Comput. Methods Appl. Mech. Engrg. 270 (2014) 113–149

Contents lists available at ScienceDirect

Comput. Methods Appl. Mech. Engrg.

journal homepage: www.elsevier.com/locate/cma

Discontinuous Galerkin methods with nodal and hybrid modal/nodal triangular, quadrilateral, and polygonal elements for nonlinear shallow water flow

D. Wirasaet [a,*], E.J. Kubatko [b], C.E. Michoski [c], S. Tanaka [a,d], J.J. Westerink [a], C. Dawson [c]

[a] Environmental Fluid Dynamics Laboratories, Department of Civil and Environmental Engineering and Earth Sciences, University of Notre Dame, Notre Dame, IN 46556, USA
[b] Department of Civil and Environmental Engineering and Geodetic Science, The Ohio State University, Columbus, OH 43210, USA
[c] Institute for Computational Engineering and Sciences, The University of Texas at Austin, Austin, TX 78712, USA
[d] Earthquake Research Institute, The University of Tokyo, Bunkyo-ku, Tokyo 113-0032, Japan

ARTICLE INFO

Article history:
Received 15 December 2011
Received in revised form 26 September 2013
Accepted 3 November 2013
Available online 23 November 2013

Keywords:
Discontinuous Galerkin finite elements
Nodal
Modal
Computational cost
Well-balanced
Shallow water equations

ABSTRACT

We present a comprehensive assessment of nodal and hybrid modal/nodal discontinuous Galerkin (DG) finite element solutions on a range of unstructured meshes to nonlinear shallow water flow with smooth solutions. The nodal DG methods on triangles and a tensor-product nodal basis on quadrilaterals are considered. The hybrid modal/nodal DG methods utilize two different synergistic polynomial bases on polygons in realizing the DG discretization; orthogonal basis functions constructed by the Gram–Schmidt process are used as trial and test functions in a DG weak formulation; and a nodal basis is used as an efficient means for area integration. These are implemented on triangular, quadrilateral, and polygonal elements. In addition, we discuss aspects to be considered in order to achieve the so-called well-balanced property that preserves steady state at rest with a spatially varying bed. The performance in terms of accuracy and computational cost is demonstrated using h and p convergence studies on a nonlinear problem with a manufactured solution and the nonlinear Stommel problem with flat and non-flat beds. To assess the performance of quadrilateral and polygonal elements in comparison to triangular elements, we consider a setting in which a quadrilateral mesh, a mixed triangular–quadrilateral mesh, and polygonal mesh are derived from a given triangular mesh and *vice versa*. The tests conducted reveal the merit of using the quadrilateral elements in terms of computational cost per accuracy and computing time. More importantly, the numerical results clearly show that high order schemes significantly improve the cost performance for a given level of accuracy, with cubic or bi-cubic interpolants particularly achieving dramatic improvements in accuracy as compared to linear and quadratic interpolants, with diminishing benefit as $p > 3$.

1. Introduction

The shallow water equations (SWE) are used extensively in modeling many important physical phenomena, such as hurricane induced flooding, tides, riverine flows, tsunami waves, dam breaks, and many others. The equations can be coupled with a range of transport equations to model problems such as salinity, heat, and contaminant movement. Simulations of

* Corresponding author.
E-mail address: dwirasae@nd.edu (D. Wirasaet).

0045-7825/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.cma.2013.11.006

such environmental flow problems frequently involve large, geometrically complicated domains and integration over long periods of times. An accurate and efficient solution of the SWE is therefore crucial in numerical simulations. While relatively young in comparison with more conventional approaches, discontinuous Galerkin (DG) finite element methods (see [1,2] for reviews of DG methods) have increasingly become a powerful alternative for solving the SWE [3–12]. Conceptually similar to finite volume methods, DG methods inherently have the property of being conservative on the element level, making them ideal for coupling flow and transport models. Additional notable advantages of DG methods include the ease of constructing high order schemes on unstructured meshes and high scalability for parallel implementation when used in conjunction with explicit time integration schemes. Since DG methods use a discontinuous approximation, they are able to accommodate non-conforming meshes and the use of different bases in each element, thus rendering them naturally well suited for a discretization with adaptive h (mesh) and p (polynomial order) refinements.

While DG methods possess a number of favorable properties, one major drawback in comparison to continuous Galerkin (CG) methods on a given mesh is the larger number of degrees of freedom, which consequently translates into greater computational costs. The preliminary comparison study in [13] of CG and DG methods for the SWE shows that, when using linear interpolation on identical meshes, the cost per time step of the DG approach on serial machines is approximately four to five times more expensive than the CG approach. The subsequent study in [6] finds that the DG approach is generally more efficient in terms of achieving a specified error level for a given computational cost and in terms of scalability on large-scale parallel machines. Note that [13,6] use triangular meshes in their studies.

A main objective of this work is to examine the numerical performance of high-order DG schemes in comparison to linear-element DG schemes for the nonlinear SWE. Here, a high-order method refers to a scheme that is formally higher than second order. We adopt this definition since widely-used SWE solvers for environmental flow applications are mostly first or second order accurate. Note that, in a DG context, such a scheme is devised by using a local expansion polynomial of degree greater than unity, *i.e.*, $p > 1$. In particular, we examine the numerical performance of two DG schemes: a nodal DG scheme [14] and hybrid modal/nodal DG scheme [15]. These two schemes use different variants of polynomial bases (hence their namesake) in the approximation. The nodal DG scheme is based on a Lagrange polynomial basis. The Lagrange polynomial basis functions possess an interpolation property, *i.e.*, their value is unity at their associated nodes and vanishes at other nodes. The nodal DG scheme takes advantage of this property in constructing an efficient quadrature free approach for evaluating integral terms appearing in the DG weak formulation. The hybrid modal/nodal scheme, devised by Gassner et al. [15] is based on a pair of the so-called polymorphic nodal bases on a polygon which consists of an orthogonal modal basis and its nodal basis counterpart. The former is utilized in realizing the DG discretization and the latter is employed in evaluating integral terms. We assess the performance of these DG schemes, in terms of accuracy, computational time, and computational cost per accuracy, through their application to test problems. Note that, in this work, we limit our test problems to those with sufficiently smooth solutions on a large simple domain. Generally, problems with smooth solutions permit high-order schemes to perform at their best. Although they do present fewer numerical challenges, smooth-solution problems are in fact frequently encountered in a large class of environmental flow applications that includes tides, hurricanes, non-breaking waves, and many others.

This work is also motivated in part by an observation that a quadrilateral element may be obtained by merging two adjacent triangular elements and *vice versa*, two triangular elements formed by bisecting a quadrilateral element. In this mesh setting, a mesh of quadrilateral elements would consist of approximately half as many elements as a mesh of triangular elements. The number of edges in the quadrilateral mesh would be approximately two-thirds that of the triangular mesh. Since evaluating area integrals and edge integrals represents the major computational cost in DG methods, the use of quadrilateral elements would appear to be an appealing means to improve the computational efficiency of DG schemes. To gain more insight into this idea, we examine the performance of DG solutions with expansion basis functions defined for various element shapes. More specifically, for the nodal DG methods, we consider the Lagrange nodal bases on triangles and tensor-product nodal bases on quadrilaterals. For the hybrid modal/nodal DG methods, we consider not only the DG solutions of polymorphic bases on triangular and quadrilateral elements but also polygonal elements.

One issue that arises in DG schemes and other methods based on the SWE in conservative form concerns their ability to preserve steady state at rest in a problem with a spatially varying bed, the well-balanced property [16,17]. A straightforward treatment of the bed term may not balance exactly (at the computational level) the gradient flux term and the bed term and thus may lead to a failure in maintaining the steady state at rest. It is demonstrated in [17] that the well-balanced property generally yields a more accurate solver. In a DG framework, several well-balanced schemes have been devised, see *e.g.*, [18–21,11,9] and references therein. In this work, we discuss treatment and realization aspects to be considered in order to achieve a well-balanced property in high-order DG scheme based on nodal bases.

This paper is organized as follows. In Section 2, we provide a description of the two-dimensional nonlinear SWE. Section 3 summarizes a general framework of the DG method employed in this work. Subsequently, we describe two different bases to use with the DG method, namely, the so-called polymorphic nodal bases and the nodal bases. In Section 4, we present a performance assessment of the hybrid modal/nodal DG schemes and the nodal DG scheme through two test problems: a nonlinear problem with a smooth manufactured solution and the nonlinear Stommel problem. Since the manufactured-solution problem has an exact solution, it permits an accurate measure of the error. We therefore use this problem in a comprehensive performance study (Section 4.2). In Section 4.3, we report numerical results of the nodal DG solution to the nonlinear Stommel problem with a flat bed as well as a non-flat bed. The non-flat bed test case is also employed in the study of the well-balanced property. Although it has a relatively simple structure, the nonlinear Stommel problem contains all the terms

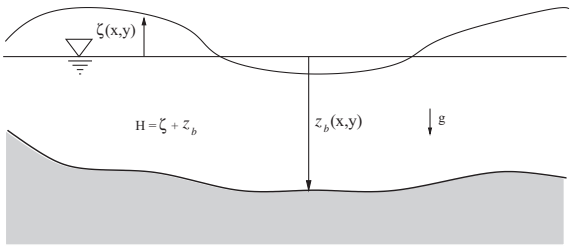


Fig. 1. Schematic diagram of the free surface and bathymetry.

present in realistic applications, including the Coriolis force, surface wind stress, and bottom friction. Conclusions from the study are drawn in Section 5.

2. Governing equations: shallow water equations

We consider the two-dimensional nonlinear SWE which consist of the depth-averaged continuity, x-, and y-momentum equations written in conservative form as follows,

$$\frac{\partial \mathbf{q}}{\partial t} + \nabla \cdot \mathbf{F}(\mathbf{q}) = \mathbf{s}(\mathbf{q}, \boldsymbol{x}, t) \tag{1}$$

where the vector of the conserved variables $\mathbf{q}$, the shallow water flux $\mathbf{F} = (\boldsymbol{f}(\mathbf{q}), \boldsymbol{g}(\mathbf{q}))$, and the vector of forcing terms $\mathbf{s}$ are

$$\begin{aligned} \mathbf{q} &= \begin{pmatrix} H \\ uH \\ vH \end{pmatrix}, \quad \boldsymbol{f} = \begin{pmatrix} uH \\ u^2H + \frac{1}{2}gH^2 \\ uvH \end{pmatrix}, \quad \boldsymbol{g} = \begin{pmatrix} vH \\ uvH \\ v^2H + \frac{1}{2}gH^2 \end{pmatrix}, \\ \mathbf{s}(\mathbf{q}, \boldsymbol{x}, t) &= \begin{pmatrix} 0 \\ gH\frac{\partial z_b}{\partial x} + F_x \\ gH\frac{\partial z_b}{\partial y} + F_y \end{pmatrix}, \end{aligned} \tag{2}$$

respectively. Here, $H(\boldsymbol{x}, t)$ denotes the total water column height, u and v represent the depth-averaged velocity in the x- and y-directions, respectively, g is the magnitude of the gravitational acceleration, $z_b(\boldsymbol{x})$ represents the bathymetric depth measured positive downwards from a horizontal reference (see Fig. 1). F_x and F_y denote forcing terms in the momentum equations which may be present *e.g.*, Coriolis force, bottom frictional stresses, surface stresses. Note that, in this study, we consider the effect of momentum diffusion from turbulence negligible and the terms describing such an effect are excluded from the equations.

3. Methodology

3.1. Discontinuous Galerkin methods for hyperbolic balance laws

We first describe a framework of the specific DG formulation employed in this study. For simplicity of presentation, we describe a DG discretization of 2-dimensional scalar hyperbolic balance laws of the form

$$\frac{\partial u(\boldsymbol{x}, t)}{\partial t} + \nabla \cdot \boldsymbol{f}(u(\boldsymbol{x}, t)) = \mathrm{s}(u(\boldsymbol{x}, t), \boldsymbol{x}, t), \quad (\boldsymbol{x}, t) \in \Omega \times [0, \infty), \quad \Omega \in \mathbb{R}^2, \tag{3}$$

where $u(\boldsymbol{x}, t)$ is a conserved variable, $\boldsymbol{f} = (f_x, f_y)$ is a nonlinear flux with f_x and f_y denoting a flux function in the x- and y-direction, respectively, and $\mathrm{s}(u, \boldsymbol{x}, t)$ is a (non-stiff) source term. A DG discretization of the SWE (1) is a straightforward extension of the procedure for discretizing (3). However, we note that the bed-slope term requires additional attention in order to obtain a scheme that preserves still water (see Section 4.3.2.1 for discussion on this issue). To discretize (3) using DG methods, the domain Ω is subdivided into a set of finite non-overlapping elements. Let $\mathcal{T}_h$ denote such a set of elements. The solution u is then replaced by a discontinuous approximate solution u_h which, in each element $K \in \mathcal{T}_h$, belongs to a finite dimensional space $V(K)$. The approximate solution on the element K is determined by requiring that,

$$\int_K \frac{\partial u_h}{\partial t} v d\boldsymbol{x} - \int_K \boldsymbol{f}(u_h) \cdot \nabla v d\boldsymbol{x} + \int_{\partial K} \widehat{\boldsymbol{f}} \cdot \mathbf{n} v ds = \int_K \mathrm{s}(u_h, \boldsymbol{x}, t) v d\boldsymbol{x} \tag{4}$$

for all $v \in V(K)$, where $\mathbf{n}$ represents the outward-pointing unit normal vector. The so-called numerical flux $\widehat{\boldsymbol{f}}$, also known as the Riemann solver, resolves the flux $\boldsymbol{f}(u_h)$ being multiply-defined on the element boundary arising from the approximation being discontinuous across the element interface. The numerical flux, which depends on the traces from both sides of the

element interface, is essential for the stability, convergence, and efficiency of the DG method (see examples of different numerical fluxes in *e.g.*, [22,23]). Note that the coupling between the approximate solution in K and in its immediate neighbors enters the weak formula (4) only through the edge integral term.

Suppose here that a finite dimensional space $V(K)$ (with desirable properties) is chosen for each element K and that $\{\widetilde{\phi}_m^K(\boldsymbol{x})\}_{m=1,\ldots,N_p}$ forms a basis of $V(K)$, where N_p denotes the dimension of the space $V(K)$. The approximate solution, when restricted to K, is then defined by

$$u_h|_K = \sum_{m=1}^{N_p} \widetilde{u}_m^K(t)\widetilde{\phi}_m^K(\boldsymbol{x}), \quad \boldsymbol{x} \in K, \tag{5}$$

where $\widetilde{u}_m^K(t)$ represents the time-dependent expansion coordinates. The global approximate solution corresponds simply to a direct sum of (5) over all elements. By adopting this basis, the local statement (4) for the element K reduces to the following system of ordinary differential equations (ODEs):

$$\sum_{n=1}^{N_p} \mathcal{M}_{m,n}^K \frac{d\widetilde{u}_n^K}{dt} - \int_K f_x(u_h)\frac{\partial \widetilde{\phi}_m^K}{\partial x}d\boldsymbol{x} - \int_K f_y(u_h)\frac{\partial \widetilde{\phi}_m^K}{\partial y}d\boldsymbol{x} + \int_{\partial K} \widehat{\boldsymbol{f}}_h \cdot \mathbf{n}\widetilde{\phi}_m^K ds = \int_K \mathrm{s}(u_h, \boldsymbol{x}, t)\widetilde{\phi}_m^K d\boldsymbol{x}, \quad m = 1, \ldots, N_p, \tag{6}$$

where $\mathcal{M}_{m,n}$, an entry of the element mass matrix, is defined by

$$\mathcal{M}_{m,n}^K = \int_K \widetilde{\phi}_m^K(\boldsymbol{x})\widetilde{\phi}_n^K(\boldsymbol{x})d\boldsymbol{x}. \tag{7}$$

Note that the superscript K is used to indicate an affiliation of the basis functions and their expansion coordinates with the element K. Hereafter, this superscript is dropped for notational simplicity.

The area and edge integrals are conventionally evaluated by using a quadrature rule; for example, the area integral involving $f_x(u_h)$ is realized through

$$\sum_{r=1}^{N_c} w_{c,r}\left(f_x(u_h)\frac{\partial \widetilde{\phi}_m}{\partial x}\right)\Bigg|_{\boldsymbol{x}_{c,r}},$$

where $(w_{c,i}, \boldsymbol{x}_{c,i})$ is a quadrature weight and point location pair and N_c is the number of quadrature points. We note that the accuracy of the quadrature to be used depends largely on the form of the integrands. In this work, we consider a technique frequently used in spectral methods and in nodal DG methods [24,14,25,15] to evaluate these integrals. This technique relies on the so-called nodal basis, another basis spanning $V(K)$, to construct a simple but efficient means in treating the nonlinear terms. Here, let $\{\phi_m \in V(K)\}_{m=1,\ldots,M_p}$ with $M_p \geqslant N_p$ be a nodal basis associated with the interpolation points $\{\boldsymbol{x}_m \in K\}_{m=1,\ldots,M_p}$. The nodal basis functions possess the so-called interpolation property, namely,

$$\phi_m(\boldsymbol{x}_n) = \delta_{m,n} = \begin{cases} 1, & \text{for } m = n, \\ 0, & \text{for } m \neq n. \end{cases} \tag{8}$$

Here, we allow the number of nodal basis functions M_p to be greater than the number of trial basis functions N_p. In the case where $M_p > N_p$, the property (8) of the nodal basis functions holds in an approximate sense only. With the nodal basis at hand, the nonlinear flux term is approximated as an interpolant as follows

$$f_x(u_h, \boldsymbol{x}) \approx (If_x)(\boldsymbol{x}) \equiv \sum_{m=1}^{M_p} f_{x,m}\phi_m(\boldsymbol{x}) = \boldsymbol{\phi}^T \boldsymbol{f}_x, \tag{9}$$

where $\boldsymbol{f}_x = \{f_{x,1}, \ldots, f_{x,M_p}\}^T$ and $\boldsymbol{\phi} = \{\phi_1, \ldots, \phi_{M_p}\}^T$. The nodal representation of the y-directed flux $(If_y)(\boldsymbol{x})$ is defined in an analogous fashion. Here, the nodal coordinates are simply defined by $f_{x,m} = f_x(u_h(\boldsymbol{x}_m))$. By adopting the nodal representation for the nonlinear flux term and the source term, the formula (6) becomes the following system of ODEs,

$$\sum_{n=1}^{N_p} \mathcal{M}_{m,n}\frac{d\widetilde{u}_n}{dt} - \sum_{n=1}^{M_p}\left(\mathcal{S}_{x,(m,n)}f_{x,n} + \mathcal{S}_{y,(m,n)}f_{y,n}\right) + \int_{\partial K} \widehat{\boldsymbol{f}}_h \cdot \mathbf{n}\widetilde{\phi}_m ds = \sum_{n=1}^{M_p} \widetilde{\mathcal{M}}_{m,n}\mathrm{s}_n, \quad m = 1, \ldots, N_p. \tag{10}$$

where $\mathrm{s}_n = \mathrm{s}(u_h(\boldsymbol{x}_n), \boldsymbol{x}_n, t)$, and the general element mass matrix and the general element stiffness matrices are

$$\widetilde{\boldsymbol{\mathcal{M}}} = (\widetilde{\mathcal{M}}_{m,n}), \quad \widetilde{\mathcal{M}}_{m,n} = \int_K \widetilde{\phi}_m \phi_n d\boldsymbol{x} \tag{11}$$

$$\boldsymbol{S}_x = (\mathcal{S}_{x,(m,n)}), \quad \mathcal{S}_{x,(m,n)} = \int_K \frac{\partial \widetilde{\phi}_m}{\partial x}\phi_n d\boldsymbol{x} \tag{12}$$

$$\boldsymbol{S}_y = (\mathcal{S}_{y,(m,n)}), \quad \mathcal{S}_{y,(m,n)} = \int_K \frac{\partial \widetilde{\phi}_m}{\partial y}\phi_n d\boldsymbol{x}. \tag{13}$$

Notice that, with these nodal representations, the volume integrals involving the nonlinear flux and the source term reduces to matrix–vector multiplications. Note that the edge integrals can be treated in a similar fashion; see Appendix B. The element matrices of each element can be computed exactly (or approximately) and stored at the initial stage of the simulation, leading to a quadrature free approach [26]. This nodal-integration approach provides a simple means to evaluating the integral terms and offers a computational advantage in the sense that the number of operations required is proportional to the number of nodes regardless of the form of the non-linear flux and source term. The disadvantage of this approach is that there is an error introduced through the interpolation of the nonlinear flux and the source term. Such an error, known as an alias error, may induce an instability for marginally resolved computations [24,25]. In this case, an instability can still be effectively controlled by employing a de-aliasing strategy [24,25].

Since there is no functional-continuity requirement in the expansion coordinates belonging to different elements, a global system of ODEs is composed simply of the system of ODEs (10) from all elements. To solve such a global system, we invert the mass matrix (a matrix associated with the time-derivative term) and apply a time stepping scheme to the resulting explicit system of ODEs. Since the expansion coordinates from different elements enter (10) only through the numerical flux term, the mass matrix is a block diagonal matrix with the element mass matrices as the diagonal block entries. The inverse of the mass matrix can thus be easily computed by inverting each element mass matrix. Note that an inversion of the element mass matrices can be done once and for all at the initial phase of the simulation. Note this procedure can be made trivial by choosing the local basis $\{\widetilde{\phi}_m\}$ forming an orthogonal set since, in this case, the element mass matrix is a diagonal matrix.

The remaining tasks in defining a DG scheme concern choosing the finite dimensional approximation space, its associated basis functions, a time discretization scheme, and a Riemann solver. The next two subsections describe two particular sets of polynomial bases, namely the polymorphic nodal bases and the nodal bases, to be used in the framework described above. Thereafter, we discuss in brief a time integration scheme employed in this study.

3.2. Polymorphic nodal elements: modal and nodal basis

Below, we summarize the construction of the so-called polymorphic nodal bases for a convex polygon devised by Gassner et al. [15]. Such bases consist of an orthogonal polynomial basis to be used as a set of trial and test functions and its associated nodal basis counterpart to be used in treating nonlinear terms.

For a given convex polygon K, we introduce a coordinate transformation $\xi_K : K \to \mathcal{K}$, connecting an element K with a so-called reference element $\mathcal{K}$, as follows

$$\boldsymbol{\xi} = \xi_K(\boldsymbol{x}) = \frac{\boldsymbol{x} - \boldsymbol{x}_c}{\Delta X}, \quad \boldsymbol{x} \in K, \tag{14}$$

where $\boldsymbol{\xi} = (\xi, \eta)$, $\boldsymbol{x} = (x, y)$, $\boldsymbol{x}_c$ denotes the centroid of K, and $\Delta X = \max(x_{\max} - x_{\min}, y_{\max} - y_{\min})$ is a scaling factor (see Fig. 2). The reference element $\mathcal{K}$ is the range of the coordinate transformation. Note that the transformation (14) amounts simply to a rigid-body translation and linear scaling of the element K. Let $\{\pi_m\}_{m=1,\ldots,N_p}$ be the monomial basis of $P^p(\mathcal{K})$, the space of polynomials with degree of at most p, namely

$$\pi_m(\boldsymbol{\xi}) = \xi^i \eta^j, \quad i, j \geqslant 0,\ i + j \leqslant p, \tag{15}$$

$$m = \frac{1}{2}(i + j + 1)(i + j + 2) - i. \tag{16}$$

The number of basis functions N_p and the order p are related through

$$N_p = \frac{(p + 1)(p + 2)}{2}. \tag{17}$$

An orthonormal basis $\{\widetilde{\phi}_m(\boldsymbol{\xi})\}_{m=1,\ldots,N_p}$ is subsequently constructed by applying a modified Gram–Schmidt process [27] with the usual L_2 inner product to the monomial basis. Consequently, the basis functions $\widetilde{\phi}_m(\boldsymbol{\xi})$ possess the orthonormal property, more precisely,

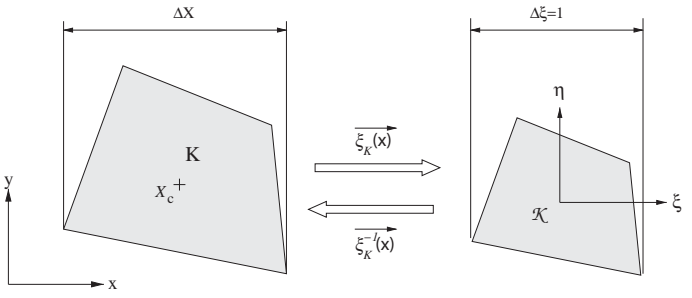


Fig. 2. Schematic diagram of the coordinate transformation.

$$\int_{\mathcal{K}} \widetilde{\phi}_m(\boldsymbol{\xi})\widetilde{\phi}_n(\boldsymbol{\xi})d\boldsymbol{\xi} = \delta_{m,n}.$$

The basis functions on the physical element K can then be defined as follows

$$\widetilde{\phi}_m(\boldsymbol{x}) = (\widetilde{\phi}_m \circ \boldsymbol{\xi}_K)(\boldsymbol{x}), \quad m = 1, \ldots, N_p. \tag{18}$$

Here, for notational simplicity, we use an identical notation for the basis functions on physical element K and on the reference element $\mathcal{K}$. Since the transformation (14) is affine, the space spanned by $\{\widetilde{\phi}_m(\boldsymbol{x})\}$ is therefore a space of polynomials with degree of at most p. In addition, $\{\widetilde{\phi}_m(\boldsymbol{x})\}$ forms an orthogonal basis over K owing to the geometric transformation (14) having a constant Jacobian. For a given set of nodal points $\Omega_I(p) = \{\boldsymbol{x}_m\}_{m=1,\ldots,M_p} \subset K$, a so-called nodal basis $\{\phi_m(\boldsymbol{x})\}_{m=1,\ldots,M_p}$ and it associated coordinate $\{u_m\}_{m=1,\ldots,M_p}$ are constructed from considering the following conditions, for $u(\boldsymbol{x}) \in P^p(K)$,

$$\phi_m(\boldsymbol{x}_n) = \delta_{m,n}, \quad u(\boldsymbol{x}) = \sum_{m=1}^{N_p} \widetilde{u}_m \widetilde{\phi}_m(\boldsymbol{x}) \doteq \sum_{m=1}^{M_p} u_m \phi_m(\boldsymbol{x}). \tag{19}$$

As a result, the transformations between two representations are determined by

$$\mathbf{u} = \mathbf{V}\widetilde{\mathbf{u}} \quad \text{and} \quad \widetilde{\boldsymbol{\phi}} = \mathbf{V}^T\boldsymbol{\phi} \tag{20}$$

where $\mathbf{u} = \{u_1, \ldots, u_{M_p}\}^T$, $\widetilde{\mathbf{u}} = \{\widetilde{u}_1, \ldots, \widetilde{u}_{N_p}\}^T$, $\boldsymbol{\phi} = \{\phi_1, \ldots, \phi_{M_p}\}^T$, $\widetilde{\boldsymbol{\phi}} = \{\widetilde{\phi}_1, \ldots, \widetilde{\phi}_{N_p}\}^T$ and $\mathbf{V}$ is a generalized Vandermonde matrix whose entries are given by

$$V_{m,n} = \widetilde{\phi}_n(\boldsymbol{x}_m), \qquad m = 1, \ldots, M_p, \quad n = 1, \ldots, N_p. \tag{21}$$

The remaining task in defining the nodal basis involves choosing a nodal set. The distribution of the nodal points has a crucial implication on the quality of an interpolant. It is known that a high quality interpolant, indicated by a small value of the Lebesgue constant [28], can be achieved with node sets having nodal points clustered in the vicinity of the boundaries of the element. Note that the Lebesgue constant indicates how far the interpolant may deviate from the best polynomial approximation of the function. Here, we use the specific framework devised by Gassner et al. [15] to generate a nodal set yielding such a desirable effect. Such a nodal set consists of a set of nodes on the element boundary and a set of nodes in the interior. The interior nodes are generated by nesting a set of the scaled-down boundary nodes in a way that the nodes are denser near the boundaries. More precisely, a nodal set on a given polygon of N_{gon} sides is constructed from the following formula,

$$\Omega_I(p) = \bigcup_{r=0}^{r_{\max}} \mathcal{M}_r(\Omega_I^S(p - (N_{\text{gon}} - p_d)r)) \tag{22}$$

with

$$r_{\max} = \text{floor}\left(\frac{p}{N_{\text{gon}} - p_d}\right) \tag{23}$$

and $0 \leqslant p_d < N_{\text{gon}}$. Here, $\Omega_I^S(q)$ denotes a set of boundary nodes which has $q+1$ nodes with the Gauss–Lobatto node distribution on each edge of the considered polygon and $\Omega_I^S(0) = \{\boldsymbol{x}_c\}$ where $\boldsymbol{x}_c$ is the centroid of the polygon. The mapping $\mathcal{M}_r$ generates the interior nodes by scaling down, with a certain factor depending on the nesting step r, the boundary point set Ω_I^S. Note that $\mathcal{M}_0$ is an identity mapping. See Gassner et al. [15] for a detailed account of the mapping $\mathcal{M}_r$. Fig. 3 shows, as an example, a nodal set for $p = 5$ on a quadrilateral, triangular, and pentagonal element. Note that the formula (22) is applicable for an arbitrary p. It uses the parameter p_d to adjust the number of interior nodes. See Fig. 3(a) and (b) for a comparison of the node sets with different values p_d. Note that including more interior points by increasing the value of p_d improves the quality of an interpolant [15], however, at the expense of computational efficiency in terms of the number of operations required. It is noted that the number of nodes M_p, p, and p_d are related thorough

$$M_p = N_{\text{gon}}(r_{\max} + 1)\left\{p - \frac{1}{2}(N_{\text{gon}} - p_d)r_{\max}\right\} + \delta_{0,p-(N_{\text{gon}}-p_d)r_{\max}}. \tag{24}$$

Table 1 tabulates N_p and M_p of the triangular and quadrilateral polymorphic elements with p ranging from 1 to 6.

The number of nodal points M_p from this construction is in general greater than N_p (except for a triangular element where $M_p = N_p$). For $M_p \neq N_p$, an inverse of the Vandermonde matrix is not uniquely defined. To circumvent this issue, a pseudo-inverse matrix defined in the least squares sense, more specifically,

$$\boldsymbol{V}^{-1} \equiv \overline{\boldsymbol{V}}^{-1}\boldsymbol{V}^T, \quad \overline{\boldsymbol{V}} = \boldsymbol{V}^T\boldsymbol{V}, \tag{25}$$

is utilized in defining the inverse transformations

$$\widetilde{\boldsymbol{u}} = \boldsymbol{V}^{-1}\boldsymbol{u} \quad \text{and} \quad \boldsymbol{\phi} = (\boldsymbol{V}^{-1})^T\widetilde{\boldsymbol{\phi}}. \tag{26}$$

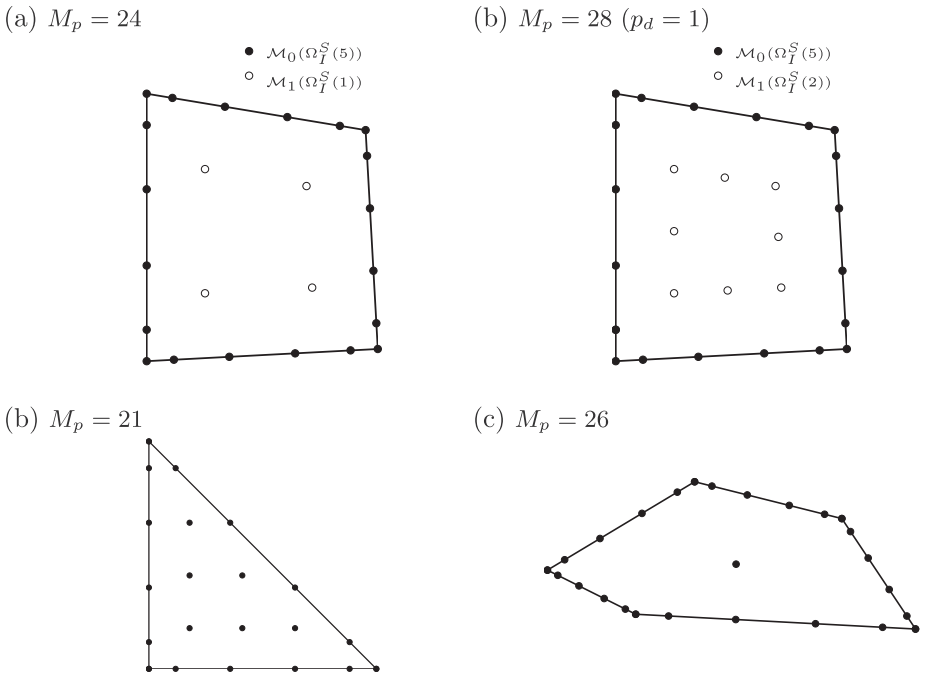


Fig. 3. Nodal distribution for $p = 5$ ($N_p = 21$): (a) quadrilateral element with $p_d = 0$ (b) quadrilateral element with $p_d = 1$ (c) triangular element with $p_d = 0$, and (d) pentagonal element $p_d = 0$.

Table 1
Degrees of freedom N_p and the number of nodal points M_p per element of triangular and quadrilateral polymorphic elements.

Degree p	Tri element $M_p = N_p$	Quad element		
		N_p	M_p	
			$p_d = 0$	$p_d = 1$
1	3	3	4	4
2	6	6	8	8
3	10	10	12	13
4	15	15	17	20
5	21	21	24	28
6	28	28	32	37

Note that for $M_p > N_p$, the set $\{\phi_m\}$ defined as above, although it spans every polynomial in $P^p(K)$, is not a basis since it is a linearly dependent set. However, for simplicity, we still call such a set the nodal basis and its members, nodal basis functions. As a consequence of using the pseudo-inverse Vandermonde matrix (25) in defining the nodal basis functions, the nodal basis function is close but not identical to unity at its associated node and is close but not identical to zero at the other nodes, *i.e.*, $\phi_m(x_n) \neq \delta_{m,n}$. Therefore, a function value at the nodal points of the polynomial approximation of a function $f(\boldsymbol{x})$ defined by

$$(I_p f)(\boldsymbol{x}) = \sum_{m=1}^{M_p} f(\boldsymbol{x}_m)\phi_m(\boldsymbol{x}) = \boldsymbol{\phi}^T \boldsymbol{f}$$

is in general not identical to the value of the nodal coordinate, *i.e.*, $(I_p f)(\boldsymbol{x}_i) \neq f(\boldsymbol{x}_i)$. Note that, in practice, an explicit form of the nodal basis functions is rarely required. Instead, interpolated values at given points are obtained by first calculating the modal coordinates $\widetilde{\boldsymbol{f}} = \{\widetilde{f}_1, \ldots, \widetilde{f}_{N_p}\}^T$ from the nodal coordinates $\boldsymbol{f} = \{f_1, \ldots, f_{M_p}\}^T$ by means of an inverse transformation and subsequently calculating the interpolated values through the modal representation.

Note that the scheme based on the polymorphic nodal bases utilizes the modal basis functions as the trial and test functions in the DG formulation. Owing to the orthogonality of the modal basis, the global mass matrix of this scheme is diagonal which can be trivially inverted. The element matrices in the ODEs (10) can be easily realized with the use of the change-of-bases transformations (20) and (26). More precisely, we evaluate the element general stiffness matrix by considering

$$\mathbf{S}_x = \mathbf{V}^T \mathscr{S}_x, \quad \text{where } \mathscr{S}_x \equiv \int_K \frac{\partial \boldsymbol{\phi}}{\partial x} \boldsymbol{\phi}^T d\boldsymbol{x}. \tag{27}$$

The calculation of the general stiffness matrix amounts to determining a stiffness matrix $\mathscr{S}_x$. We use a technique similar to that devised by Hesthaven and Warburton [25] in evaluating such a stiffness matrix. This technique, which does not require Gaussian integration, is given in Appendix A.

3.3. Nodal elements: nodal bases on triangles and quadrilaterals

The DG scheme based on the nodal bases uses a nodal basis not only as an efficient means for treating nonlinear flux terms but also as trial and test functions in the DG formulation, in other words, in this scheme, $\widetilde{\phi}_m(\boldsymbol{x})$ corresponds simply to $\phi_m(\boldsymbol{x})$. Here, a nodal basis on triangles and tensor-product nodal basis on quadrilaterals are considered.

For triangular elements, although the nodal basis on triangles constructed as in the previous subsection represents an excellent candidate, we consider a nodal basis with a set of interpolation points described in [14,29,25]. Unlike the nodal basis described in the last subsection which is constructed in an element-by-element fashion, such a nodal basis is defined in a more conventional way, *i.e.*, through a set of nodal basis functions on a single master triangle. More precisely, the nodal basis $\{\phi_m(\boldsymbol{\xi}) \in P^p(I_t)\}_{m=1,\ldots,N_p}$ associated with a given nodal set $\{\boldsymbol{\xi}_m \in I_t\}_{m=1,\ldots,N_p}$ on the master element $I_t = \{\boldsymbol{\xi} = (\xi,\eta) \mid \xi,\eta \geqslant -1 \text{ and } \xi+\eta \leqslant 0\}$ is first constructed using the approach described in the last subsection. Subsequently, nodal basis functions on the physical straight-edged triangular element K are defined as $\phi_m(\boldsymbol{x}) = (\phi_m \circ \boldsymbol{x}_K^{-1})(\boldsymbol{x})$ where $\boldsymbol{x}_K^{-1}$ is an inverse mapping of the affine mapping $\boldsymbol{x}_K : I_t \to K$:

$$\boldsymbol{x}_K(\boldsymbol{\xi}) = \sum_{i=1}^{3} L_{t,i}\boldsymbol{x}_i^K \tag{28}$$

where $\boldsymbol{x}_i^K$ denotes a coordinate of the ith-vertex of the element (the vertices are numbered in a counter clockwise manner) and the functions $L_{t,i}$ are defined by

$$L_{t,1} = -(\xi+\eta)/2, \quad L_{t,2} = (\xi+\eta)/2, \quad \text{and} \quad L_{t,3} = (1+\eta)/2.$$

Defining the nodal basis in this way presents an advantage in that element matrices, *i.e.*, mass and stiffness matrices, can be simply obtained by appropriately scaling the element matrices associated with the master elements owing to the mapping (28) having a constant Jacobian. Consequently, the amount of computer memory required and also computational costs in evaluating the element matrices are lower than a scheme using the nodal basis constructed directly on the physical elements. It is noted that we use the near-optimal set of nodal points on the master element given by Hesthaven [29] and Hesthaven and Warburton [25] (as an example, see Fig. 4(a) for such a nodal set with $p = 5$). In comparison to the nodal set on a triangle defined by (22), this near-optimal nodal set has a slightly lower value of the Lebesgue constant for the range of p considered in this work (see [25,15] for the Lebesgue constant of theses sets).

For nodal quadrilateral elements, instead of working with P^p, the approximation space on the master element $I_q = [-1,1]^2$ is selected as $Q^p(I_q) = P^p([-1,1]) \times P^p([-1,1])$, the tensor products of $P^p([-1,1])$, a space of one-dimensional polynomials of degree at most p. Let $\{P_i(x)\}_{i=0,\ldots,p}$ be the normalized Legendre polynomial basis on $[-1,1]$, a two-dimensional orthonormal basis on I_q can then be defined by

$$\psi_{(p+1)j+i+1}(\boldsymbol{\xi}) \equiv P_i(\xi)P_j(\eta), \quad 0 \leqslant i,j \leqslant p. \tag{29}$$

The number of basis functions and the order p in this case is related through

$$N_p = (p+1)^2. \tag{30}$$

Note the higher number of degrees of freedom in comparison to elements of P^p-type for an given interpolation order p. A nodal basis $\{\phi_m\}_{m=1,\ldots,N_p}$ is then constructed in an identical way described in the last subsection, provided that a set of nodal points $\{\boldsymbol{\xi}_m \in I_t\}_{m=1,\ldots,N_p}$ is given. Here, we consider the set of interpolation points with a Legendre–Gauss–Lobatto distribution, which is given by

$$\boldsymbol{\xi}_{(p+1)j+i+1} = (x_i, x_j), \quad 0 \leqslant i,j \leqslant p, \tag{31}$$

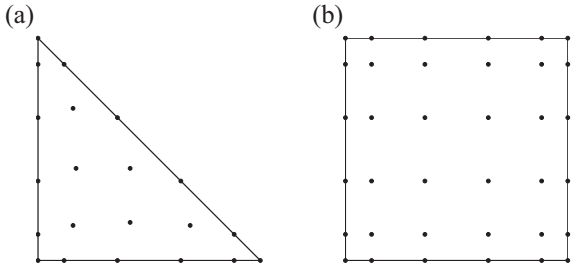


Fig. 4. Distribution of interpolation points on the master elements with $p = 5$: (a) triangular element, and (b) rectangular element.

where $\{x_i\}_{i=0,\ldots,p}$ are the zeros of the function $(1-x)^2 d(P_{p-1}(x))/dx$ (a nodal set with the classical two-dimensional Legendre–Gauss distribution was also considered in [30]). Fig. 4 depicts the nodal set for $p = 5$. The nodal basis functions on the physical (convex) quadrilateral element K are then defined as $\phi_m(\boldsymbol{x}) = (\phi_m \circ \boldsymbol{x}_K^{-1})(\boldsymbol{x})$ with a bi-linear mapping $\boldsymbol{x}_K : I_q \to K$:

$$\boldsymbol{x}_K = \sum_{i=1}^{4} \boldsymbol{x}_i^K L_{q,i}(\boldsymbol{\xi}) \tag{32}$$

where $\boldsymbol{x}_i^K$ denotes a coordinate of the ith-vertex of K (the vertices are numbered in a counter clockwise manner) and

$$L_{q,1} = (1-\xi)(1-\eta)/4, \quad L_{q,2} = (1+\xi)(1-\eta)/4,$$
$$L_{q,3} = (1+\xi)(1+\eta)/4, \quad \text{and} \quad L_{q,4} = (1-\xi)(1+\eta)/4.$$

Note that except for rectangular and four-sided parallelogram elements, the Jacobian of the mapping (32) is not a constant; as a consequence, the element matrices (*i.e.*, element mass and stiffness matrices) of each element can no longer be obtained by scaling the element matrices associated with the master element. While they can be computed accurately and subsequently stored element-by-element, we adopt a less accurate but more memory-economical approach in approximating such matrices [30]. Such an approach, owing to the use of a (fixed order) classical two-dimensional Gauss quadrature, defines the approximate element matrices as a multiplication of the precomputed matrices defined on the master element and the precomputed matrices involved with the coordinate mapping. The coordinate-mapping matrices, which vary element-by-element, are diagonal and thus require less storage.

3.4. Temporal discretization

The system of ODEs governing the time evolution of the discrete solution for all elements can be written as

$$\mathbf{M}\frac{d\widetilde{\mathbf{u}}_h}{dt} = \mathbf{r}(\widetilde{\mathbf{u}}_h, t) \tag{33}$$

where $\mathbf{M}$ represents the global mass matrix, $\widetilde{\mathbf{u}}_h$ denotes the global vector of the expansion coordinates (modal coordinates for the schemes based on polymorphic bases and nodal coordinates for nodal bases), and $\mathbf{r}(\widetilde{\mathbf{u}}_h, t)$ denotes the right-hand-side vector arising from the terms that are not associated with the time derivative.

The time-dependent system (33) is numerically integrated using an explicit fourth–fifth order Runge–Kutta–Fehlberg (RKF45) method (see *e.g.*, [31,32] for a detailed account of this scheme). RKF45 has a mechanism to automatically select the step size Δt used in the integration to control accuracy of the solution. Concisely, the integrator utilizes the fourth-order and fifth-order Runge–Kutta scheme that uses all values of substages of the fourth-order scheme. It accepts the solution from the fifth order subscheme and adjusts the step size to control the truncation error of the fourth order subscheme. Here, we use the RKF45 subroutine written by Shampine et al. [33]. This subroutine requires an external subroutine returning the right-hand-side vector of the ODEs. We summarize in Appendix B a brief outline of steps used in an implementation of the calculation of the right-hand-side vector $\mathbf{M}^{-1}\mathbf{r}$. Note that, in the RKF45, the temporal accuracy of the solution is controlled by the parameters `relerr` and `abserr`, denoted here as ε_r and ε_a ($\varepsilon_r > \varepsilon_a$). Since we focus on assessing the accuracy of the spatial discretization, the values of these parameters are set to sufficiently small values in order to keep temporal discretization errors negligible when compared with spatial errors.

4. Numerical experiments

The numerical performance of the nodal DG (NDG) method and the polymorphic nodal DG (PNDG) method (*i.e.*, the hybrid modal/nodal DG method) are assessed by evaluating their accuracy, computing times, and computational cost per accuracy. To facilitate the investigation, we consider a nonlinear problem with a smooth manufactured solution as well as the nonlinear Stommel problem as test problems. The manufactured-solution problem has an *a priori* defined exact solution and thus allows for an accurate measure of error. We therefore use this problem in our comprehensive assessment. The performance study is carried out by systematically varying the interpolation order p of the DG schemes and the element size h of the computational mesh.

In the study, we mainly use the broken L_2 norm

$$\|f(\boldsymbol{x})\|_{\Omega_h} = \left(\sum_{K\in\mathcal{T}_h}\int_K f(\boldsymbol{x})^2 d\boldsymbol{x}\right)^{1/2} \tag{34}$$

in measuring the error in the approximate solution. Computing times reported below are an average of at least two identical simulations. It is important to note that the computing times closely relate to the implementation details. The main computing cost involves evaluations of the right-hand-side of the ODEs (33). The computing times reported here correspond to the results from using an implementation outlined briefly in Appendix B for the evaluation of the right-hand-side term.

4.1. Numerical flux

In this study, we use the local Lax–Friedrichs (LLF) flux as a numerical flux in the DG discretization. To define this flux, consider two adjacent elements K^- and K^+ and let e be their common edge (which is not necessarily the entire edge of an element). The LLF flux is defined as follows, for $\boldsymbol{x} \in e$

$$\widehat{\mathbf{F}} = \frac{\mathbf{F}(\mathbf{q}_h^-) + \mathbf{F}(\mathbf{q}_h^+)}{2} + \frac{C}{2}\mathbf{n}^{\mp}(\mathbf{q}_h^{\mp} - \mathbf{q}_h^{\pm}) \tag{35}$$

where $\mathbf{q}_h^-$ and $\mathbf{q}_h^+$ are respectively the solution value at $\boldsymbol{x}$ of the element K^- and K^+, $\mathbf{n}^- = -\mathbf{n}^+$, and the constant C corresponds to the largest value, along the edge e, of the absolute maximum eigenvalue of the normal flux Jacobian matrix,

$$\max_{s\in[\mathbf{q}_h^-,\mathbf{q}_h^+]}\left|\lambda\left(n_x\frac{\partial \boldsymbol{f}}{\partial \mathbf{q}}\bigg|_s + n_y\frac{\partial \boldsymbol{g}}{\partial \mathbf{q}}\bigg|_s\right)\right| = \max_{s\in[\mathbf{q}_h^+,\mathbf{q}_h^-]}\left[\left(|\mathbf{n}\cdot\mathbf{u}| + \left|\sqrt{gH}\right|\right)\Big|_s\right] \tag{36}$$

where $\lambda(\cdot)$ denotes the eigenvalue of the matrix. Note that the boundary conditions are enforced weakly by properly specifying an exterior state in the numerical flux along the physical boundaries such that the desirable conditions are obtained in a weak sense.

4.2. Manufactured solutions

Here, a problem with an exact solution is used as a verification tool for assessing the DG schemes. Specifically, we consider the problem in which the vector of source terms $\mathbf{s}(\mathbf{q}, \boldsymbol{x}, t)$ corresponds to a vector of terms arising from substituting the *a priori* defined smooth functions below

$$\begin{aligned} H &= 2\xi_0 \frac{\cos(\sigma(x-x_1))\cos(\sigma(y-y_1))}{\cos(\sigma(x_2-x_1))\cos(\sigma(y_2-y_1))}\cos(\omega(t+\tau)) + H_0 \\ uH &= \upsilon_0 \frac{\sin(\sigma(x-x_1))\cos(\sigma(y-y_1))}{\cos(\sigma(x_2-x_1))\cos(\sigma(y_2-y_1))}\sin(\omega(t+\tau)) \\ vH &= \upsilon_0 \frac{\cos(\sigma(x-x_1))\sin(\sigma(y-y_1))}{\cos(\sigma(x_2-x_1))\cos(\sigma(y_2-y_1))}\sin(\omega(t+\tau)) \end{aligned} \tag{37}$$

into the left hand side of (1). In (37), $\sigma, \omega, \tau, x_1, x_2, y_1, y_2, \xi_0, \upsilon_0$ and H_0 are positive constants. The value of H_0 is selected sufficiently large so that H is positive everywhere. The exact solution is used to prescribe the initial condition and the boundary conditions. Note that when the value of σ is identical to that of ω and the value of ξ_0 is identical to that of υ_0, this manufactured solution leads to a vanishing forcing term for the depth-averaged continuity equation. In all numerical calculations reported below, the values of the parameters appearing in (37) are set to $\sigma = 0.0001405$ rad/m, $\omega = 0.0001405$ rad/s, $\tau = 3456$ s, $x_1 = 40\times10^3$ m, $x_2 = 150\times10^3$ m, $y_1 = 10\times10^3$ m, $y_2 = 55\times10^3$ m, $\xi_0 = 0.25$ m, $\upsilon_0 = 0.25$ m^2/s and $H_0 = 2$ m. The simulations are performed in the rectangular computational domain of $[x_1, x_2]\times[y_1, y_2]$. The integration is carried out until $t_f = 172800$ s (a period of the solution is approximately 44,720 s).

Below, we first present numerical results computed on so-called regular meshes and subsequently results computed on unstructured meshes. We consider the DG schemes with p ranging from 1 to 5. Note that, in the PNDG scheme, we employ quadrilateral elements with $p_d = 1$ for $p = 1$ and 2 and with $p_d = 2$ for $p = 3$ to 5; for triangular elements, we use $p_d = 0$ regardless of the order p. This choice of p_d stems directly from the aspect concerning accuracy and computational operations of the polymorphic bases. The values of the parameters controlling temporal error $(\varepsilon_r, \varepsilon_a)$ in the RKF45 are set to $(5\times10^{-7}, 5\times10^{-9})$.

4.2.1. Solution computed on regular meshes

We first consider three so-called regular mesh configurations, namely, a regular triangular mesh, a rectangular mesh, and a skewed-rectangular mesh. The last configuration refers to a mesh with convex quadrilaterals. In each configuration, four nested meshes are employed in order to examine the h convergence property. In all configurations, the meshes, from the coarsest to the finest resolution, are denoted as $h, h/2, h/4$, and $h/8$, respectively. Fig. 5 shows the coarsest mesh of each configuration, which is built based on a uniform grid of 25×11 points. For the skewed-rectangular mesh configuration, the coarsest mesh is obtained by relocating interior points of the uniform grid. Each interior point is relocated in either direction from its original location with a distance varying randomly from 0 to 25% of the grid spacing. Note that the coarsest triangular mesh consists of 480 elements and the coarsest quadrilateral meshes consist of 240 elements. The three finer meshes are obtained by applying successive uniform refinements to the coarsest mesh. The refinement divides each triangle into four similar sub-triangles and uniformly divides each rectangle into four sub-rectangles (*i.e.*, the number of elements increases four times in each refinement step). Note also that for the same resolution, the number of elements in the rectangular mesh is half that of the triangular mesh.

4.2.1.1. Accuracy. As an example, we plot in Fig. 6(a), without smoothing, the approximate total water column height at the final simulation time $t_f = 172800$ from the PNDG scheme with $p = 3$ and $p_d = 1$ on the rectangular mesh of h-resolution (the

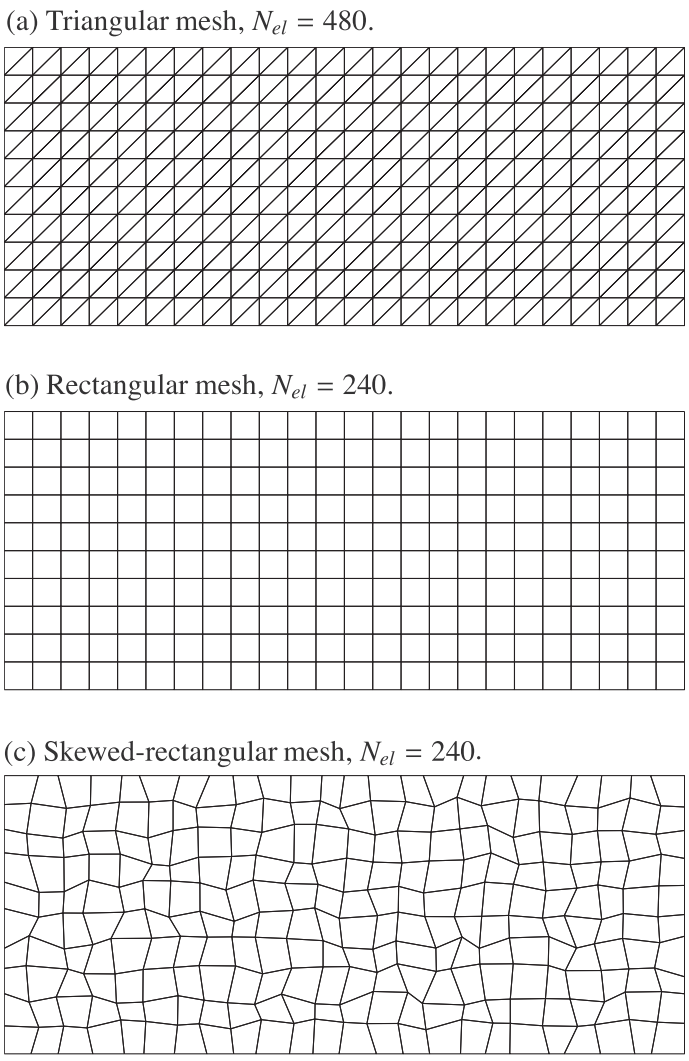


Fig. 5. Coarsest regular mesh used in the SWE with a manufactured solution; (a) triangular mesh, (b) rectangular mesh, and (c) Skew-rectangular (quadrilateral) mesh.

triangles shown there are drawn for plotting purpose so that the solution at the interior nodes can be visualized). Note the qualitative agreement with the exact solution depicted in Fig. 6(b). Table 2 tabulates the accuracy in the approximate total water column height H_h through the normalized L_2 errors, $|\Omega|^{-1/2}\|H - H_h\|_{\Omega_h}$. In this table, data from DG schemes is grouped according to an interpolation order p employed. Within each data group, we list and highlight the error of the scheme that yields the most accurate overall solution; for ease of comparison, we tabulate the errors of the other schemes as the ratio of the error of a specific scheme relative to the error of the most accurate scheme (for instance, for $p = 3$ and $h/2$-meshes, the error from the NDG scheme on the triangular mesh is 3.26 times the error from the NDG scheme on the rectangular mesh, more precisely, $3.26 \times (1.03 \times 10^{-6})$). Note that the higher the error ratio, the less accurate the solution in comparison to that of the scheme highlighted.

Evidently, the error levels in the approximate solution become smaller either as the order of basis functions p increases or as the element size decreases. It can be observed that, overall, for the same order p and similar mesh resolution, the approximate solution H_h ordering from greater to lesser accuracy corresponds to the following schemes: NDG on rectangles, NDG on skewed rectangles, NDG and PNDG on triangles, PNDG on rectangles, and PNDG on skewed rectangles. The error ratios of less accurate schemes to the most accurate scheme are higher as p increases, for example, the error ratio of the PNDG solution on rectangle meshes to the NDG solution on rectangles increases from approximately 1.1 times for $p = 1$ to roughly 24 times for $p = 5$. Note that, for triangular meshes, the PNDG and NDG schemes yield solutions with virtually indistinguishable error levels. This can be expected since both schemes use the P^p-type bases for triangular elements. It is evident that, on the rectangular mesh, the NDG scheme yields a more accurate solution than the PNDG scheme. The same can be said for the solutions from the NDG and PNDG schemes on the skewed-rectangular mesh. We believe that such a gain in accuracy is

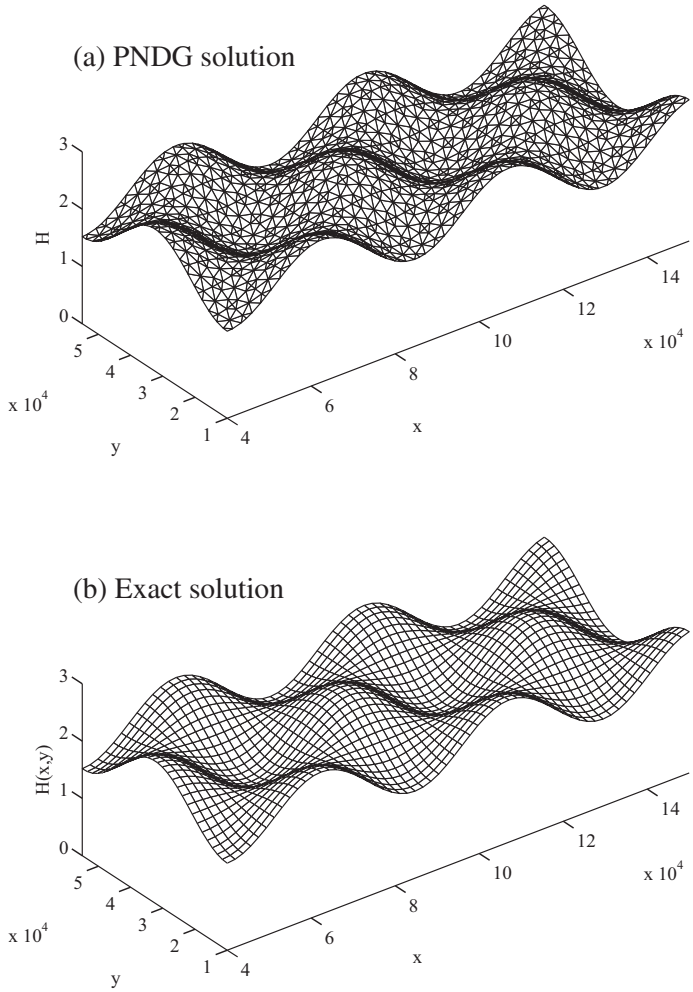


Fig. 6. Manufactured-solution test problem: total water column at $t = 2$ day; (a) H_h obtained from the PNDG scheme with $p = 3$ and $p_d = 1$ on the rectangular h-mesh; (b) manufactured exact solution.

attributed mainly to the tensor-product bases employed in the NDG scheme on rectangles being able to span additional cross polynomial terms not belonging to the span of polynomial bases employed in the PNDG scheme. Furthermore, at the same mesh resolution, the NDG scheme on rectangles yields lower error levels in H_h than the schemes on triangles even though a rectangular element used has an area that is twice that of a triangular element (however, both elements have similar edge lengths). This demonstrates to some extent the benefit of the tensor product bases in terms of accuracy. It can be noticed that, the use of skewed-rectangular elements, as expected, degrades the accuracy in H_h when compared to the use of rectangular elements. This suggests that the milder the size transition of the skewed rectangles, the more accurate the tensor-product basis solutions. Note that the NDG scheme on skewed rectangles still produces more accurate solutions than the schemes on triangles.

The numerical order of convergence, which refers to the exponent value s from fitting ch^s with c being constant to the error norm $|\Omega|^{-1/2}\|H - H_h\|_{\Omega_h}$, is reported in the last column of Table 2. We note that all the DG schemes, regardless of bases or element shape, exhibit a convergence rate of approximately $O(h^{p+1})$ for the total water column height (note that each scheme has a different value for the constant c). Note that the degradation in the order of convergence for most schemes with $p = 5$ and the $h/8$ meshes is due to the fact that the temporal errors from the RKF45 integrator, with the specific error tolerance employed, are no longer negligible in comparison to the spatial errors. Note that the observed convergence rate is higher than that of the theoretical estimate $O(h^{p+1/2})$ expected for a Lax–Friedrichs DG solution to a problem with nonlinear fluxes [34]. To examine the p-convergence properties, the error levels obtained for each mesh resolution are plotted against the order p used on the semi-log scale (error levels on a log scale and p on a linear scale). Fig. 7 shows examples of such plots for the h- and $h/2$-meshes. The curves for all DG schemes appear approximately as straight lines, indicating that all DG schemes considered exhibit the expected exponential convergence rate with respect to p. Although not reported here in

Table 2
Normalized L_2 errors in H_h, $\mathcal{E}_H \equiv |\Omega|^{-1/2}\|H - H_h\|_{\Omega_h}$, of the overall most accurate scheme for a given order p and error ratios (specific scheme relative to the most accurate scheme for that p), as computed on regular meshes, and rate of h-convergence. A code [ppmn] denotes the DG method preceding it.

DG bases & Mesh	p	$\frac{\mathcal{E}_H \text{ in [p}p\text{m}n\text{]}}{\mathcal{E}_H \text{ in [p}p\text{m1]}}$ on $\frac{h}{2^q}$-mesh				h-conv. rate
		h	$h/2$	$h/4$	$h/8$	
NDG quad [p1m1]	1	1.61e-02	4.14e-03	1.04e-03	2.61e-04	1.98
		1.00	1.00	1.00	1.00	1.98
NDG skewed-quad [p1m2]	1	1.12	1.12	1.12	1.12	1.99
PNDG quad [p1m3]	1	1.16	1.13	1.12	1.12	2.00
PNDG tri [p1m4]	1	1.14	1.14	1.14	1.14	1.98
NDG tri [p1m5]	1	1.14	1.14	1.14	1.14	1.98
PNDG skewed-quad [p1m6]	1	1.29	1.26	1.25	1.25	2.00
NDG quad [p2m1]	2	3.29e-04	3.96e-05	4.91e-06	6.16e-07	3.02
		1.00	1.00	1.00	1.00	3.02
NDG skewed-quad [p2m2]	2	1.49	1.46	1.44	1.43	3.04
PNDG tri [p2m3]	2	2.12	2.01	1.97	1.97	3.05
NDG tri [p2m4]	2	2.12	2.01	1.97	1.97	3.05
PNDG quad [p2m5]	2	3.52	3.38	3.32	3.28	3.05
PNDG skewed-quad [p2m6]	2	4.59	4.33	4.20	4.13	3.07
NDG quad [p3m1]	3	1.73e-05	1.03e-06	6.43e-08	3.94e-09	4.03
		1.00	1.00	1.00	1.00	4.03
NDG skewed-quad [p3m2]	3	1.72	1.75	1.75	1.74	4.02
PNDG tri [p3m3]	3	3.05	3.26	3.23	3.28	4.00
NDG tri [p3m4]	3	3.05	3.26	3.23	3.28	4.00
PNDG quad [p3m5]	3	6.09	6.50	6.50	6.64	3.99
PNDG skewed-quad [p3m6]	3	7.98	8.99	9.12	9.35	3.96
NDG quad [p4m1]	4	8.14e-07	2.56e-08	8.32e-10	2.77e-11	4.95
		1.00	1.00	1.00	1.00	4.95
NDG skewed-quad [p4m2]	4	2.22	2.14	2.10	2.07	4.98
PNDG tri [p4m3]	4	4.69	4.73	4.62	4.43	4.98
NDG tri [p4m4]	4	4.75	4.84	4.73	4.54	4.97
PNDG quad [p4m5]	4	10.74	10.68	10.33	9.70	5.00
PNDG skewed-quad [p4m6]	4	15.00	17.53	17.61	16.64	4.90
NDG quad [p5m1]	5	4.08e-08	6.37e-10	1.01e-11	9.34e-12*	5.99
		1.00	1.00	1.00	1.00*	5.99
NDG skewed-quad [p5m2]	5	2.63	2.67	2.60	0.93*	6.00
NDG tri [p5m3]	5	7.97	7.92	7.75	0.89*	6.01
PNDG tri [p5m4]	5	8.12	8.03	7.96	0.43*	6.00
PNDG quad [p5m5]	5	23.48	24.31	24.60	1.46*	5.95
PNDG skewed-quad [p5m6]	5	43.48	49.74	50.72	0.93	5.88

* The temporal errors are not negligible in comparison to the spatial errors.

detail, we note that the convergence rates of uH and vH are between $O(h^p)$ and $O(h^{p+1})$. The convergence of the schemes on rectangles and triangles appear to behave somewhat irregularly; the convergence rates of these schemes are typically close to the expected rate $O(h^{p+1/2})$ for even p and close to $O(h^{p+1})$ for odd p. This somewhat irregular behavior appears less pronounced in the schemes on skewed rectangles with the numerical order of convergence being typically close to $p+1$ for both odd and even p.

4.2.1.2. Computing times. Table 3 tabulates computing times (in seconds), denoted as T_c, required in the simulations. Note that data reported are an average of three identical simulations (except for the schemes with $p = 5$ and $h/8$-mesh combination where they are the results from two runs). In this table, data is grouped according to the interpolation order p used. Within each data group, the computing times of the scheme using the least computing time are listed and highlighted; the computing times of the other schemes are tabulated as the ratio of the computing time of a specific scheme relative to the computing time of the fastest scheme. In every scheme, while holding the mesh resolution unchanged, the computing time required increases as the interpolation order p of the scheme increases. Such increases in computing times stem primarily from the following two reasons. First, the degrees of freedom per element increase as p increases. Second, the time

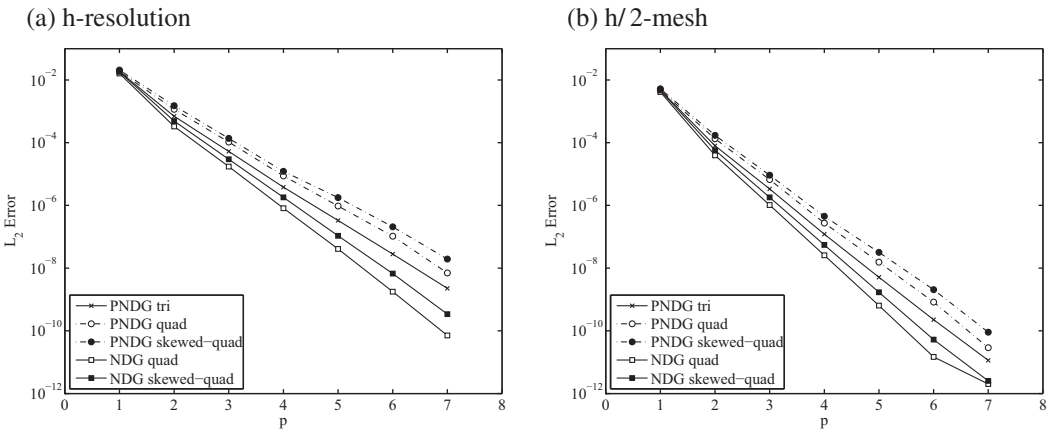


Fig. 7. Normalized L_2-error $|\Omega|^{-1/2}\|H - H_h\|_{\Omega_h}$ at $t = 2$ days as a function of order of bases p. (a) h-meshes; (b) $h/2$-meshes.

step size Δt used is smaller as p increases in order to keep the temporal accuracy sufficiently small and, as an explicit time scheme is used, to maintain numerical stability. This aspect is implicitly reflected by numerical data listed in Table 4 which shows an increase in $\mathcal{N}_{RHS}$ (*i.e.*, decrease in Δt) as p increases. Note that $\mathcal{N}_{RHS}$ denotes the total number of calls made within the RKF45 integrator to a subroutine calculating the right hand side of the ODEs (33). Likewise, while fixing p, the computing times required increases as the mesh is refined. The increasing computing time is the direct consequence of an increase in the number of elements (hence the total DOFs). Furthermore, as the mesh size decreases, the time step size Δt used is smaller in order to maintain temporal accuracy and to ensure stability; this aspect can be discerned in an increasing $\mathcal{N}_{RHS}$ as the element size decreases (see Table 4).

It can be observed from Table 3 that, in the calculations based on quadrilateral elements, the PNDG scheme requires less computing time (approximately between 1.4 and 2.4 times) than the NDG scheme. This behavior is to be expected since, on quadrilateral elements, the DOFs per element of the PNDG scheme are less than that of the NDG scheme for all p. Furthermore, it can be noticed in Table 4 that, in the quadrilateral-element calculations, $\mathcal{N}_{RHS}$ required in the PNDG scheme are also fewer than that of the NDG scheme; this results in an additional reduction in computing time for the PNDG scheme in comparison to the NDG scheme. Table 3 shows that the PNDG scheme on quadrilaterals is faster than the PNDG scheme and NDG scheme on triangles. This is an expected behavior and stems directly from the fact that the total number of nodes in the PNDG scheme on quadrilaterals is noticeably (approximately 35%) lesser than that of the PNDG and NDG schemes on triangles. It can be noticed from Table 3 that the PNDG scheme on triangular is faster than the NDG scheme on triangles. We note that this lower computing in the PNDG scheme is a result of the RKF45 time integrator automatically selecting larger time step sizes Δt for the PNDG scheme (this reflects in a fewer calls to the subroutine evaluating the RHS vector–see Table 4). The NDG scheme on quadrilaterals, due to the use of tensor product bases, has higher DOFs per element than that of the PNDG and NDG scheme on triangles, more precisely, $2 - 2/(p+2)$ times higher DOFs per element. The cost per element in evaluating one volume integral in the NDG scheme on quadrilaterals is approximately $4 - 4(2p+3)/(p+2)^2$ times higher than the NDG and PNDG scheme on triangles. Hence, it can be expected that the reduction of the number of elements associated with the rectangular mesh might offset the higher cost of using tensor-product bases only up to a certain interpolation order p. A crude estimate made in the previous work [30] shows that the cost of evaluating the RHS vector in the NDG scheme on quadrilaterals is expected to be greater than that of the NDG or PNDG scheme on triangles for $p > 1$. Although not shown here in detail, we note that the value p at which the cost of evaluating RHS vector in the NDG scheme on quadrilaterals becomes more expensive is noticeably higher than the estimate. We speculate that the efficiency of memory traffic and cache management are partial reasons explaining why this occurs at p higher than the estimate. In terms of wall clock time, Table 4 shows that the NDG scheme on quadrilateral becomes slower than the PNDG scheme on triangles when $p > 4$; the NDG scheme on quadrilateral is faster the NDG on triangles for all p considered here ($p = 1$ to 5).

It can be verified from data in Table 3 that, for a fixed interpolation order p and varying h, the computing time T_c behaves approximately like ch^s, where c and s are constant, in other words

$$T_c \sim O(h^s). \tag{38}$$

The numerical rates s are tabulated in the last column of Table 3. Notice that the differences between the rate s are relatively small (the value of s ranges from -2.7 to -2.8.) and the rates appear to be independent of the interpolation order p. The values of the constant c, as expected, vary for the different DG schemes as well as the interpolation order p.

Table 3
Computing times T_c (in seconds) of the overall fastest DG scheme for a given order p and time ratios (specific scheme relative to the fastest scheme for that p), as computed on regular meshes. A code [ppmn] denotes the DG method preceding it. s denotes the rate of computing times as function of h, *i.e.*, $T_c \sim O(h^s)$.

DG Bases& Mesh	p	$\frac{T_c \text{ of [p}p\text{m}n\text{]}}{T_c \text{ of [p}p\text{m1]}}$ on $\frac{h}{2^q}$-mesh				s-rate
		h	$h/2$	$h/4$	$h/8$	
PNDG quad [p1m1]	1	124	827	6068	41151	-2.80
		1.00	1.00	1.00	1.00	-2.80
PNDG skewed-quad [p1m2]	1	1.02	1.07	0.97	1.01	-2.78
NDG skewed-quad [p1m3]	1	1.43	1.40	1.30	1.32	-2.75
NDG quad [p1m4]	1	1.35	1.43	1.32	1.36	-2.79
PNDG tri [p1m5]	1	2.28	2.79	2.36	2.28	-2.77
NDG tri [p1m6]	1	2.91	2.83	2.62	2.53	-2.73
PNDG skewed-quad [p2m1]	2	238	1755	11550	76385	-2.77
		1.00	1.00	1.00	1.00	-2.77
PNDG quad [p2m2]	2	1.02	0.95	1.06	1.00	-2.78
NDG skewed-quad [p2m3]	2	1.73	1.54	1.48	1.51	-2.70
NDG quad [p2m4]	2	1.78	1.62	1.54	1.53	-2.70
PNDG tri [p2m5]	2	2.00	2.20	2.17	2.10	-2.79
NDG tri [p2m6]	2	3.03	2.65	2.70	2.71	-2.72
PNDG quad [p3m1]	3	411	2727	19791	126896	-2.77
		1.00	1.00	1.00	1.00	-2.77
PNDG skewed-quad [p3m2]	3	0.99	1.06	0.96	1.04	-2.78
NDG skewed-quad [p3m3]	3	1.88	1.81	1.67	1.81	-2.74
NDG quad [p3m4]	3	1.97	1.95	1.79	1.78	-2.71
PNDG tri [p3m5]	3	1.84	2.19	1.96	2.01	-2.79
NDG tri [p3m6]	3	2.80	2.70	2.48	2.58	-2.72
PNDG quad [p4m1]	4	688	4424	28808	202288	-2.73
		1.00	1.00	1.00	1.00	-2.73
PNDG skewed-quad [p4m2]	4	1.03	1.08	1.07	1.24	-2.81
PNDG tri [p4m3]	4	1.71	2.10	2.08	1.93	-2.78
NDG skewed-quad [p4m4]	4	2.09	2.08	2.02	2.22	-2.75
NDG quad [p4m5]	4	2.17	2.17	2.20	1.91	-2.68
NDG tri [p4m6]	4	2.92	2.98	3.06	2.76	-2.71
PNDG quad [p5m1]	5	1070	6905	43926	319869	-2.68
		1.00	1.00	1.00	1.00	-2.68
PNDG skewed-quad [p5m2]	5	1.03	1.09	1.14	1.47	-2.75
PNDG tri [p5m3]	5	1.66	2.00	1.75	2.08	-2.72
NDG skewed-quad [p5m4]	5	2.29	2.32	2.28	2.93	-2.67
NDG quad [p5m5]	5	2.40	2.41	2.42	2.10	-2.69
NDG tri [p5m6]	5	2.75	2.72	2.66	2.52	-2.66

4.2.1.3. Computational cost per accuracy. The critical question when comparing numerical techniques is the computational cost for a specific level of accuracy, or conversely, an error level to be achieved for a given computational cost. Figs. 8 shows on a log–log scale the accuracy of H_h through normalized L_2 errors versus the computing time. In this figure each curve represents the data computed on the four refined meshes with the interpolation order p being held constant. Figure legends indicate the combination of DG basis, mesh configuration, and interpolation order p from which the data are obtained. In each figure, we plot the data from the PNDG scheme on triangles for inter-comparison purposes. It can be observed that all the curves appear approximately as straight lines on a log–log scale. Therefore, the computational time as a function of accuracy in the total water column height H_h can be approximated by

$$T_c \sim c_2(\mathcal{E}_H)^{s_2} \tag{39}$$

where c_2 and s_2 are respectively the constant and the rate of the cost function. The discussions above on accuracy and computing times implies that

$$s_2 \approx \frac{2.7}{-(p+1)}. \tag{40}$$

Table 4
The number of calls to a subroutine computing RHS vector required in RKF45/PNDG and RKF45/NDG methods on regular meshes.

p	h	$h/2$	$h/4$	$h/8$	h	$h/2$	$h/4$	$h/8$
	PNDG tri				NDG tri			
1	15,853	25,117	39,325	61,759	19,144	30,079	47,743	76,507
2	21,565	34,807	54,193	84,075	30,475	48,547	75,679	118,123
3	27,067	42,457	66,109	104,439	39,733	61,609	95,995	156,349
4	32,053	50,293	78,765	130,015	53,581	84,673	132,133	207,202
5	36,823	57,937	93,001	171,510	62,455	96,691	151,249	248,853
	PNDG quad				NDG quad			
1	11,431	17,611	28,477	45,667	13,665	22,708	36,832	57,985
2	17,191	27,601	42,967	66,769	25,739	39,614	60,774	93,475
3	22,405	34,927	54,367	84,859	36,515	55,881	85,801	132,615
4	26,875	42,043	65,677	103,247	47,704	73,195	112,633	175,368
5	31,255	48,997	76,807	122,707	59,581	91,519	141,325	221,995

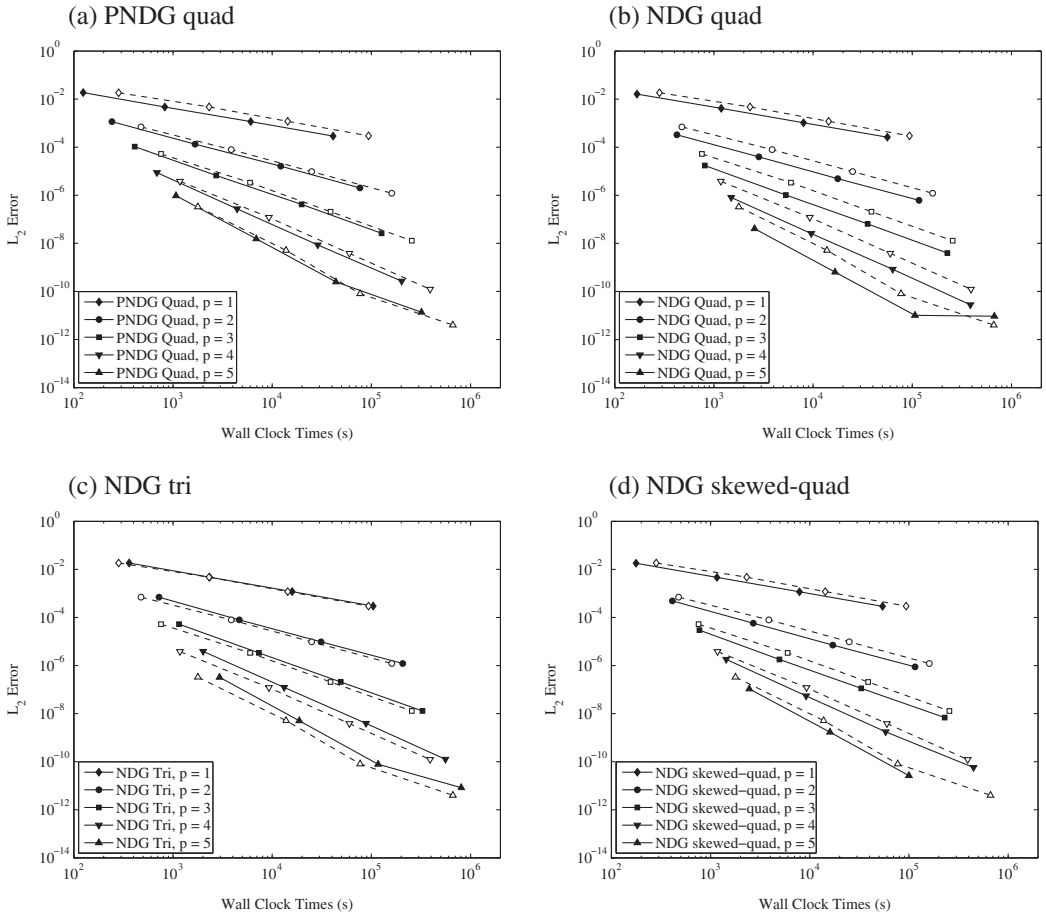


Fig. 8. Normalized errors $|\Omega|^{-1/2}\|H - H_h\|_\Omega$ at $t_f = 172800$ *vs.* computing times in seconds of DG solutions on regular grids. Solid lines represent the data of (a) PNDG on rectangles, (b) NDG on rectangles, (c) NDG on triangles, and (d) NDG on skewed-rectangles. Dashed lines in (a–d) represent the data of PNDG on triangles: $-\blacklozenge- \ p = 1, -\circ- \ p = 2, -\square- \ p = 3, -\triangledown- \ p = 4$, and $-\triangle- \ p = 5$.

The constant pairs (c_2, s_2) for the cost functions associated with the DG schemes considered are tabulated in Table 5.

It can be noticed from Figs. 8 that, for a given level of accuracy, the wall clock time decreases substantially as p increases. To gain more insight into the effect of p on a cost per accuracy viewpoint, we evaluate the computing time for a specified level of error ε from the derived cost functions, *i.e.*, finding T_c by using (39) with $\mathcal{E}_h = \varepsilon$. Table 6 tabulates the computing times required in each DG scheme with various orders p to yield a numerical solution with the specified levels of error ε. The value inside the parenthesis denotes the cost ratio of the computational cost for the identical error level using $p-1$ to that using p order interpolants. Note that such a value indicates a reduction in cost when raising the interpolation order

Table 5
Constant and rate (c_2, s_2) in the cost functions $T_c = c_2(\mathcal{E}_H)^{s_2}$ of DG schemes on regular grids.

DG bases and mesh	Cost coefficients (c_2, s_2)				
	$p=1$	$p=2$	$p=3$	$p=4$	$p=5$
NDG tri	(1.49, −1.37)	(1.07, −0.89)	(1.40, −0.68)	(2.27, −0.54)	(4.01, −0.44)
PNDG tri	(1.14, −1.40)	(0.65, −0.91)	(0.82, −0.70)	(1.15, −0.56)	(2.15, −0.45)
PNDG skewed-quad	(0.58, −1.39)	(0.69, −0.90)	(0.81, −0.70)	(1.08, −0.57)	(2.25, −0.47)
PNDG quad	(0.47, −1.40)	(0.51, −0.91)	(0.72, −0.69)	(1.16, −0.55)	(2.10, −0.45)
NDG skewed-quad	(0.68, −1.39)	(0.46, −0.89)	(0.63, −0.68)	(0.91, −0.55)	(1.93, −0.45)
NDG quad	(0.51, −1.41)	(0.33, −0.89)	(0.50, −0.67)	(0.76, −0.54)	(1.25, −0.45)

Table 6
Computing time, T_c^ε (in seconds) for a given level of error ε in H_h of various DG solutions on regular grids. Numeric values in the parenthesis are the ratio between T_c^ε of a DG scheme order $p-1$ and that of p.

DG bases and mesh	p	Projected computing time T_c^ε			
		ε = 5.0e−03	ε = 1.0e−04	ε = 1.0e−06	ε = 1.0e−08
PNDG tri	1	1898	452,923	2.85e+08	1.80e+11
	2	82(23.1)	2934(154.4)	197228(1446.1)	1.33e+07(13546.9)
	3	33(2.5)	512(5.7)	12757(15.5)	317901(41.7)
	4	22(1.5)	200(2.6)	2629(4.9)	34605(9.2)
	5	24(0.9)	140(1.4)	1133(2.3)	9154(3.8)
NDG tri	1	2173	470,487	2.64e+08	1.48e+11
	2	121(18.0)	3950(119.1)	239892(1101.1)	1.46e+07(10177.3)
	3	51(2.3)	736(5.4)	16901(14.2)	387849(37.6)
	4	41(1.3)	344(2.1)	4230(4.0)	52042(7.5)
	5	42(1.0)	235(1.5)	1799(2.4)	13768(3.8)
PNDG skewed-quad	1	924	213,139	1.29e+08	7.80e+10
	2	82(11.3)	2796(76.2)	178196(723.4)	1.14e+07(6863.8)
	3	33(2.4)	521(5.4)	13178(13.5)	333343(34.1)
	4	22(1.5)	211(2.5)	2953(4.5)	41296(8.1)
	5	27(0.8)	168(1.3)	1455(2.0)	12580(3.3)
PNDG quad	1	777	185,355	1.17e+08	7.35e+10
	2	64(12.2)	2241(82.7)	148117(787.7)	9789892(7502.1)
	3	28(2.3)	426(5.3)	10367(14.3)	252525(38.8)
	4	21(1.3)	178(2.4)	2207(4.7)	27334(9.2)
	5	23(0.9)	133(1.3)	1053(2.1)	8361(3.3)
NDG skewed-quad	1	1048	236797	1.39e+08	8.25e+10
	2	51(20.5)	1667(142.1)	100492(1390.9)	6059758(13615.6)
	3	23(2.2)	330(5.0)	7576(13.3)	173840(34.9)
	4	17(1.4)	149(2.2)	1895(4.0)	24177(7.2)
	5	21(0.8)	117(1.3)	915(2.1)	7129(3.4)
NDG quad	1	886	217,372	1.41e+08	9.20e+10
	2	37(23.8)	1227(177.1)	75121(1882.6)	4598333(20009.8)
	3	18(2.1)	247(5.0)	5487(13.7)	121693(37.8)
	4	13(1.3)	111(2.2)	1336(4.1)	16131(7.5)
	5	13(1.0)	78(1.4)	612(2.2)	4830(3.3)

from $p-1$ to p (for example, with $\varepsilon = 1.0 \times 10^{-4}$, the cost required in the NDG scheme on triangular elements reduces approximately 120 times when raising p from 1 to 2). Results shown in this table clearly indicate the appeal of using higher order schemes from the perspective of cost per accuracy. As an example, suppose that an accuracy of 10^{-6} is required, the use of schemes with $p=1$ would require approximately on the order of 3 years of computing time, a prohibitively impractical cost (this corresponds to an expected cost on the serial machine; a dramatically lower wall clock time can be achieved by utilizing a parallel implementation). By using schemes with $p=2$, the computing times required are approximately on the order of 1 to 2 days. Note that computing time decreases approximately three orders of magnitude. The schemes with $p=3$ requires approximately on the order of 1 to 2 h of computing time. Note the cost reduces roughly four orders of magnitude compared to the schemes with $p=1$ and approximately an order magnitude compared to the schemes with $p=2$. The computing times required reduce further as the interpolation order p increases. It is evident from Table 6 that the smaller the specified error level ε, the more pronounced the gain in computational cost per accuracy achieved by raising the interpolation order p of the scheme. Although the computational cost for a given level of accuracy reduces as the interpolation order p increases, the benefit diminishes as indicated by the reduction in the cost ratios inside parentheses shown in Table 6. Arguably, although the scheme with $p=2$ shows the highest gain in terms of the cost reduction in comparison to the scheme

Table 7
Manufactured-solution problem on regular grids: computing time $T_c^{p,\varepsilon}$ for a specified level of error ε in H_h of the PNDG triangular solution for a given p and time ratios (given schemes relative to the PNDG scheme on triangles for that p). A code [mn] denotes the DG scheme preceding it.

DG bases & mesh	$T_c^{p,\varepsilon}$ of [mn]/$T_c^{p,\varepsilon}$ of [m1]				
	$p = 1$	$p = 2$	$p = 3$	$p = 4$	$p = 5$
	ε = 2.0e-03	ε = 3.0e-05	ε = 9.0e-07	ε = 2.0e-08	ε = 7.0e-09
PNDG tri [m1]	6841	8816	13731	23478	10762
	1.00	1.00	1.00	1.00	1.00
NDG tri [m2]	1.12	1.31	1.32	1.52	1.50
PNDG skewed-quad [m3]	0.48	0.94	1.03	1.18	1.38
PNDG quad [m4]	0.41	0.76	0.81	0.80	0.91
NDG skewed-quad [m5]	0.55	0.55	0.59	0.70	0.78
NDG quad [m6]	0.47	0.41	0.43	0.47	0.53

with $p-1$, using the schemes with $p = 3$ appears to be an appealing choice due to an evident significant performance gain over using $p = 1$ while showing moderate gains when compared to the schemes with $p = 2$.

Table 7 shows the effect of the different combinations of DG bases and mesh configurations on the cost per accuracy performance. In this table, the corresponding computing cost for the given levels of accuracy of the PNDG scheme on triangles are highlighted. The computing costs of other combinations of DG bases and mesh configurations are reported as a ratio of the computing time for the specific scheme to the computing time for the PNDG scheme on triangles for the same interpolation order p (the higher the ratio, the higher the computational cost required to achieve a specified level of accuracy in comparison to that of the PNDG scheme on triangles). It can be seen from this table that the NDG scheme on rectangles exhibits the highest cost per accuracy performance among the combination of bases and mesh configurations considered. We note the performance gain achieved with nodal quadrilateral elements is not as pronounced in comparison to the gain realized using the high order schemes.

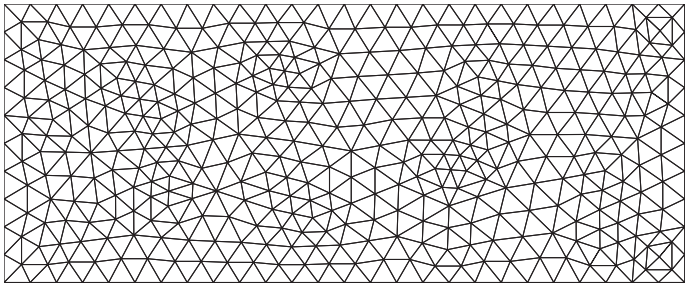
The numerical results discussed above and in the previous sections demonstrate the appeal of the use of tensor product bases on quadrilaterals, from both accuracy and cost per accuracy perspectives. Note that nodal tensor-product basis can represent more cross polynomial terms than the bases on triangles; thus it can be expected in general that, for a problem with a smooth solution, the approximate solution from the nodal tensor-product elements would have higher or approximately the same level of accuracy as those from the bases on triangles. This expectation together with the presented numerical results leads us to believe that the use of methods with nodal tensor-product bases is particularly appealing for the low to moderate interpolation order p since higher efficiency in terms of cost per accuracy is likely be achieved. Note also that although they may not be particularly appealing in terms of cost per accuracy, the schemes based on the polymorphic bases on quadrilaterals show superiority in terms of the computing times required to reach the final solution. This makes such the scheme appealing in a scenario where the computational time available is limited.

We have also examined a similar performance analysis based on the L_∞ error. Although not reported in detail here, we note that the results exhibit similar behavior to that based on the L_2 error described above.

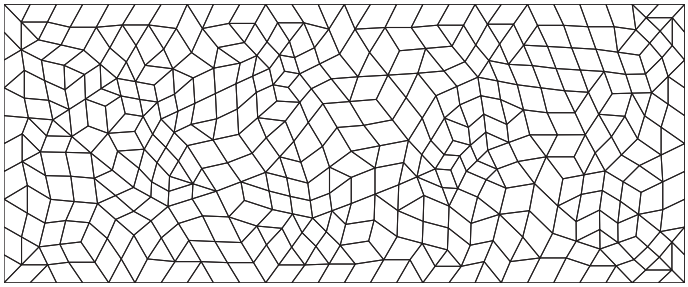
4.2.2. Solution computed on unstructured meshes

Next we consider DG solutions on unstructured meshes with various elements and configurations, namely, an unstructured triangular mesh, a quadrilateral mesh, a mixed triangular-quadrilateral mesh, and a polygonal mesh. In each configurations, we employ meshes of varying levels of resolution. They are denoted, from the coarsest to finest, $h, h/2, h/4$, and $h/8$, respectively. Fig. 9 shows the h-mesh for each configuration. The triangular h-mesh consists of 792 triangular elements with the element edges of length at most equal to 4500. The finer triangular meshes are obtained by applying successive regular refinements; see Table 8(a) for the number of triangles in each triangular mesh. We obtain other mesh configurations from the triangular meshes. The mixed triangular-quadrilateral mesh is built naively by simply merging pairs of two adjacent triangles in the triangular mesh into quadrilaterals. The merging process is conducted in such a way that every resulting quadrilateral element has a determinate Jacobian. In other words, we do not merge two triangles forming a quadrilateral with interior angles equal to or greater than 180°. As is seen in Fig. 9(b), the resulting mixed meshes contain triangular elements scattered over the computational domain. Table 8(b) lists the number of triangular and quadrilateral elements in each mixed mesh. For the polygonal mesh, an element is formed by first collecting a set of all triangles sharing a vertex and subsequently connecting a line between the centroids of any two elements in such a set having a common edge. In this way, the number of sides of the resulting polygon corresponds to the number of triangles sharing the vertex. The total number of polygons in the resulting mesh therefore equals the total number of vertices in the given triangular mesh. Note that any triangulation of a given set of n points yields $2n-2-k$ triangles [35] where k is the number of points lying on the boundary of the convex hull of the considered set. Therefore, for a triangular mesh with the number of vertices in the interior far greater than the number lying on the boundary, the number of elements in the resulting polygonal mesh would be fewer than that in the considered triangular mesh. Table 8(c) tabulates the number of elements classified by shapes in each resulting polygonal mesh. For the

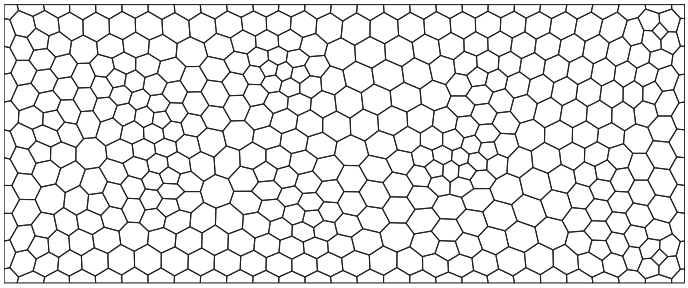
(a) Triangular mesh, N_{el} = 792



(b) Mixed triangular-quadrilateral mesh, N_{el} = 457



(c) Polygonal mesh, N_{el} = 433



(d) quadrilateral mesh, N_{el} = 594

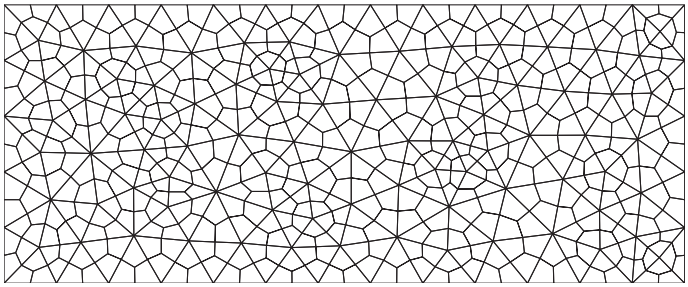


Fig. 9. Unstructured *h*-meshes employed in DG solution to the SWE with a manufactured solution; (a) triangular mesh, (b) quadrilateral mesh, (c) mixed triangular-quadrilateral mesh, and (d) polygonal mesh.

same so-called resolution, the total number of elements in the polygonal mesh is less than that of its associated triangular mesh. A quadrilateral mesh is built from a given triangular mesh by using an approach employed in [36], more precisely, by placing a point at the centroid of each triangle and forming quadrilateral elements by connecting this point to the mid points of the element edge. This strategy divides each triangle into three quadrilaterals. The mesh-size resolution of the derived quadrilateral mesh is comparable to that of a triangular mesh resulting from applying regular refinement to the given trian-

Table 8
Number of elements categorized by shapes in computational meshes; (a) triangular meshes, (b) mixed triangular-quadrilateral meshes, and (c) polygonal meshes.

Mesh res.						Tri
(a) Triangular meshes						
h						792
$h/2$						3168
$h/4$						12,672
$h/8$						50,688
(b) Mixed tri/quad meshes						
Mesh res.		Tri		Quad		Total
h		122		335		457
$h/2$		354		1407		1761
$h/4$		870		5901		6771
$h/8$		1902		24,393		26,295
(c) Polygonal meshes						
Mesh res.	Quad	Pentagon	Hexagon	Heptagon	Octagon	Total
h	2	88	327	14	2	433
$h/2$	2	160	1479	14	2	1657
$h/4$	2	304	6159	14	2	6481
$h/8$	2	592	25,023	14	2	25,633

gular element. The quadrilateral mesh at the so-called $h/2^j$-resolution is hence defined from the $h/2^{j-1}$-triangular mesh. Note that, at the same mesh resolution, the number of elements in the quadrilateral mesh is 3/4 of that in the triangular mesh.

In the numerical calculations, the parameters in the RKF45 time integrator are set to $\varepsilon_r = 1 \times 10^{-6}$ and $\varepsilon_a = 1 \times 10^{-9}$. The integration is carried out until reaching $t_f = 97200$. For the PNDG scheme, the polymorphic bases with $p_d = 1$ are utilized for quadrilateral elements. For polygonal meshes, we consider a strategy, employed by Gassner et al. [15] to solve the compressible Navier–Stokes equations, in defining a set of nodal points for the polymorphic bases. This strategy uses (22) with r_{max} (instead of p_d) as a free parameter in defining the nodal set of the elements whose the nodal sets obtained using $p_d = 1$ contain less than or equal to a single nodal point in the interior. More specifically, for such elements, their associated nodal sets are defined as those obtained by adjusting the parameter p_d in (23) so that $r_{\max} = 1$ and in addition $p - (N_{\text{gon}} - p_d) > 0$ for $p \geqslant 3$. This strategy ensures the existence of interior nodes for all elements. We find that this strategy yields noticeably more accurate approximate solutions than the strategy using the fixed value $p_d = 1$ (at least two times more accurate in the L_2-norm for $p \geqslant 3$).

Table 9 tabulates the normalized L_2 errors in the total water column height H_h at the final time of simulation t_f. As presented in the previous section, data are grouped according to p. In each group, we highlight the combination of DG basis and the mesh configuration that overall yields the most accurate solutions. The results for other combinations are tabulated as the ratio of the error from the specific scheme to the error associated with the most accurate scheme. The last column in this table reports the numerical order of convergence of each DG scheme. All DG schemes, regardless of bases or mesh configurations, converge approximately at the rate of $O(h^{p+1})$ for the total water column height H_h. It can be observed that the NDG scheme on quadrilateral meshes yields the most accurate solution among the combinations of bases and mesh configurations. The PNDG scheme on mixed meshes are less accurate than the other schemes. The data from the calculation on the mixed meshes indicates, as expected, that the less accurate element type dictates the error levels. More precisely, it can be observed that the PNDG solution on mixed meshes is less accurate in comparison to the PNDG solution on triangular meshes; their error ratios drift further apart as p increases. This clearly reflects the effect of using the less accurate quadrilateral polymorphic elements. On similar resolution, the NDG scheme on a mixed mesh, within the range of p tested, yields a less accurate solution than the schemes on triangular meshes; however, their accuracies overall appear to be closer as p increases. This indicates that the error levels in the NDG solutions on mixed meshes is strongly dictated by the presence of triangular nodal elements.

Fig. 10(a)–(f) shows on a log–log scale, the accuracy of H_h through the L_2 error versus the computing times required in the simulations. Each curve represents data from the DG solution with a given p on various mesh resolutions. See the legends accompanying the plot for the combinations of DG basis, mesh configuration, and interpolation degree p with which the curves are associated. Additionally, in each figure, the data from the PNDG scheme on triangles is plotted for comparison purposes. As the curves appear as straight lines, the cost functions are well approximated by $T_c = c_2(\mathcal{E}_H)^{s_2}$. Table 10 tabulates the constant pairs (c_2, s_2) of the cost function associated with the DG schemes tested. Roughly speaking, the computational costs of the DG schemes are proportional approximately to $\mathcal{E}_H^{-2.7/(p+1)}$.

To examine the effect of p from a cost per accuracy perspective, we evaluate from the derived cost functions the computing cost required to achieve the specified error levels ε. Table 11 tabulates these data for each DG solution on unstructured meshes. Note that the number inside the parenthesis corresponds to the computational cost ratio of the estimated runtime of the $(p-1)$ scheme to that of the p scheme for the identical accuracy level. In other words, it reflects the gain in cost efficiency

Table 9
DG solutions on unstructured meshes. Normalized L_2 errors in $H_h, \mathcal{E}_H \equiv |\Omega|^{-1/2}\|H - H_h\|_{\Omega_h}$ of the overall most accurate scheme for a given order p, error ratios (specific scheme relative to the most accurate for that p), and rate of h-convergence. A code [ppmn] denotes the DG method preceding it.

DG bases & mesh	p	$\frac{\mathcal{E}_H \text{ in [p}p\text{m}n\text{]}}{\mathcal{E}_H \text{ in [p}p\text{m1]}}$ on $\frac{h}{2^q}$-mesh				h-conv. rate
		h	$h/2$	$h/4$	$h/8$	
NDG quad [p1m1]	1	1.02e-03	2.12e-04	4.91e-05	1.21e-05	2.13
		1.00	1.00	1.00	1.00	2.13
PNDG quad [p1m2]	1	0.90	1.12	1.25	1.28	1.96
PNDG ngon [p1m3]	1	1.09	1.24	1.30	1.30	2.08
NDG tri [p1m4]	1	1.02	1.27	1.38	1.40	1.98
PNDG tri [p1m5]	1	1.02	1.27	1.38	1.40	1.98
PNDG mixed [p1m6]	1	1.40	1.47	1.56	1.67	2.10
NDG mixed [p1m7]	1	1.43	1.47	1.57	1.67	2.11
NDG quad [p2m1]	2	3.56e-05	3.23e-06	3.85e-07	4.42e-08	3.20
		1.00	1.00	1.00	1.00	3.20
NDG tri [p2m2]	2	1.18	1.28	1.30	1.52	3.09
PNDG tri [p2m3]	2	1.18	1.28	1.30	1.52	3.09
NDG mixed [p2m4]	2	2.33	2.21	2.10	2.44	3.28
PNDG quad [p2m5]	2	2.87	2.88	2.29	2.02	3.39
PNDG ngon [p2m6]	2	1.87	2.66	2.88	3.19	3.02
PNDG mixed [p2m7]	2	3.84	4.06	3.60	3.55	3.34
NDG quad [p3m1]	3	1.95e-06	1.10e-07	6.13e-09	3.02e-10	4.21
		1.00	1.00	1.00	1.00	4.21
NDG tri [p3m2]	3	1.45	1.50	1.54	1.90	4.09
PNDG tri [p3m3]	3	1.45	1.50	1.54	1.91	4.09
PNDG quad [p3m4]	3	2.00	1.89	1.80	2.08	4.20
NDG mixed [p3m5]	3	2.08	2.13	2.07	2.37	4.27
PNDG ngon [p3m6]	3	1.89	2.16	2.53	3.27	4.03
PNDG mixed [p3m7]	3	8.07	8.35	9.71	12.65	4.10
NDG quad [p4m1]	4	8.00e-08	2.48e-09	7.92e-11		4.99
		1.00	1.00	1.00		4.99
NDG tri [p4m2]	4	1.36	1.34	1.64		4.85
PNDG tri [p4m3]	4	1.52	1.39	1.65		4.93
PNDG quad [p4m4]	4	2.18	1.78	1.75		5.15
NDG mixed [p4m5]	4	2.36	2.09	2.50		5.09
PNDG ngon [p4m6]	4	2.73	2.41	2.40		5.21
PNDG mixed [p4m7]	4	14.28	10.98	8.92		5.48

achieved by increasing the interpolation order by one. The data, which exhibits a similar trend to the DG solutions on regular meshes, clearly show the benefit of using the higher order schemes. More precisely, to achieve a specified level of accuracy, the computational cost required for the high order scheme is considerably less than that required for the scheme with $p = 1$. As an example, for a specified accuracy of $\varepsilon = 1.0 \times 10^{-5}$ or 1.0×10^{-7}, the computational cost of the schemes with $p = 3$ are typically three to four orders of magnitude lower than the schemes with $p = 1$. The computational cost for a specified level of error decreases as the interpolation order p used in the scheme increases; however, the benefit gain from raising the interpolation order p eventually diminishes as indicated by the reduction in the cost ratios. Although the scheme with $p = 2$ exhibits the highest cost reduction from the perspective of comparing the cost required in the scheme with p to that required in the scheme with $p - 1$, the use of schemes with $p = 3$ appears, to some extent, to be more appealing in the sense that the scheme yields significant gains in performance over the scheme with $p = 1$ while still showing relatively large gains when compared to the scheme with $p = 2$.

Table 12 compares the cost for a given accuracy level from the different DG schemes. In this table, the results of the NDG scheme on triangles are highlighted in the gray box. The results of the other schemes are reported as a ratio of the estimated time of the specific scheme to that of the NDG scheme on triangles for the same interpolation order p (the higher the ratios, the higher the computational cost required to achieve a specified level of accuracy in comparison to the PNDG scheme on triangles). It is noticed that the PNDG scheme on mixed meshes, which is the fastest scheme, is less efficient than other schemes from a cost per accuracy performance perspective. This indicates that the gain in computing time achieved by introducing quadrilateral elements in the PNDG scheme is not enough to offset the loss of accuracy. The NDG scheme on mixed elements exhibits approximately the same cost performance as the NDG scheme on triangular elements for the ranges of interpolation order p tested. The NDG scheme on quadrilateral meshes, which yields the most accurate solution, performs

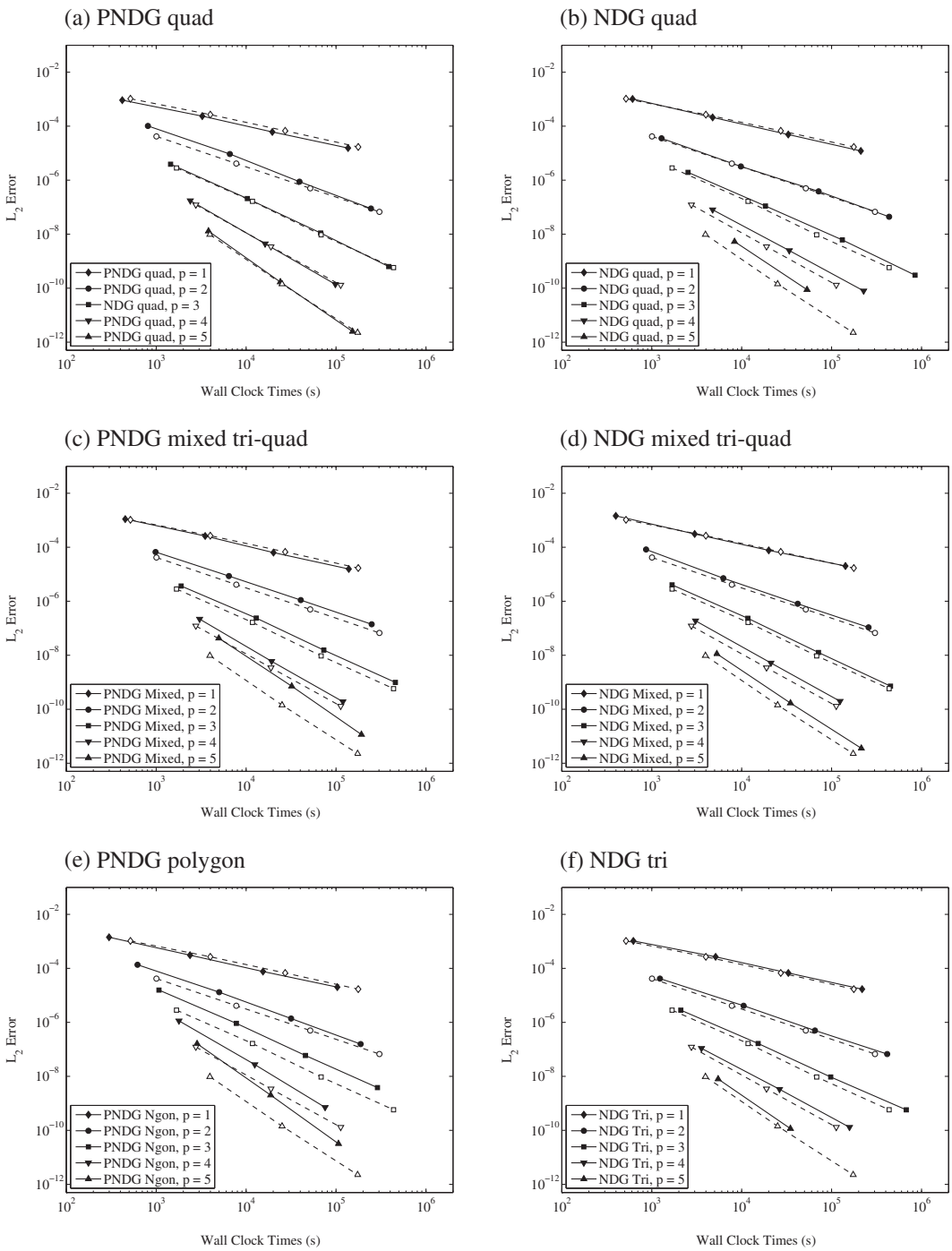


Fig. 10. Normalized errors $|\Omega|^{-1/2}\|H - H_h\|_\Omega$ at $t_f = 97200$ *vs.* computing times (in seconds) in DG solutions on unstructured meshes. Solid lines represent the data of (a) PNDG on quad meshes, (b) NDG on quad meshes, (c) PNDG on mixed tri–quad meshes, (d) NDG on mixed tri–quad meshes, (e) PNDG on polygon meshes, and (f) NDG on tri meshes. Dash lines in (a–d) represent the data of PNDG on triangles: $-\blacklozenge- \ p = 1, -\circ- \ p = 2, -\square- \ p = 3, -\triangledown- \ p = 4$, and $-\triangle- \ p = 5$.

slightly better than the NDG schemes on triangular meshes for $p \leqslant 3$. Note that, on the same mesh resolution, the wall clock times of the quadrilateral NDG scheme are higher than that of the triangular DG scheme for $p \geqslant 2$; for $p = 1$, the quadrilateral NDG scheme runs slightly faster than the triangular NDG scheme (this behavior reflects the fact that, for the considered mesh setting, the total DOFs of the quadrilateral NDG solution is higher than that of the triangular DG solution for $p \geqslant 2$). This suggests that the element size transition play a role in obtaining a full benefit of the tensor-product quadrilateral ele-

Table 10
DG solutions on unstructured meshes. Constant and rate (c_2, s_2) in the cost functions $T_c = c_2(\mathcal{E}_H)^{s_2}$.

DG bases & mesh	Cost coefficients (a,s)				
	$p=1$	$p=2$	$p=3$	$p=4$	$p=5$
NDG tri	(0.04, −1.42)	(0.14, −0.90)	(0.38, −0.68)	(0.43, −0.56)	(1.32, −0.45)
NDG quad	(0.07, −1.32)	(0.16, −0.88)	(0.42, −0.67)	(0.52, −0.56)	(1.11, −0.47)
NDG mixed	(0.05, −1.37)	(0.26, −0.86)	(0.58, −0.65)	(0.72, −0.54)	(1.18, −0.46)
PNDG tri	(0.03, −1.42)	(0.13, −0.89)	(0.43, −0.65)	(0.49, −0.54)	(0.91, −0.45)
PNDG quad	(0.02, −1.41)	(0.52, −0.81)	(0.52, −0.64)	(0.73, −0.52)	(1.51, −0.43)
PNDG mixed	(0.04, −1.37)	(0.36, −0.84)	(0.67, −0.67)	(1.78, −0.51)	(3.75, −0.42)
PNDG ngon	(0.05, −1.34)	(0.18, −0.90)	(0.48, −0.66)	(1.03, −0.52)	(2.69, −0.44)

Table 11
DG solutions on unstructured grids. Computing time T_c^ε (in seconds) required to achieve a given level of error ε in H_h. A numeric value in the parenthesis denotes the ratio between T_c^ε of a DG scheme with interpolation order $p-1$ and that of p.

DG bases & mesh	p	Computing time T_c^ε			
		ε = 5.0e−04	ε = 1.0e−05	ε = 1.0e−07	ε = 1.0e−09
NDG tri	1	1898	488,329	3.359e+08	2.311e+11
	2	134(14.1)	4548(107.4)	287597(1168.0)	1.819e+07(12704.5)
	3	64(2.1)	910(5.0)	20509(14.0)	462393(39.3)
	4	31(2.1)	283(3.2)	3788(5.4)	50687(9.1)
	5	39(0.8)	223(1.3)	1736(2.2)	13496(3.8)
NDG quad	1	1550	269,879	1.172e+08	5.092e+10
	2	124(12.5)	3848(70.1)	218479(536.6)	1.240e+07(4104.7)
	3	66(1.9)	890(4.3)	19022(11.5)	406775(30.5)
	4	36(1.8)	322(2.8)	4225(4.5)	55453(7.3)
	5	38(0.9)	238(1.4)	2040(2.1)	17464(3.2)
NDG mixed	1	1641	352,711	1.964e+08	1.093e+11
	2	175(9.4)	5019(70.3)	261323(751.4)	1.361e+07(8033.4)
	3	78(2.2)	975(5.1)	19015(13.7)	370794(36.7)
	4	44(1.8)	360(2.7)	4312(4.4)	51679(7.2)
	5	37(1.2)	225(1.6)	1875(2.3)	15628(3.3)
PNDG quad	1	1040	256,843	1.683e+08	1.103e+11
	2	239(4.4)	5603(45.8)	230058(731.6)	9445666(11677.9)
	3	68(3.5)	826(6.8)	15740(14.6)	299839(31.5)
	4	38(1.8)	292(2.8)	3202(4.9)	35154(8.5)
	5	41(0.9)	219(1.3)	1573(2.0)	11289(3.1)
PNDG mixed	1	1237	262,204	1.436e+08	7.866e+10
	2	218(5.7)	5886(44.5)	284620(504.6)	1.376e+07(5715.3)
	3	109(2.0)	1494(3.9)	32705(8.7)	716092(19.2)
	4	84(1.3)	606(2.5)	6252(5.2)	64476(11.1)
	5	85(1.0)	467(1.3)	3459(1.8)	25637(2.5)
NDG polygon	1	1356	252,596	1.188e+08	5.584e+10
	2	164(8.3)	5511(45.8)	345118(344.1)	2.161e+07(2583.7)
	3	75(2.2)	1012(5.4)	21551(16.0)	458959(47.1)
	4	54(1.4)	411(2.5)	4523(4.8)	49727(9.2)
	5	53(1.0)	346(1.2)	3119(1.5)	28132(1.8)

ments. It can be seen from Table 12 that the PNDG scheme on triangles exhibits higher cost-per-accuracy performance than the NDG scheme on triangles. This stems, however, from the fact that the RKF45 time integrator employ larger values of Δt for the PNDG solution on triangles, thus resulting in faster runtimes and higher performance. We note that the PNDG scheme and NDG scheme on triangles show similar performance when using the SSPRK4 time integrator with the time step size being selected based on the CFL-type condition.

As indicated by the numerical results reported above and in the previous section, we note that the high order schemes offer significant benefits in terms of cost per accuracy. Although the considered choices of DG polynomial bases and element shapes have an implication on the numerical performance, their impact are not as noticeable in comparison to the use of a high-order scheme.

4.3. Nonlinear Stommel problem

We note that realistic scenarios of coastal flow problems usually involve a number of factors *e.g.*, spatially varying bathymetry, curved boundaries, bottom friction, surface wind stress. In this section, we apply the DG schemes to the

Table 12
DG solutions on unstructured meshes. Computing times $T_c^{p,\varepsilon}$ for a specified level of error ε in H_h of the PNDG triangular solution with a given p and time ratios (given schemes relative to the PNDG scheme on triangles for that p). A code [mn] denotes the DG scheme preceding it.

DG bases & mesh	$T_c^{p,\varepsilon}$ of [mn] / $T_c^{p,\varepsilon}$ of [m1]				
	$p = 1$	$p = 2$	$p = 3$	$p = 4$	$p = 5$
	ε = 2.0e-04	ε = 1.0e-06	ε = 1.0e-07	ε = 1.0e-08	ε = 1.0e-09
NDG tri [m1]	6964	36165	20509	13857	13496
NDG quad [m2]	0.75	0.80	0.93	1.10	1.29
NDG mixed [m3]	0.83	1.00	0.93	1.08	1.16
PNDG tri [m4]	0.81	0.76	0.74	0.77	0.78
PNDG quad [m5]	0.54	0.99	0.77	0.77	0.84
PNDG mixed [m6]	0.62	1.13	1.59	1.45	1.90
PNDG ngon[a] [m7]	0.66	1.21	1.05	1.08	2.08

[a] Polygon elements with $r_{\max} = 1$.

nonlinear Stommel problem. Although it is relatively simple, the Stommel problem contains a number of physical processes encountered in the realistic application of SWE and serves as a good prototype for ocean circulation problems.

The so-called nonlinear Stommel problem [7,37] modifies the Stommel problem [38] by including the nonlinear advective term. More precisely, we consider the flow problem governed by the SWE (1) in a rectangular ocean basin of $[0, L]^2$ with source terms that include Coriolis force, surface wind stress, and linear bottom friction, *i.e.*,

$$F_x = f\nu H + \frac{\tau_{sx}}{\rho_0} - \gamma u H, \quad F_y = -fuH + \frac{\tau_{sy}}{\rho_0} - \gamma \nu H \tag{41}$$

where f denotes the Coriolis parameter, (τ_{sx}, τ_{sy}) represents the surface wind stress, ρ_0 is the water density, and the constant γ is the bottom friction coefficient. The Coriolis parameter is taken as $f(y) = f_0 + \beta(y - L/2)$ and the wind stress as

$$\tau_{sx} = -\tau_0 \cos\left(\frac{\pi y}{L}\right), \quad \tau_{sy} = 0. \tag{42}$$

Note that (42) is a simple form of the surface stress associated with the Trades and Westerlies [38]. At the basin boundaries, we consider the no-normal flow boundary condition

$$\mathbf{u} \cdot \mathbf{n} = 0. \tag{43}$$

The no-normal flow condition is imposed weakly using an implementation given in Appendix C.

In the numerical simulations, the values of parameters are as follows: $L = 10^6$, $f_0 = 10^{-4}$, $\beta = 10^{-11}$ 1/m, $g = 10$ m/s^2, $\rho_0 = 1000$ kg/m^3, $\tau_0 = 0.2$ N/m^2, and $\gamma = 2 \times 10^{-6}$ (except for the value of γ, these parameters are identical with those employed in [37]). The steady state is declared when the difference between the solution at time level $t = (n+1)\delta_f$ and at $t = n\delta_f$ are sufficiently small, more specifically,

$$\|H(\boldsymbol{x}, (n+1)\delta_f) - H(\boldsymbol{x}, n\delta_f)\|_\infty < \varepsilon_s, \quad n \in \mathbf{N}. \tag{44}$$

Here, the condition above is checked every $\delta_f = 7200$ s and unless otherwise indicated $\varepsilon_s = 5 \times 10^{-6}$. Unless otherwise indicated, the numerical calculations are initiated with quiescent flow

$$\zeta(\boldsymbol{x}, 0) = H(\boldsymbol{x}, 0) - z_b = 0, \quad (\mathbf{u}H)(\boldsymbol{x}, 0) = \mathbf{0}. \tag{45}$$

Here, the Stommel problems with a flat and non-flat bathymetry are considered. Results for the test problem with flat bathymetry are presented in the following subsection. Subsequently, in Section 4.3.2, we report numerical results for the non-flat bathymetry problem. We also discuss in Section 4.3.2 issues concerning a preserving-still-water property (also known as the well-balanced property) of the DG schemes for a problem with non-flat bathymetry.

4.3.1. Flat bathymetry problem

For the flat bed problem, we consider the ocean basin with a bathymetric depth $z_b = 1000$ m. We examine the numerical performance of the NDG scheme on rectangles and on triangles. For brevity, we present only the results from the NDG scheme on rectangles. We consider five sequentially refined meshes of uniform rectangular elements; the coarsest mesh consists of 5×5 rectangles ($\Delta x = \Delta y = L/5$) and the finest mesh consists of 80×80 rectangles ($\Delta x = \Delta y = L/80$). We conduct the study for the NDG scheme with $p = 1, 2$, and 3. The values of $(\varepsilon_r, \varepsilon_a)$ in the RKF45 time integrator are set to $(7.5 \times 10^{-6}, 1 \times 10^{-9})$. It is noted that the time-independent linear Stommel problem has an exact solution (see [38,7,37]). However, there is no exact solution to the nonlinear Stommel problem. To measure errors in the numerical solutions, we use the approximate solution obtained from a high-resolution calculation, more precisely, from the DG scheme with $p = 7$ on the 10×10 rectangular mesh and $(\varepsilon_r, \varepsilon_a) = (1 \times 10^{-7}, 1 \times 10^{-12})$, as a reference solution.

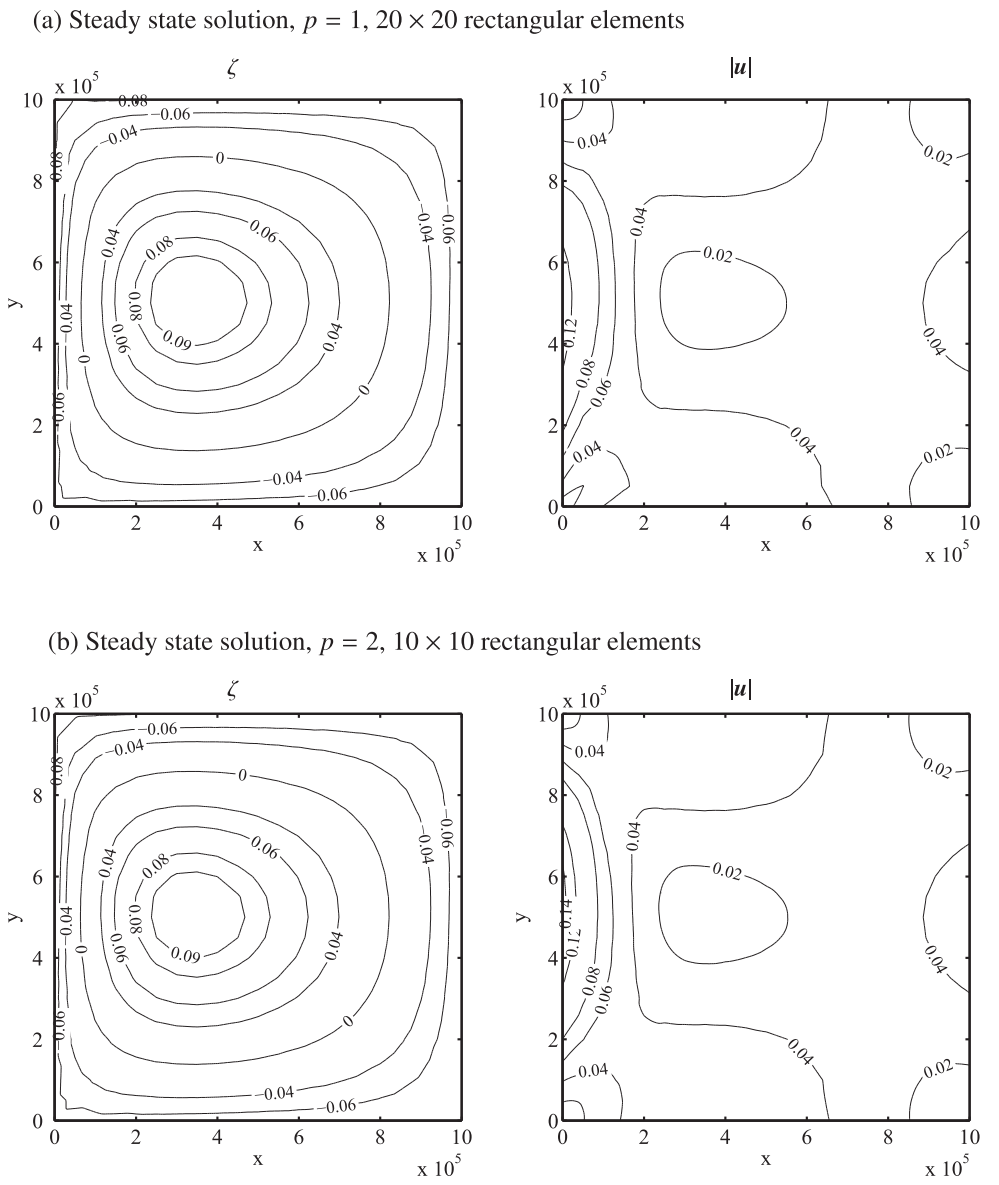


Fig. 11. The free surface elevation $\zeta = H - z_b$ (left) and the velocity magnitude $|\mathbf{u}| = \sqrt{u^2 + v^2}$ (right) at the steady state of the nonlinear Stommel problem obtained from the nodal DG scheme on rectangles. (a) Solution from using $p = 1$ on the mesh of 20×20 rectangles; (b) solution from using $p = 2$ on the mesh of 10×10 rectangles.

Fig. 11 plots the free surface elevation $\zeta = H - z_b$ (left column) and the velocity magnitude $|\mathbf{u}| = \sqrt{u^2 + v^2}$ (right column) at steady state. The result shown in Fig. 11(a) is obtained from the scheme with $p = 1$ (bi-linear element) on the mesh of 20×20 rectangles and in Fig. 11(b) from a scheme with $p = 2$ (bi-quadratic element) on a mesh of 10×10 rectangles. It can be observed that the results from these two simulations qualitatively agree well with the reference solution shown in Fig. 12. Note that, for most calculations, the steady state is reached at approximately $t = 84.8$ days (we intentionally use a larger value of the bottom friction coefficient γ than that used in [7,37] so that the steady state is reached at an earlier time). Fig. 13 plots, on a log–log scale, errors in the approximate solution H_h and $(uH)_h$ through the L_2 norm against the element sizes. In the plots, the element sizes are measured through $\sqrt{N_{el}}$ and the values of errors are normalized by $\|Z_b\|_2$. A slope of each log–log plot, which indicates an overall numerical order of convergence, is reported to the right of the subfigures. For $p > 1$, the numerical solution exhibits an order of convergence typically close to $p + 1/2$ in the L_2-norm.

Fig. 14 shows, on a log–log scale, computing times plotted against $h^{-1} \backsimeq \sqrt{N_{el}}$. On a given mesh, computing times increase with the interpolation order p, as expected. It can be noticed that the log–log plots appear as straight lines; this implies that the computing time behaves approximately like ch^s with respect to element size. The numerical rate s is approximately -3

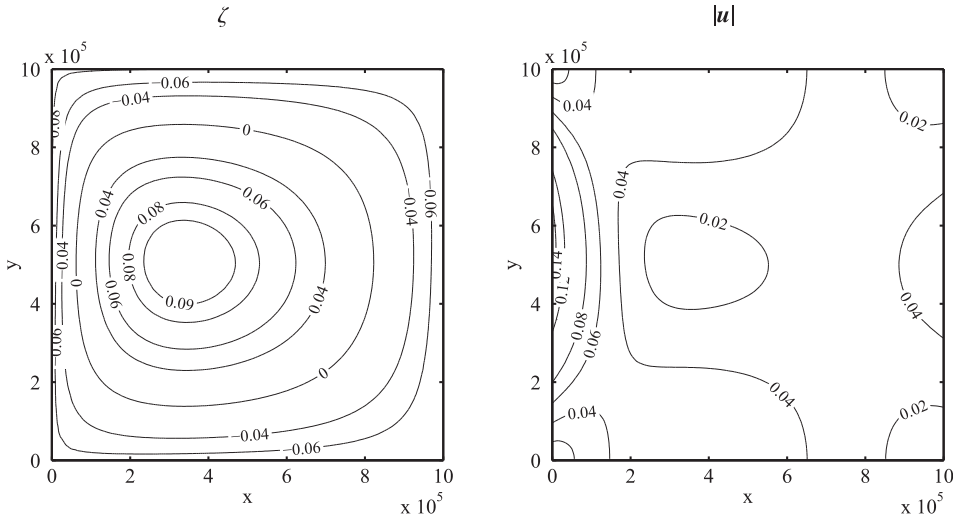


Fig. 12. Reference solution of the nonlinear Stommel problem. The free surface elevation $\zeta = H - z_b$ (left) and the velocity magnitude $|\mathbf{u}| = \sqrt{u^2 + v^2}$ (right).

and appears to be independent of p (the constant c, as expected, depends on p). Fig. 15 depicts, on a log–log scale, the normalized L_2 error in H_h as a function of computing time. Each curve shows a relation between the computing cost and accuracy of the DG solution with a given order p. Since the log–log plots appear approximately as straight lines, the cost functions can therefore be well approximated by $T_c = c_2(\mathcal{E}_H)^{s_2}$ (with $s_2 \approx -3/(p+1)$). More precisely, the cost functions for the total water column height H are as follows

$$T_c = c_2(\mathcal{E}_H)^{s_2}, \quad \text{with } (\log c_2, s_2) = \begin{cases} (-10.18, -1.40) & \text{for } p = 1 \\ (-9.60, -1.05) & \text{for } p = 2 \\ (-7.68, -0.84) & \text{for } p = 3 \end{cases} \tag{46}$$

where $\mathcal{E}_H$ represents the error in H_h in the L_2 norm normalized by $\|z_b\|_2$. In Table 13, we tabulate from (46) the computing times for different error levels. The value inside a parenthesis is a ratio between the cost of the scheme with $p-1$ to that of p. It can be seen that the high order scheme shows a clear advantage over the scheme with $p = 1$ from the cost per accuracy aspect. For instance, at the error level of 1×10^{-7} or 5×10^{-7}, the computational cost in the DG solution with $p = 3$ is about three orders of magnitude lower than the DG solution with $p = 1$. All these convergence and cost per accuracy analyzes show similar behavior to the manufactured solution problem presented in Section 4.2.

We also report in Table 13 the data that comes from the cost functions of the NDG solution on triangles (the triangular meshes considered are built in a way similar to those described in Section 4.2.1, *i.e.*, by bisecting rectangular elements of the rectangular elements). It can be observed that the scheme on rectangles has lower costs per accuracy (ranging approximately from 1.3 to 2.5 times lower) than the NDG solution on triangles. We note the numerical order of convergence in H of the NDG scheme on triangles is slightly higher than that of the scheme on rectangles (the opposite of the convergence rate in uH and vH); on the same so-called mesh resolution, the scheme on rectangles is faster than the scheme on triangles for all p considered.

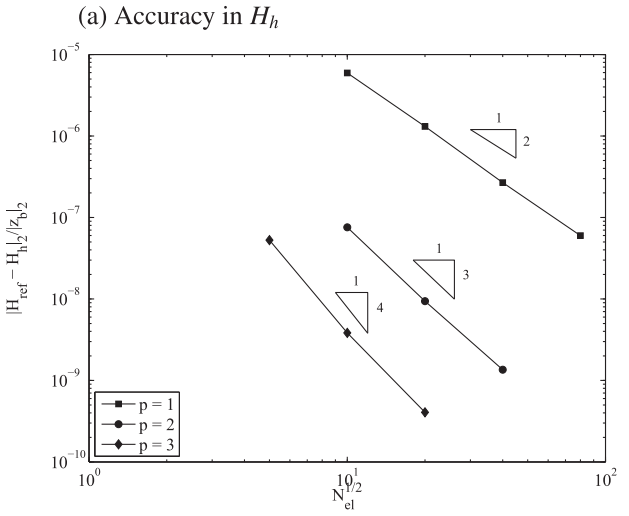
4.3.2. Non-flat bed problem

4.3.2.1. Preserving still water flow. One concern of DG or other methods that are based on the conservative form of shallow water equations involves their ability to preserve the state at rest solution

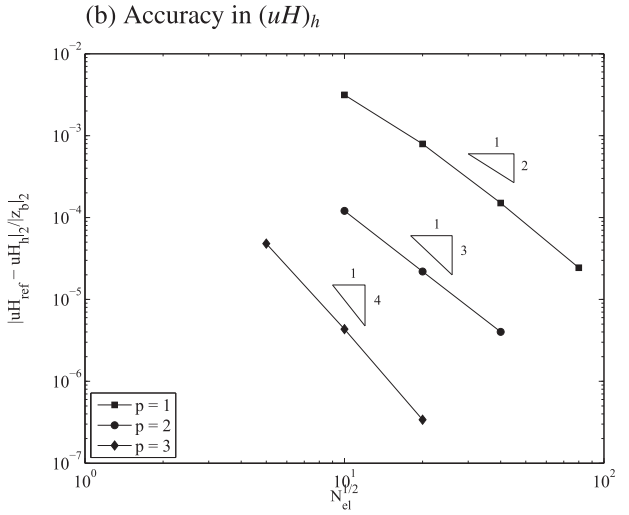
$$\mathbf{u}(\boldsymbol{x}) = \mathbf{0} \quad \text{and} \quad \zeta = H(\boldsymbol{x}) - z_b(\boldsymbol{x}) = C, \tag{47}$$

where ζ denotes the surface elevation and C is a constant value, in the time marching process. Note that a typical problem that admits (47) as a solution is flow in an enclosed basin in the absence of wind forcing term. Schemes that preserve such the state, *i.e.*, ζ_h remains constant and $\mathbf{u}_h$ remains zero at all time, are called a well-balanced scheme [16,17]. To obtain a well-balanced property, numerical schemes are devised so that the right-hand-side terms vanish for a given approximate solution of the state at rest solution.

Here, we provide a treatment for obtaining the well-balanced property in the scheme using the nodal basis and conforming mesh and order (although not discussed here, we note that this treatment is also directly applicable to some modal-based DG schemes). The approximate solution discussed below, unless specified, is associated with the steady state at rest solution (47). We also assume here that the bathymetry z_b is continuous. In this treatment, the bathymetry is replaced by an



p	Convergence rate in H_h
1	2.21
2	2.90
3	3.51



p	Convergence rate in $(uH)_h$
1	2.34
2	2.45
3	3.57

Fig. 13. Error and convergence rate in the approximate solution at steady state for the nonlinear Stommel problem. (a) The normalized L_2-error in the total water column $\|H_{\text{ref}} - H_h\|_2/\|z_b\|_2$ as a function of $h^{-1} \backsimeq \sqrt{N_{el}}$; (b) the normalized L_2 error in the x-directed water discharge $\|(uH)_{\text{ref}} - (uH)_h\|_2/\|z_b\|_2$ as a function of $h^{-1} \backsimeq \sqrt{N_{el}}$.

interpolant of degree identical to the nodal basis considered. More precisely, when employing the nodal basis of degree p, the bathymetry in the element K is approximated by the interpolant of degree p, namely,

$$z_b(\boldsymbol{x}) \approx z_{h,b} \equiv (Iz_b) = \sum_{m=1}^{M_p} z_b(\boldsymbol{x}_m)\phi_m(\boldsymbol{x}), \tag{48}$$

where, as a reminder, ϕ_m denotes nodal basis functions, $\boldsymbol{x}_m$ the location of nodal points, and M_p denotes the number of nodal points. Adopting (48) leads to $H_h - z_{h,b} = C$ for all $\boldsymbol{x} \in K$ (as a reminder, H_h is the interpolant H and in this case $H = C - z$). For continuous z_b, the approximate bathymetry $z_{h,b}$ is piecewise continuous; this results in a single-valued H_h on the element interfaces. Substituting the approximate solution $\mathbf{q}_h^* = (H_h, (\mathbf{u}H)_h = \mathbf{0})$ and $z_{b,h}$ into the DG-SWE weak formula yields the following right hand side term

$$\mathbf{r} = \begin{pmatrix} 0 \\ \int_K \frac{1}{2} g H_h^2 \frac{\partial v_h}{\partial x} d\boldsymbol{x} - \int_{\partial K} \widehat{\mathbf{F}}_2^* \cdot \mathbf{n} v_h ds + \int_K g H_h v_h \frac{\partial z_{h,b}}{\partial x} d\boldsymbol{x} \\ \int_K \frac{1}{2} g H_h^2 \frac{\partial v_h}{\partial y} d\boldsymbol{x} - \int_{\partial K} \widehat{\mathbf{F}}_3^* \cdot \mathbf{n} v_h ds + \int_K g H_h v_h \frac{\partial z_{h,b}}{\partial y} d\boldsymbol{x} \end{pmatrix} \tag{49}$$

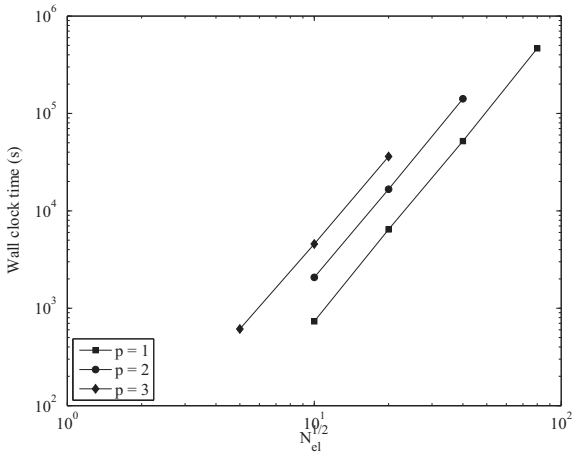


Fig. 14. Nonlinear Stommel problem: wall clock time as a function of $h^{-1} \backsimeq \sqrt{N_{el}}$.

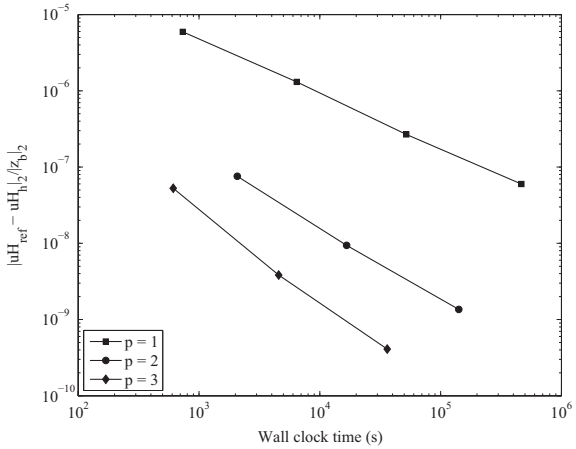


Fig. 15. Nonlinear Stommel problem: error $\|H_{\text{ref}} - H_h\|_2/\|z_b\|_2$ as a function of the wall clock time.

Table 13
Nonlinear Stommel problem. Computing time T_c^ε required to achieve a specified level of error in H_h of the nodal DG solution on rectangles and on triangles. A numeric value in the parenthesis is the ratio between T_c^ε of a scheme with $p-1$ and that of p.

Scheme	p	Computing time $T_c^\varepsilon \equiv T_c(\varepsilon)$			
		ε = 1e−06)	ε = 1e−07	ε = 5e−08	ε = 2.5e−09
NDG quad	1	8903.63	221114.35	581525.45	37981329.79
	2	133.72(66.58)	1497.62(147.64)	3099.16(187.64)	71826.86(528.79)
	3	49.08(2.72)	337.61(4.44)	603.31(5.14)	7416.56(9.68)
NDG tri	1	17228.01	462327.69	1244617.57	89915965.39
	2	402.83(42.77)	3673.19(125.87)	7145.11(174.19)	126735.88(709.48)
	3	88.55(4.55)	530.26(6.93)	908.80(7.86)	9326.77(13.59)

for all $v_h \in \mathcal{P}^p(K)$, where $\widehat{\mathbf{F}}_2^* \cdot \mathbf{n} = (1/2)gH_h^2 n_x$ and $\widehat{\mathbf{F}}_3^* \cdot \mathbf{n} = (1/2)gH_h^2 n_y$ are the normal numerical fluxes for the x- and y-momentum equations evaluated at $\mathbf{q}_h^*$, respectively. It can be verified by integrating by-parts the first term of (49) and using $H_h = C - z_{h,b}$ that the right hand side term $\mathbf{r}$ vanishes, thus yielding a well-balanced property. Note that it is assumed here that the bathymetry has no discontinuity. We refer to approaches in [18,20,11] for handling the discontinuous bed. Note that the approaches devised in [18,20] use the L_2-projection on $\mathcal{P}^p(K)$ to approximate the bathymetry in each element. Generally, this results in a discontinuous approximate bed. The well-balanced property in these approaches are accomplished through the modified numerical fluxes that are based on the hydrostatic reconstruction technique [17].

Note that, the integral formula (49) must be computed exactly in a numerical realization to obtain the well balance property. For triangular elements (tensor product rectangle elements), this can be done by using quadrature rules that integrate exactly the polynomials of degree $3p-1$ ($3p$) for the volume integral terms and $3p$ ($3p$) for the edge integrals (note that the values in the parentheses are for the tensor-product rectangular elements). It can be checked that the nodal-integration procedure described in Section 3.1 does not meet this requirement, thus rendering the NDG scheme non well-balanced (see numerical results below). We find that one can reduce the order of quadrature required in the numerical implementation, which implies less computational work, by considering the widely-used equivalent alternative form of the SWEs [3,5], more precisely, considering (1) with

$$\mathbf{q}=\begin{pmatrix}\zeta\\ uH\\ vH\end{pmatrix},\quad \boldsymbol{f}=\begin{pmatrix}uH\\ u^2H+\frac{1}{2}g(H^2-z_b^2)\\ uvH\end{pmatrix},\quad \boldsymbol{g}=\begin{pmatrix}vH\\ uvH\\ v^2H+\frac{1}{2}g(H^2-z_b^2)\end{pmatrix},\quad \text{and}\quad \mathbf{s}=\begin{pmatrix}0\\ g\zeta\frac{\partial z_b}{\partial x}+F_x\\ g\zeta\frac{\partial z_b}{\partial y}+F_y\end{pmatrix}, \tag{50}$$

where $\zeta=H-z_b$ denotes the surface elevation. It can be shown that, with this alternative form of the SWE, the associated DG weak formula with the approximate bathymetry (48), $\mathbf{u}_h=\mathbf{0}$, $\zeta_h=C$, and $(H_h^2-z_{h,b}^2)=C(H_h+z_{h,b})$ can be computed exactly by using the quadrature rules that integrate exactly polynomials of degree up to $2p-1$ ($2p$) for the volume integrals and up to $2p$ ($2p$) for the edge integrals when the triangular elements are considered (the values in the parentheses are for the tensor-product rectangular elements). In other words, the quadratures of such accuracies are sufficient in order to achieve the well-balanced property. Note that the quadratures required are less accurate than those required in the formula associated with the SWE form of (2). In addition, it can be verified that the NDG scheme with the nodal-integration approach is well-balanced when (50) is considered.

To examine the well-balanced property, we consider a rectangular enclosed basin of $[0,L]^2, L=10^6$, with the bathymetry

$$z_b=\frac{3z_{b,0}}{4}-\frac{z_{b,0}}{4}\tanh\left[\frac{1}{2\times10^5}\left(-\frac{\sqrt{2}}{2}x+\frac{\sqrt{2}}{2}y-\frac{L}{5}\right)\right], \tag{51}$$

where $z_{b,0}=1000$ and the state at rest as the initial condition

$$\zeta(\boldsymbol{x},t=0)=H(\boldsymbol{x},t=0)-z_b(\boldsymbol{x})=\zeta_0,\quad \mathbf{u}H=\mathbf{0}. \tag{52}$$

where $\zeta_0=1/4$. In this study, we include the Coriolis force and a linear bottom friction term; surface wind stress is excluded. The values of the physical parameters are identical to those listed in Section 4.3.

Below, we report the results obtained by utilizing three different ways in realizing the integrals in the DG weak formula based on the nodal basis expansion on triangular elements. The first approach (M1) considered is the NDG scheme. This scheme, as a reminder, uses the nodal-integration approach for realizing the integration terms (see Section 3.1). The second and third approaches use the quadrature formula in evaluating the integrals. In the second approach (M2), the area integrals are computed by means of a quadrature rule that integrates exactly polynomials of degree up to $2p$ and the edge integrals are evaluated by a quadrature rule that integrates exactly polynomials of degree up to $2p+1$. The third approach (M3) employs quadrature rules that integrate polynomials of degree up to $3p-1$ for an area integration and up to $3p$ for an edge integration. Note that, in the M2 and M3 approaches, we use the cubature rules provided in [39] for the area integration over a triangle and use the classical one-dimensional Gauss quadrature rule for the edge integration. For brevity, the SWEs with (2) is termed the H-form SWE and the SWE with (50) the ζ-form SWE. The numerical solutions are computed on a triangle mesh consisting of 800 elements constructed by bisecting a uniform grid of 20×20 points. We use the RKF45 time integrator with the tolerance $\varepsilon_r=5\times10^{-7}, \varepsilon_a=5\times10^{-12}$. The calculations are performed until $t=10$ days (86,400 s) is reached. Table 14 tabulates the absolute maximum errors at the nodes in the total water column H_h and the x-directed discharge $(uH)_h$ at $t=10$ days. It can be clearly observed from this table that, in the H-form SWEs, the M3 approach exhibits the well-balanced

Table 14
Well-balanced test. Absolute maximum errors at the nodes in H_h and $(uH)_h$ at $t=10$ days.

	p	$\|H-H_h\|_{l,\infty}$			$\|uH-uH_h\|_{l,\infty}$		
		M1	M2	M3	M1	M2	M3
ζ-form	1	1.137e−13	2.274e−13	2.274e−13	3.759e−11	1.263e−10	1.263e−10
	2	2.274e−13	3.411e−13	3.411e−13	4.104e−10	5.713e−10	5.649e−10
	3	4.548e−13	5.684e−13	7.958e−13	3.175e−09	5.500e−09	1.050e−08
	4	7.958e−13	2.274e−12	1.251e−12	8.579e−09	1.914e−08	2.034e−08
H-form	1	3.410e+00	1.211e−11	1.211e−11	2.809e+07	1.568e−08	1.568e−08
	2	9.386e−02	9.997e−04	7.969e−11	2.497e+01	2.884e−01	2.269e−07
	3	3.359e−03	1.984e−05	4.940e−10	9.260e−01	1.789e−02	3.385e−06
	4	1.678e−04	2.104e−06	6.096e−10	6.978e−02	1.784e−03	6.308e−06

M1 – NDG scheme; M2 - nodal basis, quadrature rules $(2p, 2p+1)$.
M3 – nodal basis, quadrature rules $(3p-1, 3p)$.

Table 15
Nonlinear Stommel problem with linear bed and (54) surface wind stress. Absolute maximum at the nodes of surface elevation and x-directed discharge at steady state.

p	$\|\zeta_h\|_{l,\infty}$			$\|uH_h\|_{l,\infty}$		
	M1	M2	M3	M1	M2	M3
1	7.517e−08	7.394e−08	7.394e−08	5.121e−06	4.643e−06	4.640e−06
2	7.885e−08	7.474e−08	7.475e−08	6.841e−06	6.858e−06	6.858e−06
3	8.001e−08	7.579e−08	7.579e−08	7.363e−06	7.357e−06	7.357e−06
7	7.964e−07	7.560e−08	7.560e−08	7.809e−06	7.792e−06	7.793e−06

M1 – NDG scheme; M2 - nodal basis, quadrature rules $(2p, 2p+1)$.
M3 – nodal basis, quadrature rules $(3p-1, 3p)$.

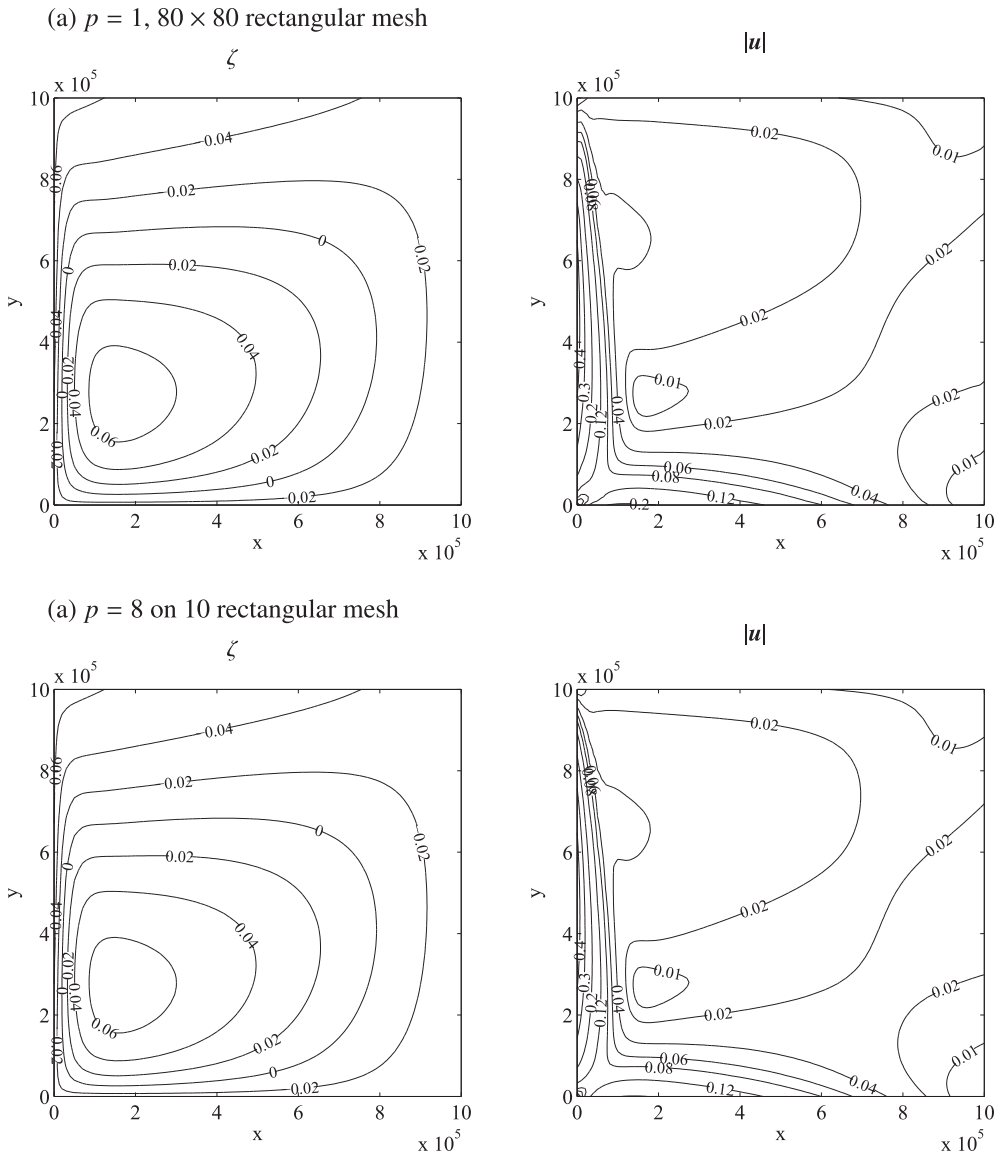


Fig. 16. The free surface elevation $\zeta = H - z_b$ (left) and the velocity magnitude $|\mathbf{u}| = \sqrt{u^2 + v^2}$ (right) at the steady state of the nonlinear Stommel problem with linear bathymetry obtained from the NDG scheme. (a) Solution from using $p = 1$ on the mesh of 80×80 rectangular elements; (b) solution from using $p = 8$ on the mesh of 10×10 rectangular elements.

property; the NDG scheme (M1) is not well-balanced and in fact it yields poor results especially for low order p. For the ζ-form SWEs, all three approaches exhibit the well-balanced property of the state at rest solution. This indicates that the well-

balanced property can be obtained with less computationally expensive realizations in the ζ-form SWEs. These observations on the numerical results verify the discussion above on the well-balanced issues.

4.3.2.2. Stommel problem with linear bed. We consider a problem described in Section 4.3 with a linear bathymetric profile

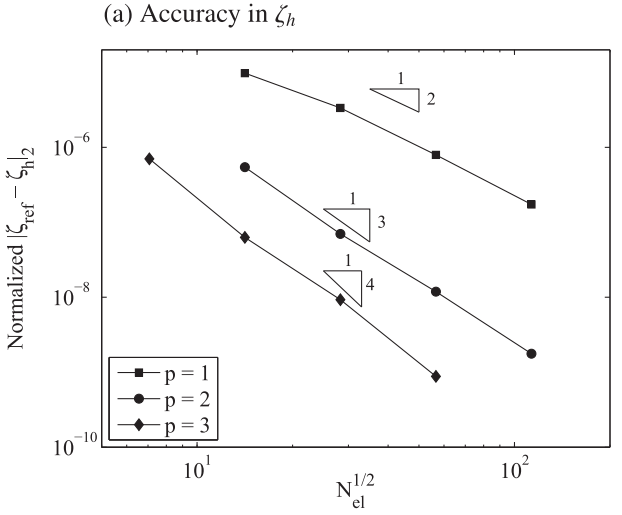
$$z_b = \frac{z_{b,0}}{2}\left[1 - \frac{1}{2L}(-x + (y - L))\right] \tag{53}$$

where $z_{b,0} = 1000$. Note that the deepest and shallowest points of the basin are at the southeast and northwest corners, respectively. In the calculations, we use identical meshes and parameters employed in the flat bathymetry test case (see Section 4.3.1). The ζ-form of SWE is considered in the study below and, unless otherwise indicated, we note that results reported below are of this SWE form.

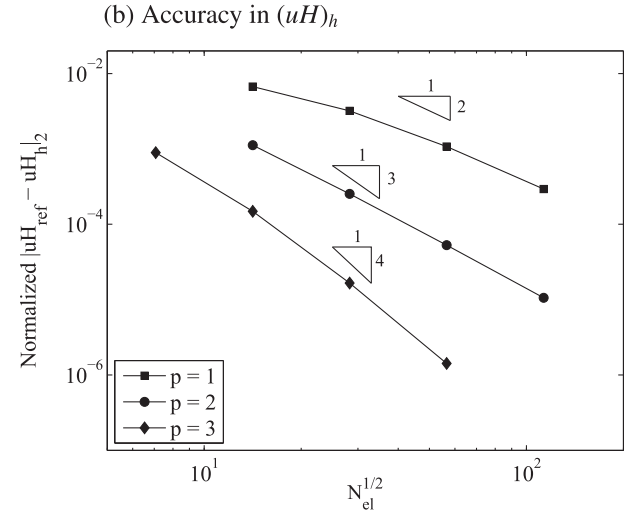
We first examine whether the approximate solutions evolve to the steady state at rest when the effect of surface wind stress is removed after a certain time. Here, we consider the case where the surface wind stress is given by

$$\tau_{sx} = -\frac{\tau_0}{2}\left[1 - \tanh\left(\frac{t - T_s}{T_r}\right)\right]\cos\left(\pi\frac{y}{L}\right), \quad \tau_{sy} = 0, \quad T_s = 8 \text{ days}, \quad T_r = 0.5 \text{ days}. \tag{54}$$

The effect of the wind forcing term begins subsiding around $t = 7.5$ days and is completely absent for large t. The integration is started with the initial condition (52) and is carried out until reaching steady state or $t = 150$ days, whichever comes first.



p	Convergence rate in H_h
1	1.95
2	2.74
3	3.17



p	Convergence rate in $(uH)_h$
1	1.51
2	2.24
3	3.10

Fig. 17. Nonlinear Stommel problem with linear bathymetry: log–log plots of errors at $t = 12$ days in the L_2-norm normalized by $\|z_{b0}\|_2$ versus $h^{-1} \backsimeq \sqrt{N_{el}}$. (a) Errors in the surface elevation. (b) Errors in the x-directed water discharge.

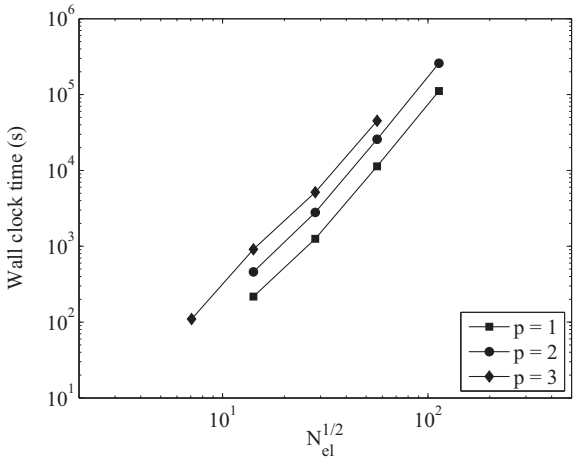


Fig. 18. Nonlinear Stommel problem with linear bathymetry: wall clock time as a function of $h^{-1} \backsimeq \sqrt{N_{el}}$.

The steady state is declared based on the criteria (44) with $\varepsilon_s = 10^{-8}$. We use the triangular mesh of 200 elements, which is built from the 10×10 uniform grid, in the calculation. Table 15 tabulates absolute maximum values at the nodes in the steady state solution from using different realizations of the nodal-basis based DG scheme (see the previous section for the description of the implementation). We note that all three approaches yield steady state solutions approximately at $t = 128$ days. The result demonstrates that the less expensive realizations (M1 and M2 approaches) do not deteriorate the quality of the solution that evolves to the steady state at rest solution.

Subsequently, we consider the problem with the persistent surface wind stress (42) and consider the ζ-form SWE. Here, we focus mainly on examining numerical performance of the NDG scheme (M1 scheme). Fig. 16 illustrates the approximate solution ζ_h and $(uH)_h$ at steady state obtained using the rectangular elements. Fig. 16(a) shows the solution using $p = 1$ on the mesh of 80×80 elements and Fig. 16(b) the solution using $p = 8$ on the mesh of 10×10 elements. Note that the steady state is declared when the criteria (44) with $\varepsilon_s = 10^{-8}$ is satisfied. It can be seen that, a center of circulation is near the southwest corner of the basin and water piles up in the vicinity of such a circulation center. The steady state is reached at approximately $t = 103$ days for most calculations. Note that, in the calculations on coarse triangular meshes and high p ($p \geqslant 7$), the NDG scheme fails to yield a steady state solution. We believe this is due to aliasing errors. Applying a mild spectral filter [25] appears to resolve this instability issue. Note that we do not see such an instability issue in the M2 and M3 approaches which utilize the quadrature rules in realizing the DG weak formula.

In assessing the numerical performance, the calculations are carried out until $t = 12$ days and unless otherwise indicated numerical results discussed below are the results at this specific time. The tolerance $(\varepsilon_r, \varepsilon_a)$ in the RKF45 time integrator are set to $(7.5 \times 10^6, 1 \times 10^{-9})$ in the calculations. As done in the flat-bathymetry case, we use the approximate solution from a high-resolution calculation as a reference solution. More specifically, the approximate solution from the NDG scheme with $p = 7$ on the 10×10 rectangular mesh and $(\varepsilon_r, \varepsilon_a) = (1 \times 10^{-7}, 1 \times 10^{-12})$ is used as the reference solution for assessing

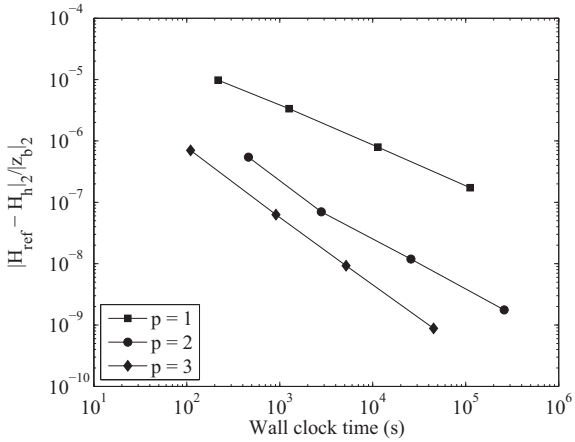


Fig. 19. Nonlinear Stommel problem with linear bathymetry: error $\|\zeta_{ref} - \zeta_h\|_2 / \|z_b\|_2$ as a function of the wall clock time.

Table 16
Nonlinear Stommel problem with linear bed: computing time T_c^ε required to achieve a specified level of error in H_h of the nodal DG solution on rectangles and on triangles. A numeric value in the parenthesis is the ratio between T_c^ε of a scheme with $p-1$ and that of p.

Scheme	p	Computing time $T_c^\varepsilon \equiv T_c(\varepsilon)$			
		ε = 1e−06	ε = 1e−07	ε = 5e−08	ε = 2.5e−09
NDG quad	1	2510.45	99096.31	299635.78	35763219.21
	2	112.89(22.24)	1630.82(60.76)	3643.57(82.24)	117599.81(304.11)
	3	50.40(2.24)	430.14(3.79)	820.21(4.44)	13348.67(8.81)
NDG tri	1	7692.93	268412.10	781995.33	79490736.19
	2	190.71(40.34)	2497.78(107.46)	5418.23(144.33)	153935.97(516.39)
	3	77.64(2.46)	618.55(4.04)	1155.28(4.69)	17189.70(8.96)

numerical performance. For brevity, we present, unless specified, the results from the NDG scheme on triangles. Fig. 17 plots, on a log–log scale, errors in ζ_h and $(uH)_h$ through the L_2-norm against the element sizes measured by $\sqrt{N_{el}}$. Note that the values of errors in these plots are normalized by $\|z_b\|_2$. Overall numerical orders of convergence are reported in the tables to the right of each subfigures. We note that the numerical solution converges at the rate close to $p+1/2$. Fig. 18 plots, on a log–log scale, wall-clock times as a function of $h^{-1}\backsimeq\sqrt{N_{el}}$. The value of overall slope of each curve, which appears to be independent of p, is approximately 3. Fig. 19 shows the log–log plots of the normalized L_2 errors in ζ_h against the wall clock times. The plots appear approximately as straight lines in the log–log scale; this indicates that the relation between the computing cost and the accuracy can be approximately described by $T = c_2(\mathcal{E}_h)^{s_2}$, with $s_2 \approx -3/(p+1/2)$. More precisely, the cost functions for a given level of accuracy $\mathcal{E}_h$ in the surface elevation ζ_h are approximately as follows

$$T_c = c_2(\mathcal{E}_\zeta)^{s_2}, \quad \text{with } (\log c_2, s_2) = \begin{cases} (-12.36, -1.54) & \text{for } p=1 \\ (-10.18, -1.12) & \text{for } p=2 \\ (-8.09, -0.90) & \text{for } p=3 \end{cases} \tag{55}$$

Table 16 tabulates from the cost functions the computing times for achieving different levels of accuracy. In addition, we include in this table data from the cost functions of the NDG scheme on rectangles. The value inside a parenthesis is a cost ratio of the scheme with $p-1$ to the scheme with p for the same given level of error. It can be observed that the high order schemes outperform the scheme with $p=1$ in terms of cost to achieve a specific level of accuracy. To achieve the same level of accuracy, the costs of the DG solution with $p=3$ are almost two to three order of magnitude lower than the linear DG solution. Data in this table also indicate that the NDG solution on rectangles has higher performance (ranging approximately between 1.3 and 3 times) than the NDG solution on triangles. The gain from employing rectangular elements is minor in comparison to the gain from using the high order schemes. We note that the characteristics of the DG solution in terms of convergence and cost per accuracy are similar to those observed in the manufactured solution test case (Section 4.2) and the flat-bed Stommel test problem (Section 4.3.1).

5. Conclusions

In this work, we present a comprehensive performance assessment of LLF-flux nodal discontinuous Galerkin (NDG) and polymorphic nodal discontinuous Galerkin (PNDG) solution of the time-dependent nonlinear SWE. The integration in time is carried out using the RKF45 time integrator, which has a mechanism to adjust Δt to control temporal errors. These methods are applied to a set of problems with sufficiently smooth solutions: a manufactured-solution problem and the nonlinear Stommel problem with flat- and non-flat bathymetry.

The numerical solutions show that all the schemes tested exhibit a convergence rate of order between $O(h^p)$ and $O(h^{p+1})$, typically close to $p+1/2$, for the water column height. The performance analyzes clearly show that the high-order schemes ($p>1$) outperform the linear-element scheme in terms of cost per accuracy performance. For a specified level of error, the computational cost required decreases noticeably as the degree p of the DG polynomial increases. In the test problems employed here, for a moderate specified level of error, the computational cost for the schemes with $p=3$ are typically about two to four orders of magnitude, in other words a hundred to ten thousand times, lower than the scheme with $p=1$. The benefit gained by employing a one-higher order interpolant however diminishes as the interpolation order p increases. We find that the use of cubic or bi-cubic interpolants ($p=3$), is particularly appealing due to dramatic improvement in cost as compared to (bi-) linear interpolants and moderate gain over (bi-) quadratic interpolants.

In addition, we examine whether element shapes other than triangles, in particular quadrilaterals, which reduce the number of elements in the computational mesh would improve the efficiency of DG solutions. Here, we consider a mesh setting in which computational meshes of various element shapes are derived from a given triangular mesh. The numerical results provide evidence that there may be a benefit in using quadrilateral elements, especially, those with nodal tensor-product bases. In the numerical experiments conducted, the NDG scheme on rectangles exhibits higher (or at worst comparable to) cost-per-accuracy performance as compared to other schemes. We believe that this promising performance stems primarily from two reasons. First, quadrilateral meshes contains fewer elements. Second, the tensor-product elements improve/retain the

accuracy level owing to the tensor-product bases spanning additional cross polynomial terms for a given degree p. We note that the performance benefit of the tensor-product schemes is however relatively minor in comparison to using high order elements.

A treatment of the bed term that leads to a well-balanced scheme has been also discussed. Such a treatment is based on replacing the bathymetry with an interpolant of the same degree with the DG interpolant and exact realization of the DG weak formula at the still water state. The latter requirement renders the schemes, which uses the so-called nodal integration approach in evaluating the integral terms, non well-balanced when the standard SWE form (2) is considered. We find that when employing instead the equivalent, frequently-used form of SWE (50), the well-balanced property can be achieved with less expensive realization technique, including the NDG scheme.

In this work, we use a manufactured-solution problem with a tide-like solution and wind-driven circulation problems. Numerical evidence shown here suggests that there is a significant cost performance benefit achieved by using the high-order DG method for these types of problems. A similar conclusion can be expected in general for smooth-solution problems, since in these cases, high-order accuracy solutions can be expected when using the high-order DG methods. We note that the cost performance benefit is also reported in high-order solutions to the Navier–Stokes equations with smooth solutions [40]. Although a problem with a curvilinear domain is not examined here, it is noted that, as demonstrated in our work [41], a proper treatment of no-normal flow on solid curved walls is crucial for an accurate DG solution to the SWE, including a linear-element DG. Performance studies for problems that contain more challenging features such as wetting/drying fronts, derivative discontinuities, and (inviscid) shocks are a subject of our future studies. Without going into detail, we note that, with the features mentioned, high-order methods will not yield high-order accuracy solutions in a global sense using fixed grid solutions. In areas away from these features where the solution is smooth, good convergence can still be expected provided that mechanisms that are in place to handle possible numerical artifacts that may be induced by these features preserve accuracy in the smooth-solution areas.

Acknowledgements

This work was supported by National Science Foundation Grants OCI-0749015, OCI-0746232, DMS-0915118, DMS-1217071, and DMS-1217218. Authors D. Wirasaet and J.J. Westerink were also supported by the Henry J. Massman and the Joseph and Nona Ahearn endowments at the University of Notre Dame.

Appendix A. Evaluation of the stiffness matrix

In this section, we describe an approach used in evaluating the stiffness matrices defined in Section 3.1. Consider the stiffness matrix associated with the x-directed flux, *i.e.*,

$$\mathscr{S}_x = (\mathcal{S}_{x,(i,j)}),\ \mathcal{S}_{x,(i,j)} = \int_K \frac{\partial \phi_i}{\partial x}\phi_j d\boldsymbol{x}, \quad i = 1,\ldots,M_p, \quad j = 1,\ldots,M_p. \tag{A.1}$$

To compute such a matrix, the partial derivative of ϕ_i with respect to x is written in the nodal representation, more precisely

$$\frac{\partial \phi_i}{\partial x} = \sum_{n=1}^{M_p} D_{x,(n,i)}\phi_n(\boldsymbol{x})$$

where

$$D_{x,(i,j)} \equiv \left.\frac{\partial \phi_j}{\partial x}\right|_{\boldsymbol{x}_i}$$

denotes an entry of the so-called derivative matrix $\mathbf{D}_x$. Substituting and manipulating yield the following result

$$\mathscr{S}_x = \mathbf{D}_x^T \mathscr{M} \tag{A.2}$$

where $\mathscr{M}$ is a mass matrix (with respect to the nodal basis functions) defined and evaluated as follows

$$\mathscr{M} \equiv \int_K \boldsymbol{\phi}\boldsymbol{\phi}^T d\boldsymbol{x} = J\int_{\mathcal{K}} (\mathbf{V}^{-1})^T \widetilde{\boldsymbol{\phi}}\widetilde{\boldsymbol{\phi}}^T \mathbf{V}^{-1} d\boldsymbol{\xi} = (\mathbf{V}^{-1})^T \mathbf{V}^{-1} \tag{A.3}$$

where $J = (\Delta X)^2$ is the Jacobian of the geometric transformation (14). The remaining task for determining the stiffness matrix is to find the derivative matrix $\mathbf{D}_x$. Since $\mathbf{V}^T\boldsymbol{\phi} = \widetilde{\boldsymbol{\phi}}$, it follows that $\mathbf{V}^T\partial\boldsymbol{\phi}/\partial x = \partial\widetilde{\boldsymbol{\phi}}/\partial x$. The derivative matrix can thus be determined from

$$\mathbf{D}_x\mathbf{V} = \mathscr{D}_x \tag{A.4}$$

where $\mathscr{D}_x$ is a matrix with the entries

$$\mathcal{D}_{x,(i,j)} \equiv \left.\frac{\partial \widetilde{\phi}_j}{\partial x}\right|_{x_i}, \quad i = 1,\ldots,M_p, \quad j = 1,\ldots,N_p$$

which can be computed easily in practice. It is noted that the stiffness matrix associated with y-directed flux, $\mathcal{S}_y = \int_K \partial\boldsymbol{\phi}/\partial y \boldsymbol{\phi}^T d\boldsymbol{x}$, can be computed in an analogous way.

Appendix B. Remarks on code implementation details

The main computing cost in the simulations involves evaluating of the right-hand-side term of the system of ODEs (33), *i.e.*, $\mathbf{M}^{-1}\mathbf{r}(\widetilde{\mathbf{u}}, t)$, which is required by the time integration solver for a given solution vector $\widetilde{\mathbf{u}}$ and time t. Algorithm 1 depicts, in brief, an outline of the steps employed in the calculation of the right-hand-side vector. Here, a one-dimensional array is used to store the global solution; the entries of the expansion coordinates belonging to the same element are kept in consecutive order. The right-hand-side term associated with the ith-element is determined within the ith-iteration of a loop over the elements. It is obtained by combining the contribution from the volume integrals and edge integrals of all edge segments. As a reminder, in Algorithm 1, $\boldsymbol{S}_x^i$ (and $\boldsymbol{S}_y^i$) denotes a (pre-computed) generalized stiffness matrix; the vectors $\boldsymbol{f}_x^i$ and $\boldsymbol{f}_y^i$ denote, respectively, a vector of nodal coordinates of x- and y-directed flux (see Section 3.1). An edge segment is a straight line and, for non-conforming elements, it is not the entire edge of an element. For conforming edges and order, an approach similar to that utilized in treating a volume integral of the nonlinear flux term is adopted for the calculation of the edge integral, *i.e.*, by writing the numerical flux $\widehat{\boldsymbol{f}}_h \cdot \mathbf{n}$ as a linear combination of one-dimensional Lagrange basis functions of order p, $\overline{\boldsymbol{\phi}} = \{\overline{\phi}_m(\varsigma(\boldsymbol{x})), \varsigma \in [-1,1], \boldsymbol{x} \in (\partial K)_j\}_{m=1,\ldots,p}$, and determining the edge integral through

Algorithm 1. Right-hand-side (RHS) vector $\mathbf{M}^{-1}\mathbf{r}(\widetilde{\mathbf{u}}, t)$ calculation

Given a vector of expansion coordinates $\widetilde{\mathbf{u}} = \{(\widetilde{\boldsymbol{u}}^1)^T \ (\widetilde{\boldsymbol{u}}^2)^T \ \cdots \ (\widetilde{\boldsymbol{u}}^{N_e-1})^T \ (\widetilde{\boldsymbol{u}}^{N_e})^T\}^T$
▷ N_e-the number of elements
for $i = 1$ to N_e **do**
  ▷ contribution from volume integration
  $\boldsymbol{r}^i \leftarrow \boldsymbol{S}_x^i \boldsymbol{f}_x^i(\widetilde{\boldsymbol{u}}^i) + \boldsymbol{S}_y^i \boldsymbol{f}_y^i(\widetilde{\boldsymbol{u}}^i)$
  ▷ contribution from edge integration
  for $j = 1$ to N_f^i **do**
    ▷ N_f^i-the number of edge segments of the ith-element
    ▷ $\bigcup_{j=1}^{N_j^i}(\partial K_i)_j = \partial K_i, ((\partial K_i)_j$-$j$th-edge segment
    $\boldsymbol{r}^i \leftarrow \boldsymbol{r}^i + \int_{(\partial K_i)_j} \widetilde{\boldsymbol{\phi}}^i \widehat{\boldsymbol{f}} \cdot \mathbf{n} ds$
  end for
  $\boldsymbol{r}^i \leftarrow (\mathscr{M}^i)^{-1} \boldsymbol{r}^i$
end for
Return RHS vector $\mathbf{r} = \left\{(\boldsymbol{r}^1)^T \ (\boldsymbol{r}^2)^T \ \cdots \ (\boldsymbol{r}^{N_e-1})^T \ (\boldsymbol{r}^{N_e})^T\right\}^T$

$$\int_{(\partial K)_j} \widetilde{\phi}_i \widehat{\boldsymbol{f}}_h \cdot \mathbf{n} ds \approx \frac{|(\partial K)_j|}{2} \sum_{m=1}^{p+1} \left[\int_{-1}^{1} \widetilde{\phi}_i \overline{\phi}_m d\varsigma\right] (\widehat{\boldsymbol{f}}_h \cdot \mathbf{n})(\boldsymbol{x}(\varsigma_m)). \tag{B.1}$$

where $\varsigma(\boldsymbol{x})$ is a linear coordinate transformation mapping $\boldsymbol{x} \in (\partial K)_j$ to $\varsigma \in [-1,1]$ and $\{\varsigma_m\}$ denotes the set of interpolation nodes with the Gauss–Lobatto node distribution. Although all numerical results reported here are solved on the conforming meshes and orders, we note that the computer program used in the numerical tests also accommodates both non-conforming elements and non-conforming orders and supports dynamically h- and p-adaptive refinement. For non-conforming edges and/or order, the edge integrals are obtained through the use of a certain Gauss-quadrature. In both cases, the integral term on an edge segment is written as a multiplication of the matrix of dimension $M_p \times \bar{p}$ and the vector of dimension $\bar{p}$ (note that $\bar{p}$ denotes the number of points used in the integration and M_p the number of elemental basis functions). In the calculation of the flux and the numerical flux, the interpolated values of the solution, when needed, are realized through the multiplication of the appropriate Vandermonde matrix and a vector of modal expansion coordinates of the solution.

The computer code employed in the numerical simulations is written in Fortran 77 and Fortran 90. It is compiled using the Intel® Fortran version 11.1 compiler with the optimization flag set to `-O3`. Note that a significant portion of computing time is spent on performing matrix–vector multiplications. We make extensive use of the Fortran Basic Linear Algebra Subprogram (BLAS) level 2 [42] `DGEMV()` for matrix–vector multiplications. Such a subroutine involves $O(M \times N)$ operations where M and N are the dimensions of the matrix considered. Numerical computations are conducted on dual six-core

2.4 GHz AMD Opteron model 2431 64 bit 12 GB RAM nodes available at the Center for Research Computing at the University of Notre Dame.

Appendix C. Implementation of no-normal flow boundary condition

To impose the no-normal flow condition (43), we use an approach similar to that traditionally employed in weakly enforcing a so-called natural boundary condition in finite element methods. In this approach, the numerical flux on the no-normal flow boundary is defined as

$$\widehat{\mathbf{F}} = \mathbf{F}(\mathbf{q}^b) \tag{C.1}$$

where the state $\mathbf{q}^b = (H^b, u^b H^b, v^b H^b)^T$ is determined by setting

$$(\mathbf{u}^b H^b) \cdot \mathbf{n} = 0, \quad (\mathbf{u}^b H^b) \cdot \boldsymbol{\tau} = (\mathbf{u} H)^- \cdot \boldsymbol{\tau}, \quad H^b = H^- . \tag{C.2}$$

where $\boldsymbol{\tau}$ is the unit-tangential vector of the no-normal flow boundary. The minus superscript is used to indicate the value of variables on the boundary when approaching from the interior of the element. It can be easily verified that this setting amounts to using the following numerical flux on the no-normal flow boundary

$$\widehat{\mathbf{F}} \cdot \mathbf{n} = \left(0, \ \mathcal{F}^b n_x, \quad \mathcal{F}^b n_y\right)^T, \quad \mathcal{F}^b \equiv \frac{1}{2} g (H^-)^2 \tag{C.3}$$

where n_x and n_y denote the x- and y-component of the unit normal vector $\mathbf{n}$, respectively. It can be seen that (C.3) has a vanishing value of flux for the continuity equation, thus weakly enforcing (43). Note that this implementation does not require a Riemann solver.

References

[1] B. Cockburn, C.-W. Shu, Runge–Kutta discontinuous Galerkin methods for convection-dominated problems, J. Sci. Comput. 16 (2001) 173–261.
[2] B. Cockburn, Discontinuous Galerkin methods, Z. Angew. Math. Mech. 83 (2003) 731–754.
[3] V. Aizinger, C. Dawson, A discontinuous Galerkin method for two-dimensional flow and transport in shallow water, Adv. Water Res. 25 (2002) 67–84.
[4] C. Eskilsson, S.J. Sherwin, A triangular spectral/hp discontinuous Galerkin method for modelling 2D shallow water equations, Int. J. Numer. Methods Fluids 45 (2004) 605–623.
[5] E.J. Kubatko, J.J. Westerink, C. Dawson, hp Discontinuous Galerkin methods for advection dominated problems in shallow water flow, Comput. Methods Appl. Mech. Eng. 196 (2006) 437–451.
[6] E.J. Kubatko, S. Bunya, C. Dawson, J.J. Westerink, C. Mirabito, A performance comparison of continuous and discontinuous finite element shallow water models, J. Sci. Comput. 40 (2009) 315–339.
[7] F.X. Giraldo, T. Warburton, A high-order triangular discontinuous Galerkin oceanic shallow water model, Int. J. Numer. Methods Fluids 56 (7) (2008) 899–925.
[8] S. Bunya, E.J. Kubatko, J.J. Westerink, C. Dawson, A wetting and drying treatment for the Runge–Kutta discontinuous Galerkin solution to the shallow water equations, Comput. Methods Appl. Mech. Eng. 198 (2009) 1548–1562.
[9] C. Dawson, E.J. Kubatko, J.J. Westerink, C. Trahan, C. Mirabito, C. Michoski, N. Panda, Discontinuous Galerkin methods for modeling hurricane storm surge, Adv. Water Res. 34 (2010) 1165–1176, http://dx.doi.org/10.1016/j.advwatres.2010.11.004.
[10] T. Kärnä, B. de Brye, O. Gourgue, J. Lambrechts, R. Comblen, V. Legat, E. Deleersnijder, A fully implicit wetting–drying method for DG-FEM shallow water models, with an application to the scheldt estuary, Comput. Methods Appl. Mech. Eng. 200 (2011) 509–524.
[11] P.A. Tassi, C.A.V.S. Rhebergen, O. Bokhove, A discontinuous Galerkin finite element model for river bed evolution under shallow flows, Comput. Methods Appl. Mech. Eng. 197 (2008) 2930–2947.
[12] C. Michoski, C. Mirabito, C. Dawson, D. Wirasaet, E.J. Kubatko, J.J. Westerink, Dynamic *p*-enrichment schemes for multicomponent reactive flows, Adv. water res. 34 (2011) 1666–1680.
[13] C. Dawson, J.J. Westerink, J.C. Feyen, D. Pothina, Continuous, discontinuous, and coupled discontinuous–continuous Galerkin finite elements methods for the shallow water equations, Int. J. Numer. Methods Fluids 52 (2006) 63–88.
[14] J.S. Hesthaven, T. Warburton, Nodal high-order methods on unstructured grids I. Time-domain solution of Maxwell's equations, J. Comput. Phys. 181 (2002) 186–221.
[15] G.J. Gassner, F. Lörcher, C. Munz, J.S. Hesthaven, Polymorphic nodal elements and their application in discontinuous Galerkin methods, J. Comput. Phys. 228 (2009) 1573–1590.
[16] R.J. LeVeque, Balancing source terms and flux gradients in high-resolution Godunov methods: the quasi-steady wave-propagation algorithm, J. Comput. Phys. 146 (1) (1998) 346–365.
[17] E. Audusse, F. Bouchut, M. Bristeau, R. Klein, B. Perthame, A fast and stable well-balanced scheme with hydrostatic reconstruction for shallow water flows, SIAM J. Sci. Comput. 25 (6) (2004) 2050–2065.
[18] A. Ern, S. Piperno, K. Djadel, A well-balanced Runge–Kutta discontinuous Galerkin method for the shallow-water equations with flooding and drying, Int. J. Numer. Methods Fluids 58 (1) (2008) 1–25.
[19] Y. Xing, X. Zhang, C.-W. Shu, Positivity-preserving high order well-balanced discontinuous Galerkin methods for the shallow water equations, Adv. Water Res. 33 (2010) 1476–1493.
[20] Y. Xing, X. Zhang, Positivity-preserving well-balanced discontinuous Galerkin methods for the shallow water equations on unstructured triangular meshes, J. Sci. Comput. 57 (2013) 19–41.
[21] G. Kesserwani, Q. Liang, A conservative high-order discontinuous Galerkin method for the shallow water equations with arbitrary topography, Int. J. Numer. Methods Eng. 86 (1) (2011) 47–69.
[22] R.J. LeVeque, Finite Volume Methods for Hyperbolic Problems, Cambridge University Press, Cambridge, 2002.
[23] J. Qiu, B.C. Khoo, C.-W. Shu, A numerical study of the performance of the Runge–Kutta discontinuous Galerkin method based on different numerical fluxes, J. Comput. Phys. 212 (2006) 540–565.
[24] D. Gottlieb, S.A. Orszag, Numerical Analysis of Spectral Methods: Theory and Applications, SIAM, 1987.
[25] J.S. Hesthaven, T. Warburton, Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Application, Springer Science+Business Media, LLC, 2008.
[26] H.L. Atkins, C.-W. Shu, Quadrature-free implementation of discontinuous Galerkin method for hyperbolic equations, AIAA J. 36 (1998) 775–782.

[27] G.H. Golub, C.F. van Loan, Matrix Computations, third ed., The Johns Hopkins University Press, 1996.
[28] A. Quarteroni, R. Sacco, F. Saleri, Numerical Mathematics, Springer, 2000.
[29] J.S. Hesthaven, From electrostatics to almost optimal nodal sets for polynomial in simplex, SIAM J. Numer. Anal. 35 (1998) 655–676.
[30] D. Wirasaet, S. Tanaka, E.J. Kubatko, J.J. Westerink, C. Dawson, A performance comparison of nodal discontinuous Galerkin methods on triangles and quadrilaterals, Int. J. Numer. Methods Fluids 64 (2010) 1326–1362.
[31] J.C. Butcher, Numerical Methods for Ordinary Differential Equations, John Wiley & Sons, 2003.
[32] L.F. Shampine, Error estimation and control for ODEs, J. Sci. Comput. 25 (2005) 3–16.
[33] L.F. Shampine, H.A. Watts, S.M. Davenport, Solving nonstiff ordinary differential equations–state of art, SIAM Rev. 18 (1976) 376–411. <http://www.netlib.org/fmm/rkf45.f>.
[34] Q. Zhang, C. Shu, Error estimates to smooth solutions of Runge–Kutta discontinuous Galerkin methods for scalar conservative laws, SIAM J. Numer. Anal. 42 (2004) 641–666.
[35] M. de Berg, O. Cheong, M. van Kreveld, M. Overmars, Computational Geometry: Algorithms and Applications, third ed., Springer, 2010.
[36] E. Bernsen, O. Bokhove, J.J.W. van der Vegt, A (dis)continuous finite element model for generalized 2d vorticity dynamics, J. Comput. Phys. 211 (2) (2006) 719–747.
[37] F.X. Giraldo, M. Resteli, High-order semi-implicit time-integrators for a triangular discontinuous Galerkin oceanic shallow water model, Int. J. Numer. Methods Fluids 63 (2010) 1077–1102.
[38] H. Stommel, The westward intensification of wind-driven ocean currents, Trans. Am. Geophys. Union 29 (1948) 202–206.
[39] R. Cools, Monomail cubature rules since Stroud: a compilation – part 2, J. Comput. Appl. Math. 112 (1999) 21–27.
[40] Z.J. Wang1, K. Fidkowski2, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, H.T. Huynh, N. Kroll, G. May, P. Persson, B. van Leer, M. Visbal, High-order CFD methods: current status and perspective, Int. J. Numer. Methods Fluids 72 (2013) 811–845.
[41] D. Wirasaet, S.R. Brus, C.E. Michoski, E.J. Kubatkob, J.J. Westerink, C. Dawson, Artificial boundary layers in discontinuous Galerkin solutions to shallow water equations in channels, J. Comput. Phys. (2013), submitted for publication.
[42] J.J. Dongarra, J.D. Croz, S.H. Hammarling, R.J. Hanson, An extended set of Fortran basic linear algebra subprograms, Technical Memorandum 41, Argonne National Laboratory, Argonne, IL (1998), see also URL <http://www.netlib.org/blas>