final time $t = 0.05$ where the solution is still smooth. We apply the plain Algorithm 4 without limiter and compute the L^2-errors of water height h_h, velocity $\boldsymbol{u}_h$, and discharge $\boldsymbol{m}_h$ on a sequence of uniform structured triangular meshes with $N \times N \times 2$ cells where $N = 25 \times 2^l$ cells for $l = 0, 1, 2, 3$. The time step size is taken to be $\Delta t = 0.005/N$, which is within the CFL limit. We take the numerical solution (with polynomial degree $k = 2$) on a fine mesh with $400 \times 400 \times 2$ cells as the *reference* solution to compute the errors. The results are recorded in Table 3. We observe the expected third order convergence for the water height, and nearly third order convergence for the velocity and discharge.

Table 3 L^2 errors and convergence rate at time $t = 0.05$ for Example 4.7(a). Polynomial degree $k = 2$

N	L^2-err in h_h	rate	L^2-err in u_h	rate	L^2-err in m_h	rate
25	1.420e−03	–	1.670e−03	–	1.379e−02	–
50	1.567e−04	3.18	2.083e−04	3.00	1.683e−03	3.03
100	1.917e−05	3.03	2.820e−05	2.89	2.371e−04	2.83
200	2.363e−06	3.02	3.880e−06	2.86	3.403e−05	2.80

Table 4 L^2 errors and convergence rate at time $t = 0.004$ for Example 4.7(b). Polynomial degree $k = 2$

N	L^2-err in h_h	rate	L^2-err in u_h	rate	L^2-err in m_h	rate
25	1.543e−04	–	1.387e−04	–	1.423e−03	–
50	1.840e−05	3.07	1.681e−05	3.05	1.703e−04	3.06
100	2.257e−06	3.03	2.057e−06	3.03	2.071e−05	3.04
200	2.805e−07	3.01	2.564e−07	3.00	2.582e−06	3.00

Table 5 L^2 errors and convergence rate at time $t = 0.004$ for Example 4.7(b). Polynomial degree $k = 3$

N	L^2-err in h_h	rate	L^2-err in u_h	rate	L^2-err in m_h	rate
25	3.704e−06	–	6.752e−06	–	6.587e−05	–
50	1.343e−07	4.79	4.155e−07	4.02	4.065e−06	4.02
100	5.853e−09	4.52	2.518e−08	4.04	2.495e−07	4.03
200	2.958e−10	4.31	1.546e−09	4.03	1.537e−08	4.02

Example 4.7(b): Method of Manufactured Solutions

We use the same setup as in Example 4.7(a), except that additional source terms are added to the SWEs so that the exact solution is give as follows:

$$\begin{aligned} h(x, y, t) &= 10 + e^{\sin(2\pi x)} \cos(2\pi y) \cos(2\pi t), \\ hu(x, y, t) &= \sin(\cos(2\pi x)) \sin(2\pi y) \cos(2\pi t), \\ hv(x, y, t) &= \cos(2\pi x) \cos(\sin(2\pi y)) \cos(2\pi t). \end{aligned}$$

Here we take the final time to be $t = 0.004$ so that temporal error is negligible compared to spatial error. We record results in Table 4 for polynomial degree $k = 2$, and in Table 5 for $k = 3$, from which we clearly observed optimal $(k + 1)$-th order of convergence.

Example 4.7(c): Stationary Vortex

Here we numerically verify the high-order spatial accuracy for a stationary vortex problem. Following [4, Section 3.5], we consider a steady radial symmetric water height and a rotating incompressible velocity field:

$$h(x, y, t) = h(r), \quad u(x, y, t) = -f(r)y, \quad v(x, y, t) = f(r)x, \quad \text{where } r = \sqrt{x^2 + y^2}. \tag{27}$$

Table 6 L^2 errors and convergence rate at time $t = 0.1$ for Example 4.7(c)

k	N	L^2-err in h_h	rate	L^2-err in u_h	rate	L^2-err in m_h	rate
	25	4.582e−01	–	7.858e−01	–	1.271e+00	–
0	50	2.428e−01	0.92	4.115e−01	0.93	6.647e−01	0.93
	100	1.245e−01	0.96	2.101e−01	0.97	3.411e−01	0.96
	200	6.300e−02	0.98	1.061e−01	0.99	1.728e−01	0.98
	25	1.023e−01	–	1.520e−01	–	2.716e−01	–
1	50	3.214e−02	1.67	4.509e−02	1.75	8.786e−02	1.63
	100	8.364e−03	1.94	1.192e−02	1.92	2.261e−02	1.96
	200	2.087e−03	2.00	2.979e−03	2.00	5.647e−03	2.00
	25	2.078e−02	–	2.763e−02	–	5.572e−02	–
2	50	3.079e−03	2.75	4.493e−03	2.62	8.014e−03	2.80
	100	4.164e−04	2.89	7.105e−04	2.66	1.163e−03	2.78
	200	5.890e−05	2.82	1.045e−04	2.77	1.654e−04	2.81
	25	2.561e−03	–	3.168e−03	–	5.631e−03	–
3	50	1.393e−04	4.20	2.425e−04	3.71	3.969e−04	3.83
	100	8.888e−06	3.97	1.675e−05	3.86	2.503e−05	3.99
	200	5.615e−07	3.98	1.065e−06	3.98	1.562e−06	4.00
	25	5.147e−04	–	5.191e−04	–	2.112e−03	–
4	50	1.736e−05	4.89	2.101e−05	4.63	6.606e−05	5.00
	100	5.633e−07	4.95	8.014e−07	4.71	2.207e−06	4.90
	200	1.798e−08	4.97	2.858e−08	4.81	7.235e−08	4.93

We take a flat bottom $b(x, y) = 0$, with gravitational constant $g = 1$. Elementary calculation shows that the steady solution (27) solves the SWEs (1) only if

$$h'(r) = \frac{f(r)^2 r}{g}.$$

In this case, the gravitational term $g\nabla h$ in the momentum equation is in perfect balance with the inertial term $\boldsymbol{u} \cdot \nabla \boldsymbol{u}$. Following [4], we take

$$f(r) = \exp(-0.5(r^2 - 1)), \text{ and } h(r) = 2 - \frac{1}{2g}\exp(-(r^2 - 1)),$$

in the numerical simulation. The computational domain is taken to be a large periodic square $\Omega = [-10, 10] \times [-10, 10]$ so that boundary effects are negligible. We check the spatial accuracy of our scheme with polynomial degree from $k = 0$ to $k = 4$ on a sequence of uniform structured triangular meshes with $N \times N \times 2$ cells where $N = 25 \times 2^l$ for $l = 0, 1, 2, 3$. The time step size is taken to be $\Delta t = 0.5/N$ for $k \leq 3$ and $\Delta t = 0.25/N$ for $k = 4$. We take final time to be $t = 0.1$, and record convergence results in Table 6, from which we observe $(k + 1)$-th order of convergence for most of the cases.

Example 4.8: A Small Perturbation Test in 2D

We test the following 2D quasi-stationary test case proposed by LeVeque [30], which is a small perturbation of the steady state solution. The computation domain is $[0, 2] \times [0, 1]$.

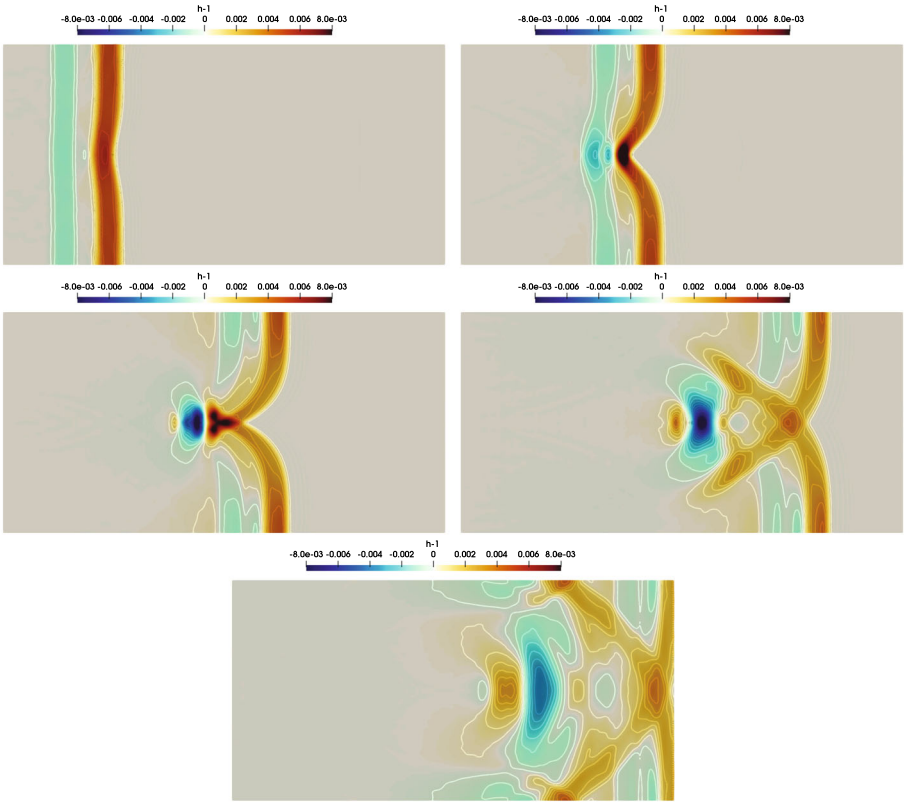

Fig. 8 Example 4.8. Contour plot for $h + b - 1$. 30 uniform contour lines from $h + b - 1 = -0.008$ to $h + b - 1 = 0.008$. From left to right, top to bottom: $t = 0.12, 0.24, 0.36, 0.48, 0.6$

The bottom topography consists of an elliptical shaped hump

$$b(x, y) = 0.8 \exp(-5(x - 0.9)^2 - 50(x - 0.5)^2),$$

and the initial conditions are

$$h(x, y, 0) = \begin{cases} 1 - b(x, y) + 0.01, & \text{if } 0.05 \le x \le 0.15, \\ 1 - b(x, y), & \text{otherwise.} \end{cases}, \quad u(x, y, 0) = v(x, y, 0) = 0.$$

Outflow boundary conditions are imposed on the left and right boundary while wall boundary condition (symmetry) are imposed on the top and bottom boundaries. Due to symmetry, we perform the calculation on half of the domain $\Omega = [0, 2] \times [0, 0.5]$ with symmetry boundary condition on the top boundary $y = 0.5$. We consider our scheme on an unstructured triangular mesh with mesh size $\tau_K = 0.01$. The water surface contour at times $t = 0.12, 0.24, 0.36, 0.48, 0.60$ are recorded in Fig. 8. Our scheme produce non oscillatory solutions and the results agrees well with those in the literature.

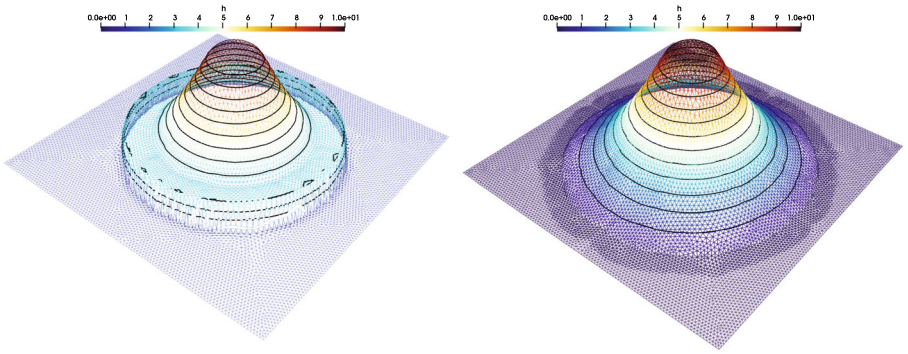

Fig. 9 Example 4.9. Contour and surface plots of water height for the circular dam-break problem at $t = 0.69$. Left: wet bed. 11 uniform contour lines from 2 to 9.4; Right dry bed. 12 uniform contour lines from 0.01 to 8.9

Example 4.9: Circular Dam Break Test in 2D

We consider the circular dam break problem used in [36, Sect. 3.2.1]. The space domain is a 50×50 square with a cylindrical dam with radius $r = 11$ and centred in the square. The initial water height is 10 inside the dam, and is either 1 outside the dam (a wet bed), or 10^{-12} outside the dam (a dry bed). The final time of the simulation is $t = 0.69$. The bottom topography is set to be zero. Here the dry bed case need special care, where we applied the dry cell limiter (23) with $\epsilon_d = 5 \times 10^{-3}$, and activated the velocity limiter (24)–(25) with $V_{\max} = 15$. These treatment were not used for the wet bed case. Due to symmetry of the problem, we perform the computation only on a quarter of the domain with symmetric boundary conditions. We take an unstructured triangular mesh with mesh size $\tau_K = 0.5$. The results of two cases are presented in Fig. 9. We note that taking $\epsilon_d = 5 \times 10^{-3}$ for the dry bed case in the right of Fig. 9 is via test and trial. In particular, visible numerical oscillations near the wet/dry interface are observed (not reported here) if we reduce ϵ_d to 10^{-3} in the simulation.

Example 4.10: Dam Break on a Closed Channel in 2D

We consider the problem [23, Sect. 8.9] which model dam break on a closed channel. The domain represents a channel of 75 length and 30 width with three mounds will wall boundary conditions. The shape of the mounds is defined by the function $b(x, y) = \max(0, m_1, m_2, m_3)$, where

$$
\begin{aligned}
m_1 &= 1 - 0.1\sqrt{(x-30)^2 + (y-22.5)^2}, \\
m_2 &= 1 - 0.1\sqrt{(x-30)^2 + (y-7.5)^2}, \\
m_3 &= 2.8 - 0.28\sqrt{(x-47.5)^2 + (y-15)^2}.
\end{aligned}
$$

The initial conditions are

$$
h(x, y, 0) = \begin{cases} 1.875, & \text{if } x < 16 \\ 10^{-12}, & \text{otherwise.} \end{cases} \qquad u(x, y, 0) = v(x, y, 0) = 0
$$

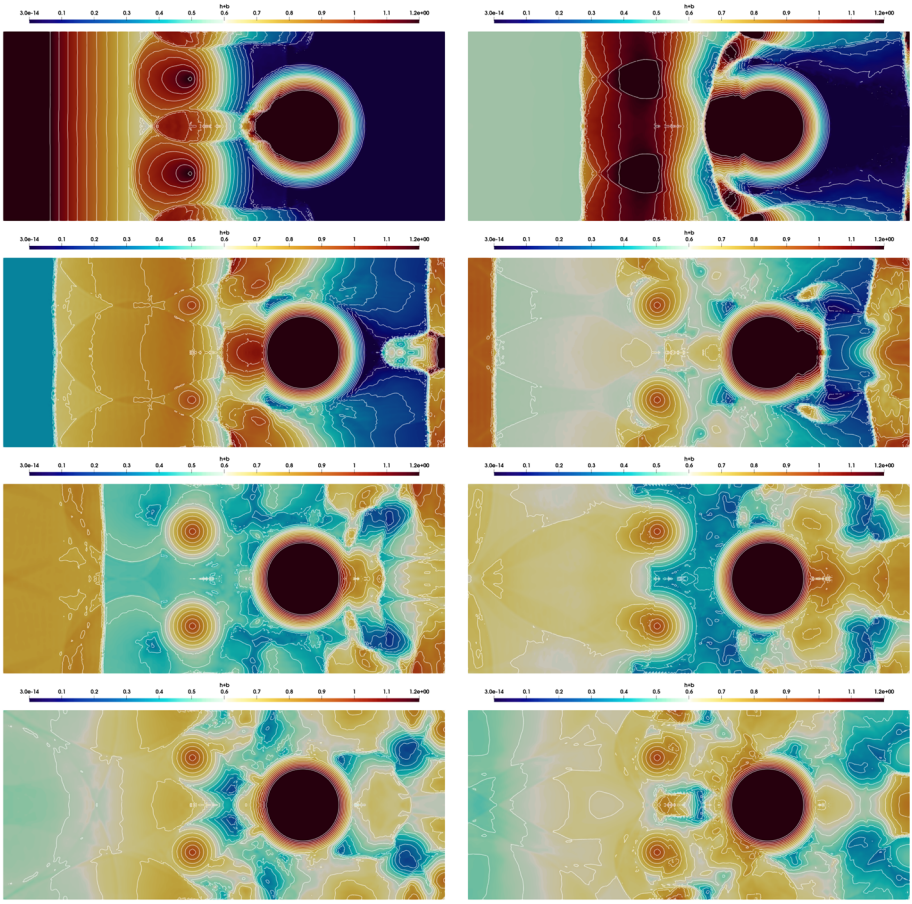

Fig. 10 Example 4.10. Contour and surface plots of water surface for the dam-break problem on a closed channel. 20 uniform contour lines from 0 to 1.2. Left to right, top to bottom: $t = 5, 10, 15, 20, 25, 30, 35, 40$

Due to the moving wet/dry interface, we activate the dry cell limiter with $\epsilon_d = 10^{-3}$, and the velocity limiter (24)–(25) with $V_{\max} = 9$. Due to symmetry, we only perform the calculation on half of the domain $[0, 75] \times [0, 15]$, and apply symmetry boundary conditions on all the boundaries. Contour plots of the water surface for the simulation results on an unstructured triangular mesh with mesh size $\tau_K = 0.5$ are shown in Fig. 10 for various times. We observe complex flow structures for this problem, and our scheme produces satisfactory results compared with those from [23, Fig. 15]. We note that, unlike the previous example, here further reducing ϵ_d from 10^{-3} to 10^{-6} does not significantly alter the quality of the numerical solution.

5 Conclusion

We proposed a novel velocity-based DG scheme for the SWEs. Our semidiscrete DG scheme is locally conservative, entropy stable, and well-balanced. We then apply the SSP-RK3

time stepping for the time discretzation, and obtained an explicit locally conservative, well-balanced, and positivity-preserving fully discrete scheme in Algorithm 5, where the treatment of strong shocks via a characteristic-wise TVB limiter and proper wetting/drying treatment near dry cells was also discussed. Both one- and two-dimensional numerical results are provided to demonstrate the performance of the proposed DG method. Our entropy stable scheme is particularly simple and competitive compared with existing entropy stable DG schemes for SWEs in the literature.

The velocity-based DG scheme can be used to construct robust entropy/energy stable DG schemes for other compressible flow problems, which will be carried out in our future studies.

Acknowledgements The author would like to thank Yulong Xing from Ohio State University for fruitful discussions on the topic.

Funding We acknowledge the partial support of this work from U.S. National Science Foundation through grant DMS-2012031.

Data Availability Enquiries about data availability should be directed to the authors.

Declarations

Competing Interests The authors have not disclosed any competing interests.

References

1. Atkins, H.: Continued development of the discontinuous Galerkin method for computational aeroacoustic applications. In: 3rd AIAA/CEAS Aeroacoustics Conference (1997)
2. Audusse, E., Bouchut, F., Bristeau, M.-O., Klein, R., Perthame, B.: A fast and stable well-balanced scheme with hydrostatic reconstruction for shallow water flows. SIAM J. Sci. Comput. **25**, 2050–2065 (2004)
3. Bonev, B., Hesthaven, J.S., Giraldo, F.X., Kopera, M.A.: Discontinuous Galerkin scheme for the spherical shallow water equations with applications to tsunami modeling and prediction. J. Comput. Phys. **362**, 425–448 (2018)
4. Boscheri, W., Dumbser, M., Righetti, M.: A semi-implicit scheme for 3D free surface flows with high-order velocity reconstruction on unstructured Voronoi meshes. Int. J. Numer. Meth. Fluids **72**, 607–631 (2013)
5. Brus, S.R., Wirasaet, D., Kubatko, E.J., Westerink, J.J., Dawson, C.: High-order discontinuous Galerkin methods for coastal hydrodynamics applications. Comput. Methods Appl. Mech. Engrg. **355**, 860–899 (2019)
6. Bunya, S., Kubatko, E.J., Westerink, J.J., Dawson, C.: A wetting and drying treatment for the Runge-Kutta discontinuous Galerkin solution to the shallow water equations. Comput. Methods Appl. Mech. Engrg. **198**, 1548–1562 (2009)
7. Busto, S., Dumbser, M., Escalante, C., Favrie, N., Gavrilyuk, S.: On high order ADER discontinuous Galerkin schemes for first order hyperbolic reformulations of nonlinear dispersive systems. J. Sci. Comput. **87**, 48 (2021)
8. Busto, S., Dumbser, M., Gavrilyuk, S., Ivanova, K.: On thermodynamically compatible finite volume methods and path-conservative ADER discontinuous Galerkin schemes for turbulent shallow water flows. J. Sci. Comput. **88**, 28 (2021)
9. Buttinger-Kreuzhuber, A., Horváth, Z., Noelle, S., Blüschl, G., Waser, J.: A fast second-order shallow water scheme on two-dimensional structured grids over abrupt topography. Advances in Water Resources **127**, 89–108 (2019)
10. Chan, J.: On discretely entropy conservative and entropy stable discontinuous Galerkin methods. J. Comput. Phys. **362**, 346–374 (2018)
11. Chen, G., Noelle, S.: A new hydrostatic reconstruction scheme based on subcell reconstructions. SIAM J. Numer. Anal. **55**, 758–784 (2017)

12. Chen, T., Shu, C.-W.: Review of entropy stable discontinuous Galerkin methods for systems of conservation laws on unstructured simplex meshes. CSIAM Transactions on Applied Mathematics **1**, 1–52 (2020)
13. Cockburn, B.: Discontinuous galerkin methods for computational fluid dynamics. In: Encyclopedia of Computational Mechanics Second Edition, pp. 1–63 (2018)
14. Cockburn, B., Hou, S., Shu, C.-W.: The Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws. IV. The multidimensional case. Math. Comp. **54**, 545–581 (1990)
15. Cockburn, B., Shu, C.-W.: The Runge-Kutta discontinuous Galerkin method for conservation laws. V. Multidimensional systems. J. Comput. Phys. **141**, 199–224 (1998)
16. Cockburn, B., Shu, C.-W.: Runge-Kutta discontinuous Galerkin methods for convection-dominated problems. J. Sci. Comput. **16**, 173–261 (2001)
17. Dafermos, C.M.: Hyperbolic conservation laws in continuum physics, vol. 325 of Grundlehren der mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], Springer-Verlag, Berlin, fourth ed. (2016)
18. Dawson, C., Kubatko, E.J., Westerink, J.J., Trahan, C., Mirabito, C., Michoski, C., Panda, N.: Discontinuous Galerkin methods for modeling hurricane storm surge. Advances in Water Resources **34**, 1165–1176 (2010)
19. Dubiner, M.: Spectral methods on triangles and other domains. J. Sci. Comput. **6**, 345–390 (1991)
20. Fjordholm, U.S., Mishra, S., Tadmor, E.: Energy preserving and energy stable schemes for the shallow water equations. In: Foundations of computational mathematics, Hong Kong 2008, vol. 363 of London Math. Soc. Lecture Note Ser., Cambridge Univ. Press, Cambridge, pp. 93–139 (2009)
21. Fjordholm, U.S., Mishra, S., Tadmor, E.: Well-balanced and energy stable schemes for the shallow water equations with discontinuous topography. J. Comput. Phys. **230**, 5587–5609 (2011)
22. Fu, G., Shu, C.-W.: A new troubled-cell indicator for discontinuous Galerkin methods for hyperbolic conservation laws. J. Comput. Phys. **347**, 305–327 (2017)
23. Gallardo, J.M., Parés, C., Castro, M.: On a well-balanced high-order finite volume scheme for shallow water equations with topography and dry areas. J. Comput. Phys. **227**, 574–601 (2007)
24. Gassner, G.J., Winters, A.R., Kopriva, D.A.: A well balanced and entropy conservative discontinuous Galerkin spectral element method for the shallow water equations. Appl. Math. Comput. **272**, 291–308 (2016)
25. Giraldo, F.X., Hesthaven, J.S., Warburton, T.: Nodal high-order discontinuous Galerkin methods for the spherical shallow water equations. J. Comput. Phys. **181**, 499–525 (2002)
26. Gottlieb, S., Shu, C.-W., Tadmor, E.: Strong stability-preserving high-order time discretization methods. SIAM Rev. **43**, 89–112 (2001)
27. Hughes, T.J.R., Franca, L.P., Mallet, M.: A new finite element formulation for computational fluid dynamics. I. Symmetric forms of the compressible Euler and Navier-Stokes equations and the second law of thermodynamics. Comput. Methods Appl. Mech. Engrg. **54**, 223–234 (1986)
28. Kurganov, A., Petrova, G.: A second-order well-balanced positivity preserving central-upwind scheme for the Saint-Venant system. Commun. Math. Sci. **5**, 133–160 (2007)
29. Lax, P., Wendroff, B.: Systems of conservation laws. Comm. Pure Appl. Math. **13**, 217–237 (1960)
30. LeVeque, R.J.: Balancing source terms and flux gradients in high-resolution Godunov methods: the quasi-steady wave-propagation algorithm. J. Comput. Phys. **146**, 346–365 (1998)
31. LeVeque, R.J.: Finite volume methods for hyperbolic problems. Cambridge Texts in Applied Mathematics, Cambridge University Press, Cambridge (2002)
32. Qiu, J., Shu, C.-W.: A comparison of troubled-cell indicators for Runge-Kutta discontinuous Galerkin methods using weighted essentially nonoscillatory limiters. SIAM J. Sci. Comput. **27**, 995–1013 (2005)
33. Rhebergen, S., Bokhove, O., van der Vegt, J.J.W.: Discontinuous Galerkin finite element methods for hyperbolic nonconservative partial differential equations. J. Comput. Phys. **227**, 1887–1922 (2008)
34. Schöberl, J.: C++11 Implementation of Finite Elements in NGSolve. ASC Report 30/2014, Institute for Analysis and Scientific Computing, Vienna University of Technology (2014)
35. Schwanenberg, D., Köngeter, J.: A discontinuous Galerkin method for the shallow water equations with source terms. In: Discontinuous Galerkin methods (Newport, RI, 1999), vol. 11 of Lect. Notes Comput. Sci. Eng., Springer, Berlin, pp. 419–424 (2000)
36. Seaïd, M.: Non-oscillatory relaxation methods for the shallow-water equations in one and two space dimensions. Int. J. Numer. Meth. Fluids **46**, 457–484 (2004)
37. Shu, C.-W., Osher, S.: Efficient implementation of essentially nonoscillatory shock-capturing schemes. J. Comput. Phys. **77**, 439–471 (1988)
38. Tadmor, E.: Entropy conservative finite element schemes. In: Numerical Methods for Compressible Flows - Finite Difference Element and Volume Techniques, T. Tezduyar and T. Hughes, eds., vol. AMD-Vol.

78 of Proceedings of the winter annual meeting of the American Society of Mechanical Engineering, pp. 149–158 (1986)
39. Tadmor, E.: The numerical viscosity of entropy stable schemes for systems of conservation laws. I. Math. Comp. **49**, 91–103 (1987)
40. Tadmor, E.: Entropy stable schemes, in Handbook of numerical methods for hyperbolic problems, vol. 17 of Handb. Numer. Anal., Elsevier/North-Holland, Amsterdam, pp. 467–493 (2016)
41. Tadmor, E., Zhong, W.: Energy-preserving and stable approximations for the two-dimensional shallow water equations. In: Mathematics and computation, a contemporary view, vol. 3 of Abel Symp., Springer, Berlin, pp. 67–94 (2008)
42. Tavelli, M., Dumbser, M.: A high order semi-implicit discontinuous Galerkin method for the two dimensional shallow water equations on staggered unstructured meshes. Appl. Math. Comput. **234**, 623–644 (2014)
43. Toro, E.: Shock-Capturing Methods for Free-Surface Shallow Flows. Wiley (2001)
44. Wen, X., Don, W.S., Gao, Z., Xing, Y.: Entropy stable and well-balanced discontinuous Galerkin methods for the nonlinear shallow water equations. J. Sci. Comput. **83**, 66 (2020)
45. Wintermeyer, N., Winters, A.R., Gassner, G.J., Kopriva, D.A.: An entropy stable nodal discontinuous Galerkin method for the two dimensional shallow water equations on unstructured curvilinear meshes with discontinuous bathymetry. J. Comput. Phys. **340**, 200–242 (2017)
46. Wintermeyer, N., Winters, A.R., Gassner, G.J., Warburton, T.: An entropy stable discontinuous Galerkin method for the shallow water equations on curvilinear meshes with wet/dry fronts accelerated by GPUs. J. Comput. Phys. **375**, 447–480 (2018)
47. Wu, X., Kubatko, E.J., Chan, J.: High-order entropy stable discontinuous Galerkin methods for the shallow water equations: curved triangular meshes and GPU acceleration. Comput. Math. Appl. **82**, 179–199 (2021)
48. Xing, Y.: Numerical methods for the nonlinear shallow water equations. In: Handbook of numerical methods for hyperbolic problems, vol. 18 of Handb. Numer. Anal., Elsevier/North-Holland, Amsterdam, pp. 361–384 (2017)
49. Xing, Y., Shu, C.-W.: High order well-balanced finite volume WENO schemes and discontinuous Galerkin methods for a class of hyperbolic systems with source terms. J. Comput. Phys. **214**, 567–598 (2006)
50. Xing, Y., Shu, C.-W.: A survey of high order schemes for the shallow water equations. J. Math. Study **47**, 221–249 (2014)
51. Xing, Y., Zhang, X.: Positivity-preserving well-balanced discontinuous Galerkin methods for the shallow water equations on unstructured triangular meshes. J. Sci. Comput. **57**, 19–41 (2013)
52. Xing, Y., Zhang, X., Shu, C.-W.: Positivity-preserving high order well-balanced discontinuous galerkin methods for the shallow water equations. Advances in Water Resources **33**, 1476–1493 (2010)
53. Xu, Z., Zhang, X.: Bound-preserving high-order schemes. In: Handbook of numerical methods for hyperbolic problems, vol. 18 of Handb. Numer. Anal., Elsevier/North-Holland, Amsterdam, pp. 81–102 (2017)
54. Zhang, X., Shu, C.-W.: Maximum-principle-satisfying and positivity-preserving high-order schemes for conservation laws: survey and new developments. Proc. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci. **467**, 2752–2776 (2011)