Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational Physics 212 (2006) 566–589

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

Spectral/*hp* discontinuous Galerkin methods for modelling 2D Boussinesq equations

Claes Eskilsson [a], Spencer J. Sherwin [b,*]

[a] *Department of Civil and Environmental Engineering, Chalmers University of Technology, SE-412 96 Gothenburg, Sweden*
[b] *Department of Aeronautics, Imperial College London, South Kensington Campus, London, SW7 2AZ, UK*

Received 3 December 2004; received in revised form 20 June 2005; accepted 13 July 2005
Available online 16 September 2005

Abstract

We present spectral/*hp* discontinuous Galerkin methods for modelling weakly nonlinear and dispersive water waves, described by a set of depth-integrated Boussinesq equations, on unstructured triangular meshes. When solving the equations two different formulations are considered: directly solving the coupled momentum equations and the 'scalar method', in which a wave continuity equation is solved as an intermediate step. We demonstrate that the approaches are fully equivalent and give identical results in terms of accuracy, convergence and restriction on the time step. However, the scalar method is shown to be more CPU efficient for high order expansions, in addition to requiring less storage.
© 2005 Elsevier Inc. All rights reserved.

Keywords: Nonlinear dispersive water waves; Boussinesq-type equations; Discontinuous Galerkin method; Spectral/*hp* discretization

1. Introduction

Simulation of nonlinear and dispersive waves are routinely performed in coastal and ocean engineering for solving problems in the nearshore zone, such as harbour resonance and wave-induced circulation, e.g. [26,39,48]. This type of simulation is increasingly carried out in the time-domain using numerical models based on Boussinesq-type equations, sometimes referred to as dispersive shallow water systems. Over the last decade significant advancement has been made in deriving more and more accurate – and more complex – Boussinesq-type equations [29,33,31,19,32].

[*] Corresponding author. Tel.: +44 207 594 5052; fax: +44 207 584 8120.
E-mail addresses: claes.eskilsson@chalmers.se (C. Eskilsson), s.sherwin@imperial.ac.uk, (S.J. Sherwin).

doi:10.1016/j.jcp.2005.07.017

Boussinesq-type equations are higher order approximations to the nonlinear shallow water equations (SWE). There is a growing body of work concerning application of high-order Galerkin methods to the SWE. The motivation being the potential savings in computational time – especially for large-scale problems involving long-time integration – in combination with the geometrical flexibility of the Galerkin approach. In 1993, Ma [28] presented the first continuous spectral element model and was soon to be followed by other spectral element modellers, including [21,41,17]. More recently spectral/*hp* discontinuous Galerkin (DG) models have been suggested [12,18,14,16].

The salient feature in DG methods is that the solution is allowed to be discontinuous over elemental boundaries, while the elements are coupled through the use of continuous so-called numerical fluxes. See [8] for a general overview and [9] for a review of the Runge–Kutta DG method. Lately DG methods has been applied to nonlinear and dispersive wave equations, e.g. KdV-type equations [51,49], Schrödinger equations [50] and Boussinesq-type equations [13,15].

In light of the interest in Boussinesq-type modelling in coastal engineering spectral/*hp* element methods have received surprisingly little interest. Despite the fact that the introduction of dispersive terms will require a higher spatial resolution, typically low-order finite differences [29,33,45,38], or finite elements [2,1,25,43,47,40] are used. The authors recently outlined a 2D spectral/*hp* DG model for the Boussinesq equations [15] – with the restriction of constant depth and periodic boundary conditions. In this work the concept of a 'scalar method' was proposed for treating the dispersive terms arising in these equations. The motivation behind the scalar method is to reduce the size of the matrix system by introducing the time rate of change of momentum divergence as a dependent variable – resulting in an $N_{\mathrm{dof}} \times N_{\mathrm{dof}}$ scalar Helmholtz problem (N_{dof} denoting total degrees of freedom). This can be compared to the more traditional approach of solving the coupled momentum equations leading to a $2N_{\mathrm{dof}} \times 2N_{\mathrm{dof}}$ matrix system. On the other hand, the scalar method requires a variable recovery step and, for the present DG implementation, several additional numerical flux evaluations.

Thus, in addition to extending the DG Boussinesq model to the more general setting of variable depth and non-periodic boundary conditions, we in this paper focus on investigating the effect on accuracy and efficiency of using the scalar method rather than solving the coupled momentum equations. This is done through examining the eigenspectra of the semi-discretized equations, as well as through numerical experiments. Further, we: (i) consider different elliptic DG formulations for the dispersive part, and (ii) compare the DG model against a finite difference model in terms of efficiency.

The paper is organised as follows. In Section 2, we present the governing equations. Section 3 is devoted to the numerical methods and is divided into several subsections dealing with the discrete Galerkin approximations, the advective and dispersive numerical fluxes, boundary conditions, expansion basis, the time stepping scheme and eigenspectra. In Section 4, we numerically evaluate the accuracy and efficiency of the models by examining the case of a linear standing wave. Here, we also compare the DG model against the finite difference model. We present additional computational examples in Section 5. The findings of the work are then summarised in Section 6.

2. Governing equations

For an incompressible, irrotational and inviscid fluid the wave motion is described by the Laplace equation with appropriate boundary conditions. A major difficulty in solving the full problem is that the location of the water surface – which is a boundary – is a priori unknown, giving that the domain in which the equations are to be solved is also unknown. The moving boundary problem of the free surface can be avoided by using the Boussinesq approach, i.e., expanding the velocity potential in powers of the vertical coordinate and integrating the Laplace equation over the water depth. By using the Boussinesq approach a simpler problem that approximates the full problem, expressed only in the horizontal dimensions, is obtained. The Boussinesq equations used in this study are valid for $d/L_0 \leqslant 0.22$, where d is the water depth and L_0 is the deep water wavelength [29] (the traditional deep water limit corresponds to $d/L_0 = 0.5$).

We write the Boussinesq equations [34] in terms of conservative variables as

$$\partial_t \mathbf{U} + \partial_t \mathbf{D}(\mathbf{U}) + \nabla \cdot \mathbf{F}(\mathbf{U}) = \mathbf{S}(\mathbf{U}), \tag{1}$$

in which $\mathbf{F}(\mathbf{U}) = [\mathbf{E}(\mathbf{U}), \mathbf{G}(\mathbf{U})]^{\mathrm{T}}$ is the flux vector and

$$\mathbf{U} = \begin{bmatrix} H \\ Hu \\ Hv \end{bmatrix}, \quad \mathbf{D} = \begin{bmatrix} 0 \\ (d^3/6)\partial_x(\nabla \cdot (H\mathbf{u}/d)) - (d^2/2)\partial_x(\nabla \cdot (H\mathbf{u})) \\ (d^3/6)\partial_y(\nabla \cdot (H\mathbf{u}/d)) - (d^2/2)\partial_y(\nabla \cdot (H\mathbf{u})) \end{bmatrix}, \tag{2a}$$

$$\mathbf{E} = \begin{bmatrix} Hu \\ Hu^2 + gH^2/2 \\ Huv \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} Hv \\ Huv \\ Hv^2 + gH^2/2 \end{bmatrix}. \tag{2b}$$

The vector $\mathbf{U}$ contains the conserved variables, where $H(\mathbf{x}, t) = \zeta(\mathbf{x}, t) + d(\mathbf{x})$ is the total water depth, $d(\mathbf{x})$ is the still water depth and $\zeta(\mathbf{x}, t)$ is the free surface elevation. $\mathbf{u} = [u(\mathbf{x}, t), v(\mathbf{x}, t)]^{\mathrm{T}}$ denotes the depth-averaged velocities in the x- and y-direction, respectively. The acceleration of gravity is denoted by g. We note that the SWE are recovered if $\mathbf{D} \equiv \mathbf{0}$. In this paper the source term $\mathbf{S}(\mathbf{U})$ accounts for forcing due to bed slopes, i.e.

$$\mathbf{S} = \begin{bmatrix} 0 \\ gHS_0^x \\ gHS_0^y \end{bmatrix}, \tag{2c}$$

where S_0^x and S_0^y are the bed slopes in the x- and y-direction, respectively.

Expanding the parenthesis in the dispersive terms, leads to

$$\partial_t \mathbf{D} = \begin{bmatrix} 0 \\ -(d^2/3)\partial_x(\nabla \cdot \partial_t(H\mathbf{u})) - (d/6)\partial_x d(\nabla \cdot \partial_t(H\mathbf{u})) - (d^3/6)\partial_x((\nabla d \cdot \partial_t(H\mathbf{u}))/d^2) \\ -(d^2/3)\partial_y(\nabla \cdot \partial_t(H\mathbf{u})) - (d/6)\partial_y d(\nabla \cdot \partial_t(H\mathbf{u})) - (d^3/6)\partial_y((\nabla d \cdot \partial_t(H\mathbf{u}))/d^2) \end{bmatrix}. \tag{3}$$

We write the dispersive term as $\partial_t \mathbf{D}(\mathbf{U}) = \partial_t \mathbf{D}^{\mathbf{m}}(\mathbf{U}) + \mathbf{D}^{\mathbf{s}}(\mathbf{U})$, where

$$\mathbf{D}^{\mathbf{m}}(\mathbf{U}) = \begin{bmatrix} 0 \\ -(d^2/3)\partial_x(\nabla \cdot (H\mathbf{u})) - (d/3)\partial_x d(\nabla \cdot (H\mathbf{u})) \\ -(d^2/3)\partial_y(\nabla \cdot (H\mathbf{u})) - (d/3)\partial_y d(\nabla \cdot (H\mathbf{u})) \end{bmatrix} \tag{4}$$

and, using the mild-slope assumption,

$$\mathbf{D}^{\mathbf{s}}(\mathbf{U}) = \begin{bmatrix} 0 \\ (d\partial_x d/6)\partial_{yt}(Hv) - (d\partial_y d/6)\partial_{xt}(Hv) \\ (d\partial_y d/6)\partial_{xt}(Hu) - (d\partial_x d/6)\partial_{yt}(Hu) \end{bmatrix}. \tag{5}$$

Now, we apply the linear long wave approximation

$$\partial_t(H\mathbf{u}) \approx -gd\nabla\zeta, \tag{6}$$

to rewrite the $\mathbf{D}^{\mathbf{s}}$ terms as

$$\mathbf{D}^{\mathrm{s}}(\mathbf{U}) = \begin{bmatrix} 0 \\ (d\partial_x d/6)\partial_y(-gd\partial_y\zeta) - (d\partial_y d/6)\partial_x(-gd\partial_y\zeta) \\ (d\partial_y d/6)\partial_x(-gd\partial_x\zeta) - (d\partial_x d/6)\partial_y(-gd\partial_x\zeta) \end{bmatrix}. \tag{7}$$

3. Numerical schemes

The most obvious way to solve the Boussinesq equations is to directly discretize (1) and solve the coupled momentum equations (from here on referred to as the coupled method) as outlined in Section 3.2. An alternative way is to use the scalar method as discussed in Section 3.3.

3.1. Preliminaries

Let $\mathscr{T}_h$ be a partition of the domain Ω into N triangular elements $\mathscr{T}_e$. The element $\mathscr{T}_e$ has a boundary $\partial\mathscr{T}_e$. For each element we denote the diameter of $\mathscr{T}_e$ with h_e and set $h = \max(h_1, h_2, \ldots, h_N)$. We also define the following discrete spaces

$$\mathscr{V}_\delta = \left\{ v \in L^2(\Omega) : v|_{\mathscr{T}_e} \in \mathscr{P}^P(\mathscr{T}_e),\ \forall \mathscr{T}_e \in \mathscr{T}_h \right\}, \tag{8a}$$

$$\mathscr{W}_\delta = \left\{ \mathbf{w} \in (L^2(\Omega))^2 : \mathbf{w}|_{\mathscr{T}_e} \in (\mathscr{P}^P(\mathscr{T}_e))^2,\ \forall \mathscr{T}_e \in \mathscr{T}_h \right\}, \tag{8b}$$

where $\mathscr{P}^P(\mathscr{T}_e)$ is the space of polynomials of degree at most P in the element $\mathscr{T}_e$.

The variables will be approximated using a polynomial expansion basis $\phi_{pq}(\xi_1, \xi_2)$. For an arbitrary function $f(\mathbf{x}, t)$ the approximation $f_\delta \in \mathscr{V}_\delta$ reads

$$f_\delta(\mathbf{x}, t) = \sum_{p=0}^{P} \sum_{q=0}^{P-p} \tilde{f}_{pq}(t)\phi_{pq}(\xi_1, \xi_2), \quad \mathbf{x} \in \mathscr{T}_e, \tag{9}$$

where $\tilde{f}_{pq}(t)$ are the local expansion coefficients. The orthogonal modal basis ϕ_{pq} used in the current work is discussed in Section 3.6.

3.2. Coupled method

The variational form of Eq. (1) is obtained by approximating $\mathbf{U}$ with a polynomial expansion $\mathbf{U}_\delta \in \mathscr{V}_\delta$, multiplying with a test function $q_\delta \in \mathscr{V}_\delta$ and integrating over the element $\mathscr{T}_e$

$$\int_{\mathscr{T}_e} q_\delta \partial_t(\mathbf{U}_\delta + \mathbf{D}^{\mathbf{m}}(\mathbf{U}_\delta))\,\mathrm{d}\mathbf{x} + \int_{\mathscr{T}_e} q_\delta \mathbf{D}^{\mathrm{s}}(\mathbf{U}_\delta)\,\mathrm{d}\mathbf{x} + \int_{\mathscr{T}_e} q_\delta(\nabla \cdot \mathbf{F}(\mathbf{U}_\delta))\,\mathrm{d}\mathbf{x} = \int_{\mathscr{T}_e} q_\delta \mathbf{S}(\mathbf{U}_\delta)\,\mathrm{d}\mathbf{x}. \tag{10}$$

Integration by parts gives the weak formulation

$$\begin{aligned} &\int_{\mathscr{T}_e} q_\delta \partial_t(\mathbf{U}_\delta + \mathbf{D}^{\mathbf{m}}(\mathbf{U}_\delta))\,\mathrm{d}\mathbf{x} + \int_{\mathscr{T}_e} q_\delta \mathbf{D}^{\mathrm{s}}(\mathbf{U}_\delta)\,\mathrm{d}\mathbf{x} - \int_{\mathscr{T}_e} \nabla q_\delta \cdot \mathbf{F}(\mathbf{U}_\delta)\,\mathrm{d}\mathbf{x} + \int_{\partial\mathscr{T}_e} q_\delta(\mathbf{F}(\mathbf{U}_\delta) \cdot \mathbf{n})\,\mathrm{d}S \\ &\quad = \int_{\mathscr{T}_e} q_\delta \mathbf{S}(\mathbf{U}_\delta)\,\mathrm{d}\mathbf{x}, \end{aligned} \tag{11}$$

where $\mathbf{n} = [n_{\mathrm{x}}, n_{\mathrm{y}}]^{\mathrm{T}}$ is the outward unit normal to $\partial\mathscr{T}_e$. We exchange the flux $\mathbf{F}(\mathbf{U}_\delta)$ in the boundary term with a numerical flux denoted with a hat, $\hat{\mathbf{F}}(\mathbf{U}_\delta)$, defined in Section 3.4.1. Integrating by parts once more and we can write the discrete DG formulation as: find $\mathbf{U}_\delta \in \mathscr{V}_\delta$ such that for all $q_\delta \in \mathscr{V}_\delta$ and for all $\mathscr{T}_e \in \mathscr{T}_h$

$$\int_{\mathscr{T}_e} q_\delta \partial_t(\mathbf{U}_\delta + \mathbf{D}^{\mathbf{m}}(\mathbf{U}_\delta))\,\mathrm{d}\mathbf{x} + \int_{\mathscr{T}_e} q_\delta \mathbf{D}^{\mathbf{s}}(\mathbf{U}_\delta)\,\mathrm{d}\mathbf{x} + \int_{\mathscr{T}_e} q_\delta(\nabla\cdot\mathbf{F}(\mathbf{U}_\delta))\,\mathrm{d}\mathbf{x}$$
$$+ \int_{\partial\mathscr{T}_e} q_\delta\Big(\Big(\hat{\mathbf{F}}(\mathbf{U}_\delta) - \mathbf{F}(\mathbf{U}_\delta)\Big)\cdot\mathbf{n}\Big)\,\mathrm{d}S = \int_{\mathscr{T}_e} q_\delta \mathbf{S}(\mathbf{U}_\delta)\,\mathrm{d}\mathbf{x}. \tag{12}$$

Introducing the auxiliary variable $a = \nabla\cdot(H\mathbf{u})$ we rewrite the $\mathbf{D}^{\mathbf{m}}_{(i)}$ $(i = 1,2,3)$ term as a first-order system:

$$\mathbf{D}^{\mathbf{m}}_{(2,3)}(\mathbf{U}) = -\gamma\nabla a - 2\boldsymbol{\Gamma}a, \tag{13a}$$
$$a = \nabla\cdot(H\mathbf{u}), \tag{13b}$$

where $\gamma(\mathbf{x}) = d(\mathbf{x})^2/3$ and $\boldsymbol{\Gamma}(\mathbf{x}) = d(\mathbf{x})\nabla d(\mathbf{x})/6$. The integral containing $\mathbf{D}^{\mathbf{m}}$ in Eq. (12) is thus evaluated as

$$\int_{\mathscr{T}_e} q_\delta \mathbf{D}^{\mathbf{m}}_{(2,3)}(\mathbf{U}_\delta)\,\mathrm{d}\mathbf{x} = -\int_{\mathscr{T}_e} q_\delta\gamma_\delta\nabla a_\delta\,\mathrm{d}\mathbf{x} - \int_{\partial\mathscr{T}_e} q_\delta\gamma_\delta(\hat{a}_\delta - a_\delta)\mathbf{n}\,\mathrm{d}S - 2\int_{\mathscr{T}_e} q_\delta\boldsymbol{\Gamma}_\delta a_\delta\,\mathrm{d}\mathbf{x}, \tag{14a}$$
$$\int_{\mathscr{T}_e} q_\delta a_\delta\,\mathrm{d}\mathbf{x} = \int_{\mathscr{T}_e} q_\delta(\nabla\cdot(H\mathbf{u})_\delta)\,\mathrm{d}\mathbf{x} + \int_{\partial\mathscr{T}_e} q_\delta\Big(\Big((\widehat{H\mathbf{u}})_\delta - (H\mathbf{u})_\delta\Big)\cdot\mathbf{n}\Big)\,\mathrm{d}S, \tag{14b}$$

where the dispersive numerical fluxes $\hat{a}_\delta$ and $(\widehat{H\mathbf{u}})_\delta$ will be discussed in Section 3.4.2. As an aside we note that at the discrete level a_δ in (14b) is directly substituted into (14a) and therefore does not need to be explicitly solved.

By introducing a further auxiliary variable $\mathbf{b} = -gd\nabla\zeta$ we likewise rewrite the $\mathbf{D}^{\mathbf{s}}_i$ term as

$$\mathbf{D}^{\mathbf{s}}_{(2)}(\mathbf{U}) = \boldsymbol{\Gamma}_{(1)}\partial_y\mathbf{b}_{(2)} - \boldsymbol{\Gamma}_{(2)}\partial_x\mathbf{b}_{(2)}, \tag{15a}$$
$$\mathbf{D}^{\mathbf{s}}_{(3)}(\mathbf{U}) = \boldsymbol{\Gamma}_{(2)}\partial_x\mathbf{b}_{(1)} - \boldsymbol{\Gamma}_{(1)}\partial_y\mathbf{b}_{(1)}, \tag{15b}$$
$$\mathbf{b} = -gd\nabla\zeta. \tag{15c}$$

Hence the integral containing $\mathbf{D}^{\mathbf{s}}$ in Eq. (12) is evaluated as

$$\int_{\mathscr{T}_e} q_\delta\mathbf{D}^{\mathbf{s}}_{(2)}(\mathbf{U}_\delta)\,\mathrm{d}\mathbf{x} = \int_{\mathscr{T}_e} q_\delta\boldsymbol{\Gamma}_{(1)\delta}\partial_y\mathbf{b}_{(2)\delta}\,\mathrm{d}\mathbf{x} + \int_{\partial\mathscr{T}_e} q_\delta\boldsymbol{\Gamma}_{(1)\delta}(\hat{\mathbf{b}}_{(2)\delta} - \mathbf{b}_{(2)\delta})n_{\mathrm{y}}\,\mathrm{d}S$$
$$- \int_{\mathscr{T}_e} q_\delta\boldsymbol{\Gamma}_{(2)\delta}\partial_x\mathbf{b}_{(2)\delta}\,\mathrm{d}\mathbf{x} - \int_{\partial\mathscr{T}_e} q_\delta\boldsymbol{\Gamma}_{(2)\delta}(\hat{\mathbf{b}}_{(2)\delta} - \mathbf{b}_{(2)\delta})n_{\mathrm{x}}\,\mathrm{d}S, \tag{16a}$$
$$\int_{\mathscr{T}_e} q_\delta\mathbf{D}^{\mathbf{s}}_{(3)}(\mathbf{U}_\delta)\,\mathrm{d}\mathbf{x} = \int_{\mathscr{T}_e} q_\delta\boldsymbol{\Gamma}_{(2)\delta}\partial_x\mathbf{b}_{(1)\delta}\,\mathrm{d}\mathbf{x} + \int_{\partial\mathscr{T}_e} q_\delta\boldsymbol{\Gamma}_{(2)\delta}(\hat{\mathbf{b}}_{(1)\delta} - \mathbf{b}_{(1)\delta})n_{\mathrm{x}}\,\mathrm{d}S$$
$$- \int_{\mathscr{T}_e} q_\delta\boldsymbol{\Gamma}_{(1)\delta}\partial_y\mathbf{b}_{(1)\delta}\,\mathrm{d}\mathbf{x} - \int_{\partial\mathscr{T}_e} q_\delta\boldsymbol{\Gamma}_{(1)\delta}(\hat{\mathbf{b}}_{(1)\delta} - \mathbf{b}_{(1)\delta})n_{\mathrm{y}}\,\mathrm{d}S, \tag{16b}$$
$$\int_{\mathscr{T}_e} q_\delta\mathbf{b}_\delta\,\mathrm{d}\mathbf{x} = -g\int_{\mathscr{T}_e} q_\delta d_\delta\nabla\zeta_\delta\,\mathrm{d}\mathbf{x} - g\int_{\partial\mathscr{T}_e} q_\delta d_\delta(\hat{\zeta}_\delta - \zeta_\delta)\mathbf{n}\,\mathrm{d}S, \tag{16c}$$

where once again the additional dispersive fluxes $\hat{\zeta}_\delta$ and $\hat{\mathbf{b}}_\delta$ will be discussed in Section 3.4.2.

Following the standard Galerkin formulation the test functions q_δ are represented by ϕ_{pq}. We denote the global mass matrix with $\mathbf{M}$ and the global weak derivative matrices with $\mathbf{D_x}$ and $\mathbf{D_y}$. In the derivative matrices we incorporate the numerical fluxes, and the superscripts $\mathbf{a}$ and $\mathbf{d}$ are used to distinguish between advective and dispersive numerical fluxes. If we for simplicity of presentation consider the constant depth case the coupled system can be written in matrix form as

$$\begin{bmatrix} \mathbf{M} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}-\gamma\mathbf{D}^{\mathbf{d}}_{\mathbf{xx}} & -\gamma\mathbf{D}^{\mathbf{d}}_{\mathbf{xy}} \\ \mathbf{0} & -\gamma\mathbf{D}^{\mathbf{d}}_{\mathbf{yx}} & \mathbf{M}-\gamma\mathbf{D}^{\mathbf{d}}_{\mathbf{yy}} \end{bmatrix} \begin{bmatrix} \partial_t\tilde{H} \\ \partial_t(\widetilde{Hu}) \\ \partial_t(\widetilde{Hv}) \end{bmatrix} = \begin{bmatrix} -\mathbf{D}^{\mathbf{a}}_{\mathbf{x}}\tilde{\mathbf{E}}_{(1)} - \mathbf{D}^{\mathbf{a}}_{\mathbf{y}}\tilde{\mathbf{G}}_{(1)} \\ -\mathbf{D}^{\mathbf{a}}_{\mathbf{x}}\tilde{\mathbf{E}}_{(2)} - \mathbf{D}^{\mathbf{a}}_{\mathbf{y}}\tilde{\mathbf{G}}_{(2)} \\ -\mathbf{D}^{\mathbf{a}}_{\mathbf{x}}\tilde{\mathbf{E}}_{(3)} - \mathbf{D}^{\mathbf{a}}_{\mathbf{y}}\tilde{\mathbf{G}}_{(3)} \end{bmatrix}. \tag{17}$$

Here $\mathbf{D}^{\mathbf{d}}_{\mathbf{xx}} = \mathbf{D}^{\mathbf{d}}_{\mathbf{x}}\mathbf{M}^{-1}\mathbf{D}^{\mathbf{d}}_{\mathbf{x}}$, $\mathbf{D}^{\mathbf{d}}_{\mathbf{xy}} = \mathbf{D}^{\mathbf{d}}_{\mathbf{x}}\mathbf{M}^{-1}\mathbf{D}^{\mathbf{d}}_{\mathbf{y}}$, $\mathbf{D}^{\mathbf{d}}_{\mathbf{yx}} = \mathbf{D}^{\mathbf{d}}_{\mathbf{y}}\mathbf{M}^{-1}\mathbf{D}^{\mathbf{d}}_{\mathbf{x}}$ and $\mathbf{D}^{\mathbf{d}}_{\mathbf{yy}} = \mathbf{D}^{\mathbf{d}}_{\mathbf{y}}\mathbf{M}^{-1}\mathbf{D}^{\mathbf{d}}_{\mathbf{y}}$. We observe that with a suitable choice of basis functions, see Section 3.6, the mass matrix is diagonal for regular shaped triangles.

3.3. Scalar method

In the scalar method we begin by formally grouping the mixed derivatives with the time derivatives. All terms not containing any time derivative are collected into the term $\mathbf{f}(\mathbf{U})$. We write Eq. (1) as

$$\mathbf{f}(\mathbf{U}) = -\mathbf{D}^{\mathbf{s}}(\mathbf{U}) - \nabla\cdot\mathbf{F}(\mathbf{U}) + \mathbf{S}(\mathbf{U}), \tag{18a}$$

$$\partial_t\mathbf{U} + \partial_t\mathbf{D}^{\mathbf{m}}(\mathbf{U}) = f(\mathbf{U}). \tag{18b}$$

The DG method for computing $\mathbf{f}(\mathbf{U})$ is stated as: find $\mathbf{f}(\mathbf{U}_\delta) \in \mathscr{V}_\delta$ such that for all $q_\delta \in \mathscr{V}_\delta$ and for all $\mathscr{T}_e \in \mathscr{T}_h$

$$\int_{\mathscr{T}_e} q_\delta\mathbf{f}(\mathbf{U}_\delta)\mathrm{d}\mathbf{x} = -\int_{\mathscr{T}_e} q_\delta\mathbf{D}^{\mathbf{s}}(\mathbf{U}_\delta)\,\mathrm{d}\mathbf{x} - \int_{\mathscr{T}_e} q_\delta(\nabla\cdot\mathbf{F}(\mathbf{U}_\delta))\,\mathrm{d}\mathbf{x} - \int_{\partial\mathscr{T}_e} q_\delta((\hat{\mathbf{F}}(\mathbf{U}_\delta) - \mathbf{F}(\mathbf{U}_\delta))\cdot\mathbf{n})\,\mathrm{d}S + \int_{\mathscr{T}_e} q_\delta\mathbf{S}(\mathbf{U}_\delta)\,\mathrm{d}\mathbf{x}, \tag{19}$$

where the dispersive $\mathbf{D}^{\mathbf{s}}$ term is treated as in the coupled method in Section 3.2.

As there are no dispersive terms present in the continuity equation, we write the remaining components of Eq. (18b) as

$$\partial_t(H\mathbf{u}) - \gamma\nabla(\nabla\cdot\partial_t(H\mathbf{u})) - 2\mathbf{\Gamma}\nabla\cdot\partial_t(H\mathbf{u}) = \mathbf{f}_{(2,3)}(\mathbf{U}). \tag{20}$$

Introducing the time rate of change of momentum divergence as a new scalar variable, i.e., $z = \nabla\cdot\partial_t(H\mathbf{u})$, an equivalent statement to problem (20) is

$$\partial_t(H\mathbf{u}) = \gamma\nabla z + 2\mathbf{\Gamma}z + \mathbf{f}_{(2,3)}(\mathbf{U}), \tag{21a}$$

$$z - \nabla\cdot\partial_t(H\mathbf{u}) = 0. \tag{21b}$$

Substituting (21a) into (21b), we obtain an advection–diffusion type equation

$$\nabla\cdot(\gamma\nabla z) + \nabla\cdot(2\mathbf{\Gamma}z) - z = -\nabla\cdot\mathbf{f}_{(2,3)}(\mathbf{U}). \tag{22}$$

Eq. (22) constitute a 'wave continuity equation' [27] for the Boussinesq equations. To solve (22) we can rewrite the equation as a first-order system by introducing the auxiliary variables $\mathbf{w} = \gamma\nabla z$ and $\mathbf{v} = 2\mathbf{\Gamma}z$, i.e.

$$\nabla\cdot\mathbf{w} + \nabla\cdot\mathbf{v} - z = -\nabla\cdot\mathbf{f}_{(2,3)}(\mathbf{U}), \tag{23a}$$

$$\mathbf{w} = \gamma\nabla z, \tag{23b}$$

$$\mathbf{v} = 2\mathbf{\Gamma}z. \tag{23c}$$

The DG formulation finally reads: find $(z_\delta, \mathbf{w}_\delta, \mathbf{v}_\delta) \in \mathscr{V}_\delta \times \mathscr{W}_\delta \times \mathscr{W}_\delta$ such that for all $(s_\delta, \mathbf{r}_\delta, \mathbf{t}_\delta) \in \mathscr{V}_\delta \times \mathscr{W}_\delta \times \mathscr{W}_\delta$ and for all $\mathscr{T}_e \in \mathscr{T}_h$

$$\begin{aligned}
&\int_{\mathscr{T}_e} s_\delta(\nabla\cdot\mathbf{w}_\delta)\,\mathrm{d}\mathbf{x} + \int_{\partial\mathscr{T}_e} s_\delta((\hat{\mathbf{w}}_\delta-\mathbf{w}_\delta)\cdot\mathbf{n})\,\mathrm{d}S + \int_{\mathscr{T}_e} s_\delta(\nabla\cdot\mathbf{v}_\delta)\,\mathrm{d}\mathbf{x} + \int_{\partial\mathscr{T}_e} s_\delta((\hat{\mathbf{v}}_\delta-\mathbf{v}_\delta)\cdot\mathbf{n})\,\mathrm{d}S - \int_{\mathscr{T}_e} s_\delta z_\delta\,\mathrm{d}\mathbf{x}\\
&\quad = -\int_{\mathscr{T}_e} s_\delta(\nabla\cdot\mathbf{f}_{(2,3)}(\mathbf{U}_\delta))\,\mathrm{d}\mathbf{x} - \int_{\partial\mathscr{T}_e} s_\delta((\hat{\mathbf{f}}_{(2,3)}(\mathbf{U}_\delta)-\mathbf{f}_{(2,3)}(\mathbf{U}_\delta))\cdot\mathbf{n})\,\mathrm{d}S, \qquad (24a)\\
&\int_{\mathscr{T}_e} \mathbf{r}_\delta\cdot\mathbf{w}_\delta\,\mathrm{d}\mathbf{x} = \int_{\mathscr{T}_e} \mathbf{r}_\delta\cdot\gamma_\delta\nabla z_\delta \mathrm{d}\mathbf{x} + \int_{\partial\mathscr{T}_e} (\mathbf{r}_\delta\gamma_\delta(\hat{z}_\delta - z_\delta))\cdot\mathbf{n}\,\mathrm{d}S, \qquad (24b)\\
&\int_{\mathscr{T}_e} \mathbf{t}_\delta\cdot\mathbf{v}_\delta\,\mathrm{d}\mathbf{x} = 2\int_{\mathscr{T}_e} (\mathbf{t}_\delta\cdot\mathbf{\Gamma}_\delta)z_\delta\,\mathrm{d}\mathbf{x}. \qquad (24c)
\end{aligned}$$

At the discrete level the right-hand side of (24b) and (24c) are decoupled at the elemental level and so the auxiliary variables $\mathbf{w}_\delta$ and $\mathbf{v}_\delta$ can be substituted into (24a) to recover the primal form of the equation. The $\mathbf{f}_{(2,3)}(\mathbf{U}_\delta)$ term is discretely evaluated by the solution of Eq. (19). We recover the original variables by subsequently solving the discrete problem (21a): find $(H\mathbf{u})_\delta \in \mathscr{W}_\delta$ such that for all $\mathbf{t}_\delta \in \mathscr{W}_\delta$ and for all $\mathscr{T}_e \in \mathscr{T}_h$

$$\begin{aligned}
\int_{\mathscr{T}_e} \mathbf{t}_\delta\cdot\partial_t(H\mathbf{u})_\delta\,\mathrm{d}\mathbf{x} &= \int_{\mathscr{T}_e} (\mathbf{t}_\delta\cdot\gamma_\delta\nabla z_\delta)\,\mathrm{d}\mathbf{x} + \int_{\partial\mathscr{T}_e} (\mathbf{t}_\delta\gamma_\delta(\hat{z}_\delta + z_\delta))\cdot\mathbf{n}\,\mathrm{d}S + 2\int_{\mathscr{T}_e} (\mathbf{t}_\delta\cdot\mathbf{\Gamma}_\delta)z_\delta\,\mathrm{d}\mathbf{x}\\
&\quad + \int_{\mathscr{T}_e} \mathbf{t}_\delta\cdot\mathbf{f}_{(2,3)}(\mathbf{U}_\delta)\,\mathrm{d}\mathbf{x}. \qquad (25)
\end{aligned}$$

To summarise the scalar method: at every time step (or sub step) n^k do

- compute $\mathbf{f}(\mathbf{U}_\delta)$ from Eq. (19);
- compute z_δ using Eqs. (24a)–(24c);
- return to $\partial_t(H\mathbf{u})_\delta$ by Eq. (25);
- advance to the next time level n^{k+1} using the explicit time stepping scheme in Section 3.7.

As the mass matrix in general is diagonal, solving (19) and (25) are relatively cheap.

For the constant depth case the scalar method corresponds to solving the following global matrix systems

$$\begin{bmatrix} \mathbf{M} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{M} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{M}\end{bmatrix}\begin{bmatrix}\tilde{\mathbf{f}}_{(1)}\\ \tilde{\mathbf{f}}_{(2)}\\ \tilde{\mathbf{f}}_{(3)}\end{bmatrix} = \begin{bmatrix} -\mathbf{D}^{\mathbf{a}}_{\mathbf{x}}\tilde{\mathbf{E}}_{(1)} - \mathbf{D}^{\mathbf{a}}_{\mathbf{y}}\tilde{\mathbf{G}}_{(1)}\\ -\mathbf{D}^{\mathbf{a}}_{\mathbf{x}}\tilde{\mathbf{E}}_{(2)} - \mathbf{D}^{\mathbf{a}}_{\mathbf{y}}\tilde{\mathbf{G}}_{(2)}\\ -\mathbf{D}^{\mathbf{a}}_{\mathbf{x}}\tilde{\mathbf{E}}_{(3)} - \mathbf{D}^{\mathbf{a}}_{\mathbf{y}}\tilde{\mathbf{G}}_{(3)}\end{bmatrix}, \qquad (26a)$$

$$(\gamma\mathbf{D}^{\mathbf{d}}_{\mathbf{xx}} + \gamma\mathbf{D}^{\mathbf{d}}_{\mathbf{yy}} - \mathbf{M})\tilde{z} = -\mathbf{D}^{\mathbf{d}}_{\mathbf{x}}\tilde{\mathbf{f}}_{(2)} - \mathbf{D}^{\mathbf{d}}_{\mathbf{y}}\tilde{\mathbf{f}}_{(3)}, \qquad (26b)$$

$$\begin{bmatrix} \mathbf{M} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{M} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{M}\end{bmatrix}\begin{bmatrix}\partial_t\tilde{H}\\ \partial_t(\widetilde{Hu})\\ \partial_t(\widetilde{Hv})\end{bmatrix} = \begin{bmatrix}\mathbf{M}\tilde{\mathbf{f}}_{(1)}\\ \gamma\mathbf{D}^{\mathbf{d}}_{\mathbf{x}}\tilde{z} + \mathbf{M}\tilde{\mathbf{f}}_{(2)}\\ \gamma\mathbf{D}^{\mathbf{d}}_{\mathbf{y}}\tilde{z} + \mathbf{M}\tilde{\mathbf{f}}_{(3)}\end{bmatrix}. \qquad (26c)$$

3.4. Numerical fluxes

In order to enforce a suitable inter-elemental coupling the numerical fluxes need to be defined. In doing so we distinguish between the advective numerical flux and the dispersive numerical fluxes. We introduce a notation of subscripts L and R. The subscript L stands for left-hand state of the element boundary, which is assumed to be internal to the element. The subscript R denotes the right-hand state, which is internal to the adjacent element.

3.4.1. Advective numerical flux

Throughout this study we use the contact wave modified Harten–Lax–van Leer (HLLC) Riemann solver with the two-rarefaction assumption [42]. This flux was numerically shown in [14] to give optimal convergence of $P + 1$ for the SWE.

Introducing the rotation matrix and its inverse

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & n_x & n_y \\ 0 & -n_y & n_x \end{bmatrix}, \quad \mathbf{T}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & n_x & -n_y \\ 0 & n_y & n_x \end{bmatrix}, \tag{27}$$

we can define $\mathbf{Q} = \mathbf{T}\mathbf{U}_\delta = [H, Hu^\perp, Hu^\parallel]^T$, where $u^\perp$ and $u^\parallel$ are the velocities in the direction normal and tangential to the edge, respectively. The advective flux can now be written as

$$\hat{\mathbf{F}}(\mathbf{U}_\delta) \cdot \mathbf{n} = \mathbf{T}^{-1}\hat{\mathbf{E}}(\mathbf{Q}). \tag{28}$$

The HLLC flux is given by [42]

$$\hat{\mathbf{E}}(\mathbf{Q}) = \begin{cases} \mathbf{E}(\mathbf{Q}_L) & \text{if } S_L \geqslant 0, \\ \mathbf{E}(\mathbf{Q}_L) + S_L(\mathbf{Q}_{*L} - \mathbf{Q}_L) & \text{if } S_L \leqslant 0 \leqslant S_*, \\ \mathbf{E}(\mathbf{Q}_R) + S_R(\mathbf{Q}_{*R} - \mathbf{Q}_R) & \text{if } S_* \leqslant 0 \leqslant S_R, \\ \mathbf{E}(\mathbf{Q}_R) & \text{if } S_R \leqslant 0, \end{cases} \tag{29}$$

where $\mathbf{Q}_{*L}$ and $\mathbf{Q}_{*R}$ are obtained from

$$\mathbf{Q}_{*(L,R)} = H_{(L,R)} \left(\frac{S_{(L,R)} - u^\perp_{(L,R)}}{S_{(L,R)} - S_*} \right) \begin{bmatrix} 1 \\ S_* \\ u^\parallel_{(L,R)} \end{bmatrix}. \tag{30}$$

The wave speeds are estimated as [42]

$$S_L = u^\perp_L - \sqrt{gH_L}s_L, \tag{31}$$

$$S_R = u^\perp_R + \sqrt{gH_R}s_R, \tag{32}$$

$$S_* = \frac{S_L H_R(u^\perp_R - S_R) - S_R H_L(u^\perp_L - S_L)}{H_R(u^\perp_R - S_R) - H_L(u^\perp_L - S_L)}, \tag{33}$$

where

$$s_{(L,R)} = \begin{cases} \sqrt{(H_*^2 + H_* H_{(L,R)})/(2H^2_{(L,R)})} & \text{if } H_* > H_{(L,R)}, \\ 1 & \text{if } H_* \leqslant H_{(L,R)}. \end{cases} \tag{34}$$

The water depth in the star region, H_*, is approximated by the two-rarefaction Riemann solver

$$H_* = \frac{1}{g}\left(\frac{1}{2}(\sqrt{gH_L} + \sqrt{gH_R}) + \frac{1}{4}\left(u^\perp_L - u^\perp_R\right) \right)^2. \tag{35}$$

3.4.2. Dispersive numerical fluxes

In [15], the scalar approach was used and the dispersive flux was evaluated using the Bassi–Rebay (BR) flux [4]. Although exponential convergence was numerically demonstrated, for elliptic problems the BR flux is known to give sub-optimal convergence for odd P [22]. Furthermore, the BR method uses quite a wide stencil, 10 elements in the two-dimensional case.

In 1998, Cockburn and Shu [7] generalised the method of Bassi and Rebay and constructed the local discontinuous Galerkin (LDG) method. The LDG method employs a different flux and includes a penalty term. Setting the flux to be alternating, i.e., upwinding $\hat{z}$ and downwinding $\hat{\mathbf{w}}$, or vice versa – the method gives optimal convergence for both odd and even P and a stencil that is at most 6 elements wide.

In this study, we will examine the influence on different choices of dispersive flux. In addition to the two methods mentioned above we will examine a stabilized version of the BR flux (sBR), in which we have introduced a penalty term.

We introduce the notation described in [3], i.e., let $\{\cdot\}$ denote averaging across the element boundary and $[\![\cdot]\!]$ denote the jump over the element boundary. For an arbitrary scalar function f we have

$$\{f\} = \frac{1}{2}(f_{\mathrm{L}} + f_{\mathrm{R}}), \quad [\![f]\!] = (f_{\mathrm{L}} - f_{\mathrm{R}})\mathbf{n} \tag{36}$$

and for an arbitrary vector-valued function $\mathbf{f}$

$$\{\mathbf{f}\} = \frac{1}{2}(\mathbf{f}_{\mathrm{L}} + \mathbf{f}_{\mathrm{R}}), \quad [\![\mathbf{f}]\!] = (\mathbf{f}_{\mathrm{L}} - \mathbf{f}_{\mathrm{R}}) \cdot \mathbf{n}. \tag{37}$$

The numerical fluxes associated with the different DG methods are presented in Table 1. In the penalty terms $\eta \geqslant 0$ denotes the stabilisation parameter. The stability and conditioning of elliptic operators with respect to this parameter has been investigated by Castillo [6] and Sherwin et al. [36]. We observe that sBR is equal to LDG with $\boldsymbol{\beta} = \mathbf{0}$, giving a 10 element stencil. Additionally, setting $\eta = 0$ gives the BR formulation. For an upwind/downwind LDG flux we note that, as the Boussinesq equations allow multi-directional wave propagation, the implicit matrix becomes time-dependent. We, however, disregard the time dependency and construct the implicit matrix as if the flow was constant over time in a pre-defined direction $\boldsymbol{\Delta}$. Hence, the factor $\boldsymbol{\beta}$ in the LDG flux is given by: if $\boldsymbol{\Delta} \cdot \mathbf{n} > 0$, then $\boldsymbol{\beta} \cdot \mathbf{n} = 1/2$; otherwise $\boldsymbol{\beta} \cdot \mathbf{n} = -1/2$.

The additional dispersive numerical fluxes $\hat{\zeta}_\delta$ and $\hat{\mathbf{b}}_\delta$ present in the $\mathbf{D}^{\mathbf{s}}$ term (16a)–(16c) are evaluated using the BR flux. The $\hat{\mathbf{f}}_{(2,3)}(\mathbf{U})_\delta$ terms in Eq. (24a) are calculated using averaging.

3.5. Boundary conditions

Generally, boundary conditions are imposed through the numerical fluxes by setting the right-hand state to a specific value. We first note that all boundary conditions arising from the $\mathbf{D}^{\mathbf{m}}$ term are treated as Neumann conditions:

$$\hat{\mathbf{w}}_\delta = \mathbf{w}_{\mathcal{N}}, \quad \hat{\mathbf{v}}_\delta = \mathbf{v}_{\mathcal{N}}, \quad (\widehat{H\mathbf{u}})_\delta = (H\mathbf{u})_{\mathcal{N}}, \quad \hat{z}_\delta = z_{\mathrm{L}}, \quad \hat{a}_\delta = a_{\mathrm{L}} \text{ on } \partial\Omega_{\mathcal{N}}. \tag{38}$$

At slip wall boundaries we have the impermeability condition $\mathbf{u} \cdot \mathbf{n} = 0$, implemented by setting the right-hand state to

$$H_{\mathrm{R}} = H_{\mathrm{L}}, \quad \zeta_{\mathrm{R}} = \zeta_{\mathrm{L}}, \quad u_{\mathrm{R}}^{\perp} = -u_{\mathrm{L}}^{\perp}, \quad u_{\mathrm{R}}^{\parallel} = u_{\mathrm{L}}^{\parallel}, \quad b_{\mathrm{R}}^{\perp} = -b_{\mathrm{L}}^{\perp}, \quad b_{\mathrm{R}}^{\parallel} = b_{\mathrm{L}}^{\parallel}, \tag{39}$$

as well as imposing $\mathbf{w}_{\mathcal{N}} = \mathbf{v}_{\mathcal{N}} = (H\mathbf{u})_{\mathcal{N}} = \mathbf{0}$.

For inflow/outflow boundaries we can simply impose the *a priori* known values at the right-hand state and for $\mathbf{w}_{\mathcal{N}}$, $\mathbf{v}_{\mathcal{N}}$ and $(H\mathbf{u})_{\mathcal{N}}$. However, in many cases the numerical model is 'cold-started' from a motionless

Table 1
The fluxes for the dispersive part ($\hat{a}_\delta$ and $(\widehat{H\mathbf{u}})_\delta$ are analogously defined)

Method	$\hat{z}_\delta$	$\hat{\mathbf{w}}_\delta$
BR	$\{z_\delta\}$	$\{\mathbf{w}_\delta\}$
sBR	$\{z_\delta\}$	$\{\mathbf{w}_\delta\}-(\eta/h)[\![z_\delta]\!]$
LDG	$\{z_\delta\} + \boldsymbol{\beta} \cdot [\![z_\delta]\!]$	$\{\mathbf{w}_\delta\}-\boldsymbol{\beta}[\![\mathbf{w}_\delta]\!]-(\eta/h)[\![z_\delta]\!]$

initial condition. Hence, in order to minimise the numerical noise created during the cold-start we have adopted the approach of relaxation zones. This approach has been reported to work satisfactorily for highly dispersive Boussinesq equations [32]. Inside the relaxation zones the primitive variables are given as:

$$\mathbf{u}^{\mathrm{r}} = c^{\mathrm{r}}\mathbf{u}_\delta + (1 - c^{\mathrm{r}})\mathbf{u}^{\mathrm{p}}, \tag{40a}$$

$$\zeta^{\mathrm{r}} = c^{\mathrm{r}}\zeta_\delta + (1 - c^{\mathrm{r}})\zeta^{\mathrm{p}}, \tag{40b}$$

where $0 \leqslant c^{\mathrm{r}}(\mathbf{x}) \leqslant 1$ is the relaxation coefficient, $\mathbf{u}^{\mathrm{p}}(\mathbf{x}, t)$ and $\zeta^{\mathrm{p}}(\mathbf{x}, t)$ are the prescribed values, while $\mathbf{u}^{\mathrm{r}}(\mathbf{x}, t)$ and $\zeta^{\mathrm{r}}(\mathbf{x}, t)$ are the redefined values inside the relaxation zone. Open boundaries are given by (40a) and (40b) setting the prescribed values to be zero.

3.6. Expansion basis

As mentioned above, the variables are approximated using a polynomial expansion basis $\phi_{pq}(\xi_1, \xi_2)$, such that

$$f_\delta(\mathbf{x}, t) = \sum_{p=0}^{P} \sum_{q=0}^{P-p} \tilde{f}_{pq}(t)\phi_{pq}(\xi_1, \xi_2), \quad \mathbf{x} \in \mathscr{T}_e, \tag{41}$$

where $\tilde{f}_{pq}(t)$ contains the local degrees of freedom expansion coefficients.

The orthogonal hierarchical basis $\phi_{pq}(\xi_1, \xi_2)$ in a standard triangular region $\{-1 \leqslant \xi_1, \xi_2; \xi_1 + \xi_2 \leqslant 0\}$ is based on a collapsed coordinate [22] which is generated through the transformation $(\xi_1, \xi_2) \to (\eta_1, \eta_2)$ given by:

$$\eta_1 = 2\frac{(1 + \xi_1)}{(1 - \xi_2)} - 1, \quad \eta_2 = \xi_2. \tag{42}$$

This collapsed coordinate transformation can be interpreted as a mapping to a standard quadrilateral region from the standard triangular region.

An orthogonal basis on these coordinates has been independently derived in a range of works including [35,23,11]. Following the formulation in [11,22], the expansion modes ϕ_{pq} are defined in terms of principal functions $\tilde{\psi}^a_p(z)$ and $\tilde{\psi}^b_{pq}(z)$ as

$$\phi_{pq}(\xi_1, \xi_2) = \tilde{\psi}^a_p(\eta_1)\tilde{\psi}^b_{pq}(\eta_2). \tag{43}$$

The principal functions are

$$\tilde{\psi}^a_p(z) = P^{0,0}_p(z), \quad \tilde{\psi}^b_{pq}(z) = \left(\frac{1 - z}{2}\right)^p P^{2p+1,0}_q(z), \tag{44}$$

where $P^{\alpha,\beta}_p(z)$ denotes the pth order Jacobi polynomial.

3.7. Time stepping and eigenspectra

In spectral/*hp* formulations advective terms are usually handled explicitly in time, while diffusive terms typically are treated implicitly. This is due to the rapid growth of the spectral radius, $\mathcal{O}(P^4)$, of the weak Laplacian operator [22]. This implies that the $\mathbf{D}^{\mathrm{s}}$ term, which contains second-order spatial derivatives, ought to be treated implicitly in time. However, the grouping of the third-order mixed derivatives with the first-order time derivatives causes the spectral radius to grow as $\mathcal{O}(P^2)$, as will be shown below. The restriction on the explicit time step will therefore be of the same order as if only advective terms were present. Thus we can use a standard explicit time-stepping scheme, in this work we have adopted the explicit third-order TVD Runge–Kutta scheme [9].

We write the Boussinesq equations in quasi-linear form

$$\partial_t(\mathbf{U}+\mathbf{D^m}(\mathbf{U}))+\mathbf{D^s}(\mathbf{U})+\mathbf{A}(\mathbf{U})\partial_x\mathbf{U}+\mathbf{B}(\mathbf{U})\partial_y\mathbf{U}=\mathbf{S}(\mathbf{U}). \tag{45}$$

Here $\mathbf{A}(\mathbf{U})$ and $\mathbf{B}(\mathbf{U})$ are the Jacobians of the flux functions

$$\mathbf{A}(\mathbf{U})=\begin{bmatrix}0 & 1 & 0\\ c^2-u^2 & 2u & 0\\ -uv & v & u\end{bmatrix},\quad \mathbf{B}(\mathbf{U})=\begin{bmatrix}0 & 0 & 1\\ -uv & v & u\\ c^2-v^2 & 0 & 2v\end{bmatrix}, \tag{46}$$

in which $c=\sqrt{gH}$ is the long-wave speed. Writing Eq. (45) as $\mathbf{X}\partial_t\tilde{\mathbf{U}}_\delta=\mathbf{Y}\tilde{\mathbf{U}}_\delta$ we are interested in the behaviour of the eigenvalues, λ, of the semi-discrete operator $\mathbf{X}^{-1}\mathbf{Y}$ (in order to simplify the procedure we evaluate the advective flux in this section by component-wise averaging).

We will consider the case of a sinusoidal wave in a periodic domain of size $[-1\leqslant x,y\leqslant 1]$. The wave has a wavelength of 20 m and the water depth is 5 m, giving a d/L_0 ratio of 0.22. The amplitude is set to 0.1 m and the bed slope in the wave direction is 1:30 (as the slope is mild we further simplify and treat d and ∇d as constants). The domain is discretized into two triangles, where one triangle is in the standard space, and we will examine wave directions at different θ angles to the horizontal (see Fig. 1).

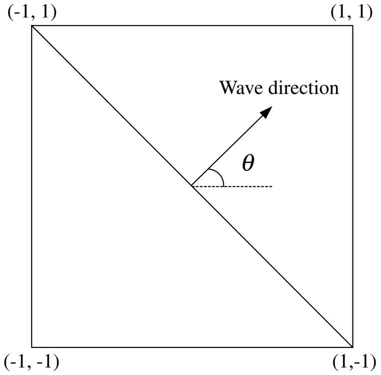

Fig. 1. The wave propagates at the angle θ to the horizontal in the periodic domain.

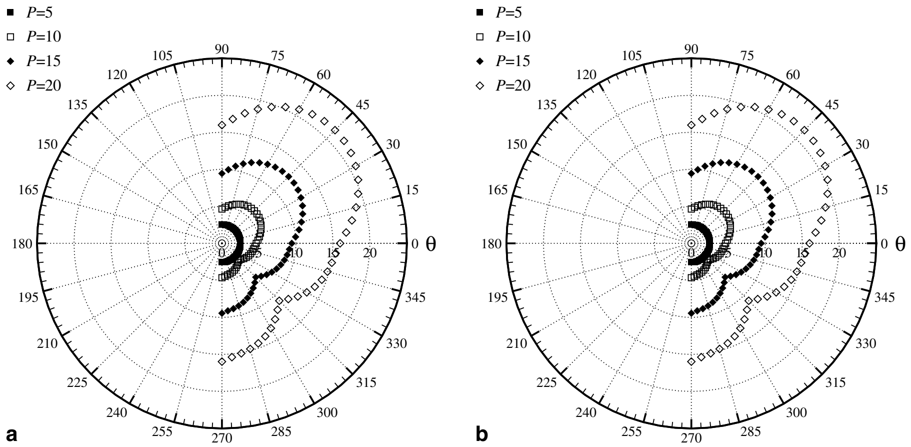

Fig. 2. The maximum eigenvalue of $\mathbf{X}^{-1}\mathbf{Y}$ using the BR flux: (a) coupled method and (b) scalar method.

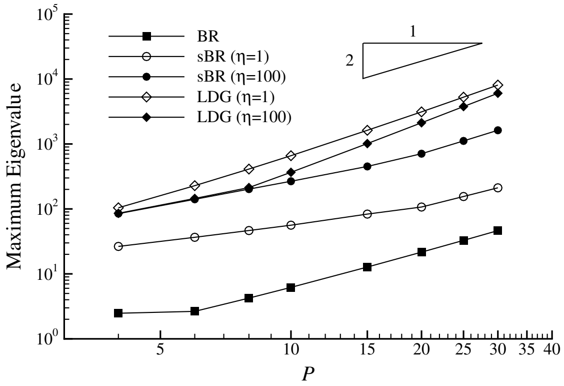

Fig. 3. Growth of the maximum eigenvalue with respect to polynomial order P.

Fig. 2 shows the maximum spectral radius, using the BR flux, for the scalar and coupled methods. As evident from Fig. 2, the maximum eigenvalues are identical, illustrating the equivalence of the two solution approaches. The maximum eigenvalues occurs at $\theta = 45°$ which corresponds to the shortest distance across the triangular elements.

In Fig. 3, we illustrate the growth of the maximum eigenvalue for the different dispersive flux formulations, using the scalar method ($\theta = 45°$). Regardless of flux formulation the growth rate is of $\mathcal{O}(P^2)$. For the sBR flux the eigenvalues are highly dependent on η, a large η gives a larger maximum eigenvalue. For the LDG flux the dependence is less significant. As a larger eigenvalue implies a harsher restriction on the explicit time step the BR flux permits the largest time step of the three fluxes considered.

4. Accuracy and efficiency

In this section, we numerically examine the influence of the choice of: (i) solution approach; and (ii) dispersive fluxes on the convergence rate and CPU time.

Consider the simple case of a linear standing wave in a frictionless rectangular basin of constant depth. The analytic solution can be written as:

$$H(\mathbf{x},t) = d + a\cos(kx)\cos(\omega t), \tag{47a}$$

$$u(\mathbf{x},t) = a\frac{\omega}{kd}\sin(kx)\sin(\omega t), \tag{47b}$$

$$v(\mathbf{x},t) = 0, \tag{47c}$$

where a is the amplitude and k is the wave number. The frequency ω is obtained from the linear dispersion equation

$$\frac{\omega^2}{gdk^2} = \frac{1}{1+(1/3)(kd)^2}. \tag{48}$$

The dimension of the basin is $L \times L/2$, where $L = 100$ m is the wavelength. The still water depth of the basin is set to $d = 25$ m, giving $d/L_0 \approx 0.22$. We compute one wave period, using 10 000 time steps, for a standing wave with an amplitude of 0.1 m using the linearised Boussinesq equations. Three structured meshes having 16, 64 and 256 evenly distributed elements are used.

The resulting matrix systems are solved using the sparse matrix solver UMFPACK [10] and we measure the accuracy in the L^2 and L^∞ norms. In comparing the computational efficiency and storage requirement we will use the scalar approach with BR flux as reference. We therefore introduce the ratios

$$r(N_{\mathrm{nz}}) = N_{\mathrm{nz}}/N_{\mathrm{nz}}^{\mathrm{ref}}, \quad r(\mathrm{CPU}) = \mathrm{CPU}/\mathrm{CPU}^{\mathrm{ref}}, \tag{49}$$

where N_{nz} denotes the number of non-zero entries in the implicit matrix and CPU is simply the computational time measured in seconds.

4.1. Coupled versus scalar method

We start by comparing the coupled and scalar methods using the BR fluxes. The L^2 and L^∞ errors and order of convergence are presented in Table 2. We see no difference between the two approaches in terms of accuracy and convergence. Indeed, the results are identical, as could be expected from the eigenspectra analysis.

From the difference in size of the implicit matrices we would expect the storage requirement of the coupled method to be at most four times the scalar method. However, from Table 3 we see that for the present case the coupled method requires three times the storage of the scalar method. This is simply due to the use of structured meshes aligned along the Cartesian axes – otherwise a ratio of four is obtained.

From Table 3, we see that for linear expansions the coupled method is the computationally most efficient approach, but for $P \geqslant 2$ the scalar method requires less CPU time. Importantly, the computational efficiency of the scalar method is seen to increase with increasing expansion order.

4.2. Influence of dispersive flux

In this section, we use the scalar method. For the sBR and LDG fluxes we have taken a stabilisation factor of $\eta = 100$. The error and order of convergence are presented in Table 4. The choice of dispersive

Table 2
Error and order of convergence for the H-component using the scalar and coupled methods with BR flux

Norm	Method	P	$N = 16$	$N = 64$		$N = 256$	
			Error	Error	Order	Error	Order
L^2	Scalar/BR	1	1.1226E − 02	2.6768E − 03	2.07	5.8530E − 04	2.19
		2	1.3230E − 03	1.9539E − 04	2.76	2.5891E − 05	2.92
		3	1.0811E − 04	6.6072E − 06	4.03	8.3836E − 07	2.98
		4	1.0052E − 05	3.3676E − 07	4.90	1.0690E − 08	4.98
	Coupled/BR	1	1.1226E − 02	2.6768E − 03	2.07	5.8530E − 04	2.19
		2	1.3230E − 03	1.9539E − 04	2.76	2.5891E − 05	2.92
		3	1.0811E − 04	6.6072E − 06	4.03	8.3836E − 07	2.98
		4	1.0052E − 05	3.3676E − 07	4.90	1.0690E − 08	4.98
L^∞	Scalar/BR	1	1.7617E − 02	5.6046E − 03	1.65	1.2268E − 03	2.19
		2	3.0141E − 03	3.9179E − 04	2.94	5.5468E − 05	2.82
		3	2.8636E − 04	2.5172E − 05	3.51	2.6100E − 06	3.27
		4	3.3832E − 05	8.7677E − 07	5.27	2.9137E − 08	4.91
	Coupled/BR	1	1.7617E − 02	5.6046E − 03	1.65	1.2268E − 03	2.19
		2	3.0141E − 03	3.9179E − 04	2.94	5.5468E − 05	2.82
		3	2.8636E − 04	2.5172E − 05	3.51	2.6100E − 06	3.27
		4	3.3832E − 05	8.7677E − 07	5.27	2.9137E − 08	4.91

Table 3
Ratios of N_{nz} and CPU time between the coupled and scalar methods using the BR flux

P	$N = 16$		$N = 64$		$N = 256$	
	$r(N_{\mathrm{nz}})$	$r(\mathrm{CPU})$	$r(N_{\mathrm{nz}})$	$r(\mathrm{CPU})$	$r(N_{\mathrm{nz}})$	$r(\mathrm{CPU})$
1	3.01	0.71	2.88	0.91	3.00	0.91
2	2.99	1.38	3.00	1.53	3.00	1.56
3	3.00	1.60	2.99	2.00	3.00	1.99
4	3.00	2.54	2.99	2.56	2.99	2.30

Table 4
Error and order of convergence for the H-component using the scalar method with different dispersive fluxes

Norm	Method	P	$N = 16$	$N = 64$		$N = 256$	
			Error	Error	Order	Error	Order
L^2	Scalar/BR	1	1.1226E − 02	2.6768E − 03	2.07	5.8530E − 04	2.19
		2	1.3230E − 03	1.9539E − 04	2.76	2.5891E − 05	2.92
		3	1.0811E − 04	6.6072E − 06	4.03	8.3836E − 07	2.98
		4	1.0052E − 05	3.3676E − 07	4.90	1.0690E − 08	4.98
	Scalar/sBR	1	1.0116E − 02	2.1906E − 03	2.21	4.9405E − 04	2.15
		2	9.8322E − 04	1.3377E − 04	2.88	1.7206E − 05	2.96
		3	9.3879E − 05	5.7200E − 06	4.04	3.5945E − 07	3.99
		4	7.0653E − 06	2.3228E − 07	4.93	7.5176E − 09	4.95
	Scalar/LDG	1	1.0128E − 02	2.1932E − 03	2.21	4.9361E − 04	2.15
		2	9.8134E − 04	1.3328E − 04	2.88	1.7180E − 05	2.96
		3	9.4113E − 05	5.7090E − 06	4.04	3.5769E − 07	4.00
		4	7.0674E − 06	2.3552E − 07	4.91	7.6665E − 09	4.94
L^∞	Scalar/BR	1	1.7617E − 02	5.6046E − 03	1.65	1.2268E − 03	2.19
		2	3.0141E − 03	3.9179E − 04	2.94	5.5468E − 05	2.82
		3	2.8636E − 04	2.5172E − 05	3.51	2.6100E − 06	3.27
		4	3.3832E − 05	8.7677E − 07	5.27	2.9137E − 08	4.91
	Scalar/sBR	1	1.6093E − 02	5.8062E − 03	1.47	1.5763E − 03	1.88
		2	4.1474E − 03	5.7118E − 04	2.86	5.9436E − 05	3.26
		3	4.0362E − 04	3.7917E − 05	3.41	2.3903E − 06	3.99
		4	4.8363E − 05	1.7865E − 06	4.76	4.7637E − 08	5.23
	Scalar/LDG	1	1.6219E − 02	5.8982E − 03	1.46	1.5866E − 03	1.89
		2	4.1902E − 03	5.5147E − 04	2.93	5.8349E − 05	3.24
		3	4.3846E − 04	4.0372E − 05	3.44	2.4660E − 06	4.03
		4	4.8400E − 05	1.6822E − 06	4.85	4.6965E − 08	5.16

DG formulation clearly makes a difference. The sBR and LDG fluxes give optimal convergence of order $P + 1$ for both odd and even P, as expected since the penalty terms are of order h^{-1} [5]. As seen in other work [7] the BR flux, which lacks a penalty term, gives optimal convergence of order $P + 1$ for even P, but for odd P it can degenerate to order P.

As the stencil of the LDG flux is at most 6 elements, compared to 10 elements of the BR and sBR fluxes, a theoretical upper bound of the storage ratio is 0.6. In Table 5, we present the storage and CPU ratios. We see that the sBR does not require any additional storage and that the storage ratio of the LDG flux is close to the theoretical bound. The benefit of the small N_{nz} of the LDG flux also carries over into the computational time which is smaller per time step than for the BR and sBR fluxes.

Table 5
Ratios of $N_{\rm nz}$ and CPU time relative the scalar method using the BR flux

Method	P	$N = 16$		$N = 64$		$N = 256$	
		$r(N_{\rm nz})$	r(CPU)	$r(N_{\rm nz})$	r(CPU)	$r(N_{\rm nz})$	r(CPU)
Scalar/sBR	1	1.04	1.06	1.00	1.03	1.00	1.02
	2	1.00	1.00	1.00	1.02	1.00	1.07
	3	1.00	1.01	1.00	1.01	1.00	1.09
	4	1.00	1.00	1.00	0.95	1.00	1.05
Scalar/LDG	1	0.68	1.01	0.63	0.97	0.61	0.95
	2	0.62	0.86	0.63	0.74	0.62	0.85
	3	0.62	0.83	0.63	0.86	0.62	0.77
	4	0.61	0.77	0.63	0.84	0.62	0.69

4.3. Influence of the stabilisation parameter

We examine the results obtained from the scalar method with sBR and LDG fluxes using a parameter in the interval $10^{-3} \leqslant \eta \leqslant 10^{3}$. Fig. 4 shows the ratios of the errors between the fluxes using penalty term and the BR flux (no stabilisation), as a function of the stabilisation parameter.

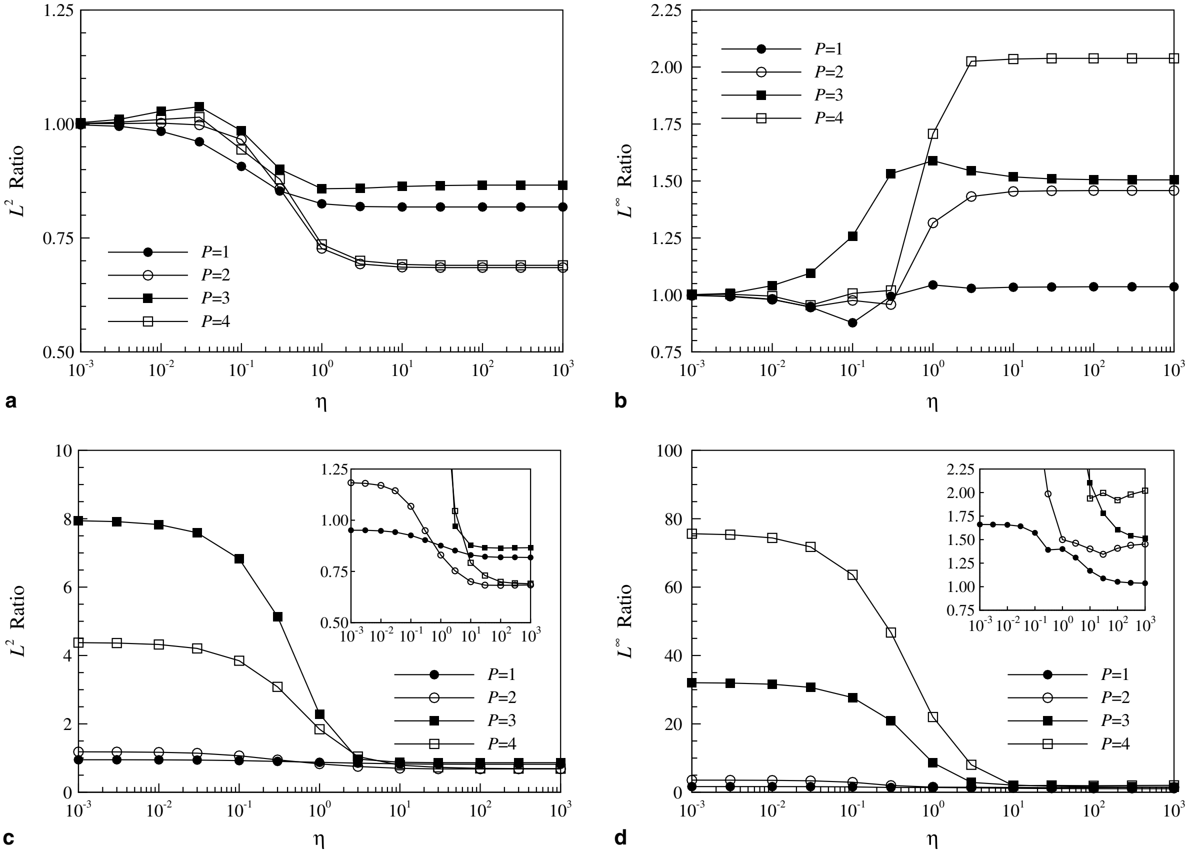

Fig. 4. Ratios of the L^2 and L^∞ errors for the H-component ($N = 64$): (a,b) sBR and (c,d) LDG.

We see that the effect of the penalty term is very different for the sBR and LDG fluxes, although the results become more similar as η increases. For $\eta > 10^2$ the results are fairly equivalent for the two fluxes. For large η we also see that the penalty term is beneficial in the L^2 norm but not in the L^∞ norm.

Not surprisingly, for small η the sBR flux approaches the results of the BR flux. However, the LDG flux gives relatively bad results for $\eta < 10^1$. This is due to the "constant-in-time" approximation of the implicit matrix.

4.4. Comparison with finite differences

In this section, the DG models are compared against two finite difference (FD) models – based on the coupled and scalar methods, respectively, for the linearized Boussinesq equations. We adopt the for enhanced Boussinesq-type equations popular FD scheme proposed by Wei and Kirby [45]. In order to avoid truncation terms in the form of third-order derivatives, the advective terms are approximated to fourth order in space by a centred five-point stencil, while higher-order spatial derivatives are resolved using centred differences of second-order accuracy. We refer to [45] for a full description of the finite differences employed. As for the DG models the semi-discrete equations are advanced in time using the third-order Runge–Kutta scheme and the resulting sparse matrix system is solved using UMFPACK.

For the FD models the computational domain is uniformly discretized with a grid size h and in Table 6 we present the L^∞ error and order of convergence of the models. We see that the models are quite accurate, with the coupled method giving results being generally an order of magnitude better. Although the schemes are formally second-order accurate in space, the smoothness and weak dispersion of the standing wave case gives that the leading truncation term generally stems from the fourth-order differences – explaining the convergence of order 4.

For the finite difference models the $r(N_{\rm nz})$ ratio approaches 2.8 and the r(CPU) ratio is roughly 5 (using the finite difference model based on the scalar approach as reference), indicating that the scalar method is potentially beneficial also for FD Boussinesq models.

We compare the scalar DG (using the BR-flux) and FD model in terms of efficiency. In Fig. 5 we plot CPU times as a function of accuracy for three different integration times: 1, 10 and 100 wave periods. Here, we emphasis that the time steps have been chosen to be the maximum value not influencing the total error, i.e., temporal errors are an order of magnitude less than spatial discretization errors.

Comparing the FD model and the DG model using $P = 3$, the FD model is substantially more efficient for short integration times. However, as the integration time increases the DG method becomes the most efficient method. This is caused by the favourable dispersion properties of the finite element method [20,37]. Additionally, Fig. 5 demonstrates that there are, generally speaking, gains in CPU time using higher-order polynomials compared to lower-order and that p-type refinement is more beneficial than h-type refinement, see also [24].

The results presented in Fig. 5 are in no way to be read as fully conclusive, changing time stepping scheme, using staggered grids, etc., will naturally influence the results. Nevertheless, it indicates that the high-order DG method is competitive for long-time integration and highly accurate results, even in simple

Table 6
L^∞ error and order of convergence for the H-component for the finite difference schemes

Method	$h = 5$ m	$h = 2.5$ m		$h = 1.25$ m	
	Error	Error	Order	Error	Order
Coupled	4.6414E − 06	3.8855E − 07	3.58	2.5985E − 08	3.90
Scalar	4.9790E − 05	3.3161E − 06	3.91	2.1049E − 07	3.98

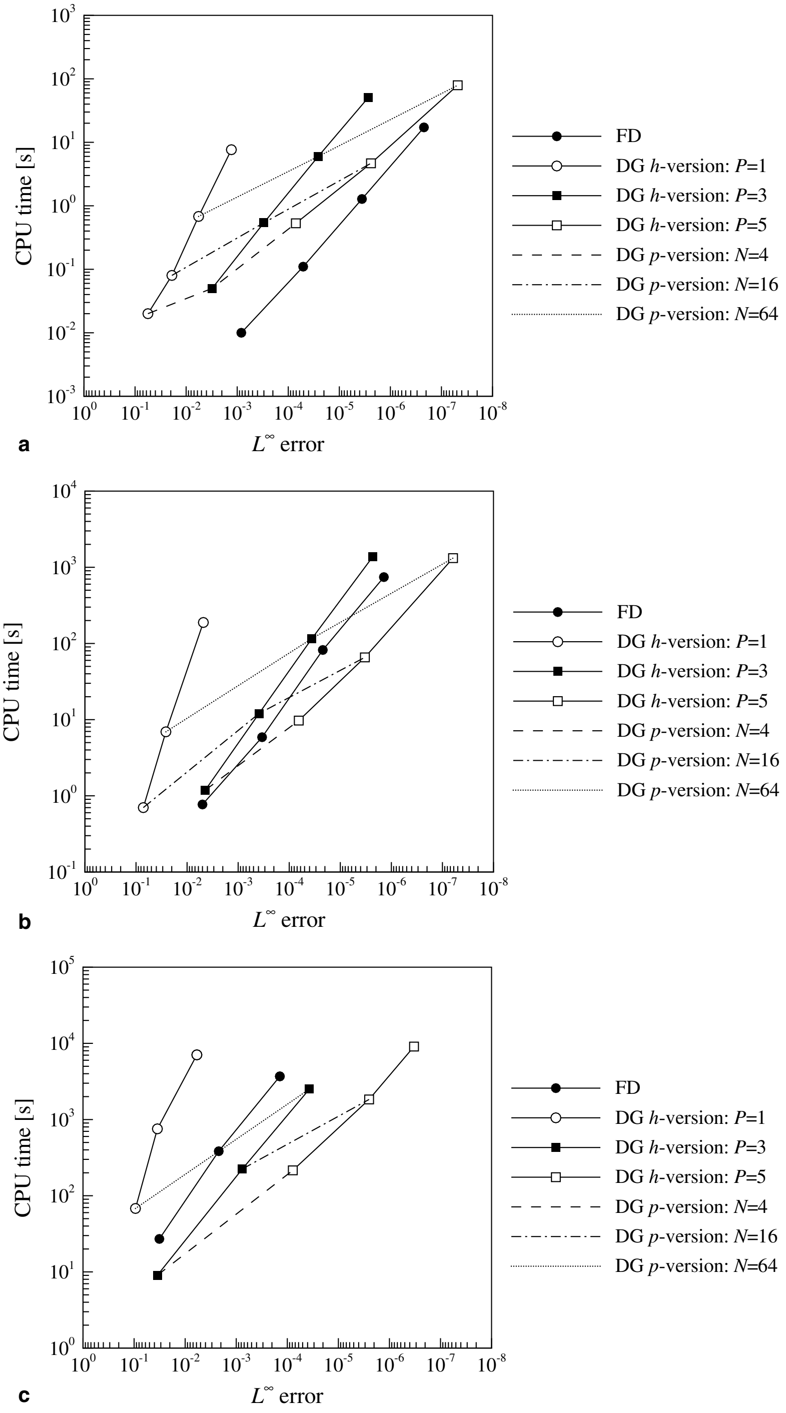

Fig. 5. CPU time versus L^∞ error for the H-component: (a) after one wave period; (b) after 10 wave periods; and (c) after 100 wave periods.

geometries. For complex geometries the flexibility of unstructured meshes will further benefit the DG method.

5. Computational examples

5.1. Propagation of a solitary wave

Consider a solitary wave of amplitude 0.1 m propagating in a channel with an undisturbed water depth of 1.0 m. The computational domain is 100 × 50 m and all boundaries are treated as walls. The solitary wave is initially located at $x = 20$ m and the shape is given by the sech-profile solution [45]. The domain is divided into 64 unstructured elements, see Fig. 6. The solution was approximated using a $P = 8$ order polynomial expansion and integrated for 20 s using 1000 time steps.

Fig. 7 shows the computed water depths in a slice through the centreline, $y = 25$ m, compared to the approximate analytical solutions. As can be seen in these plots there is a general good agreement (the small amplitude trailing waves are not caused by under resolution, but is due to the non-exact initial condition, as discussed in [45]).

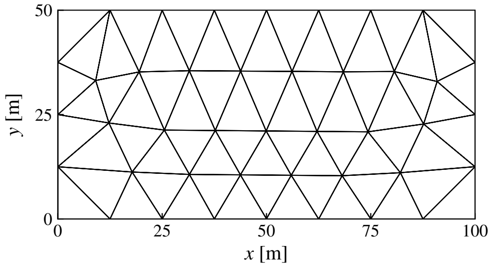

Fig. 6. Computational mesh for the solitary wave case.

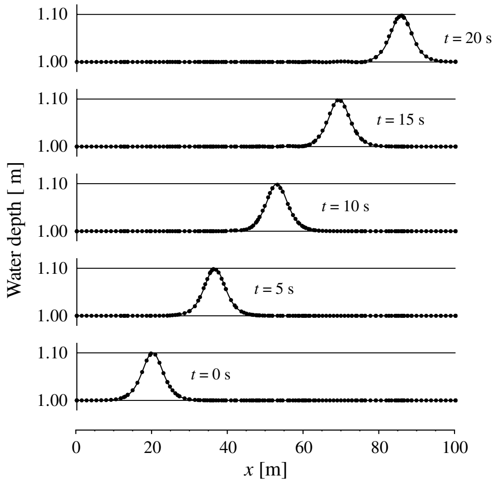

Fig. 7. Analytical (solid line) and computed (dots) solitary wave propagation along the centreline.

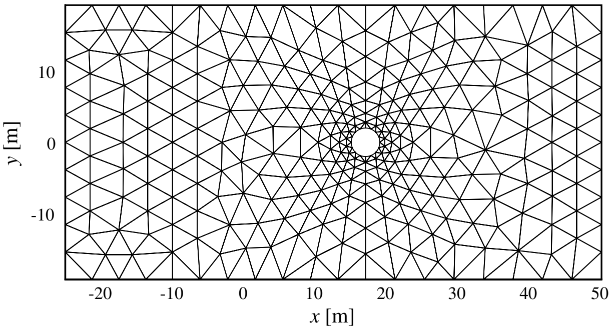

Fig. 8. Computational mesh for the scattering of a solitary wave case.

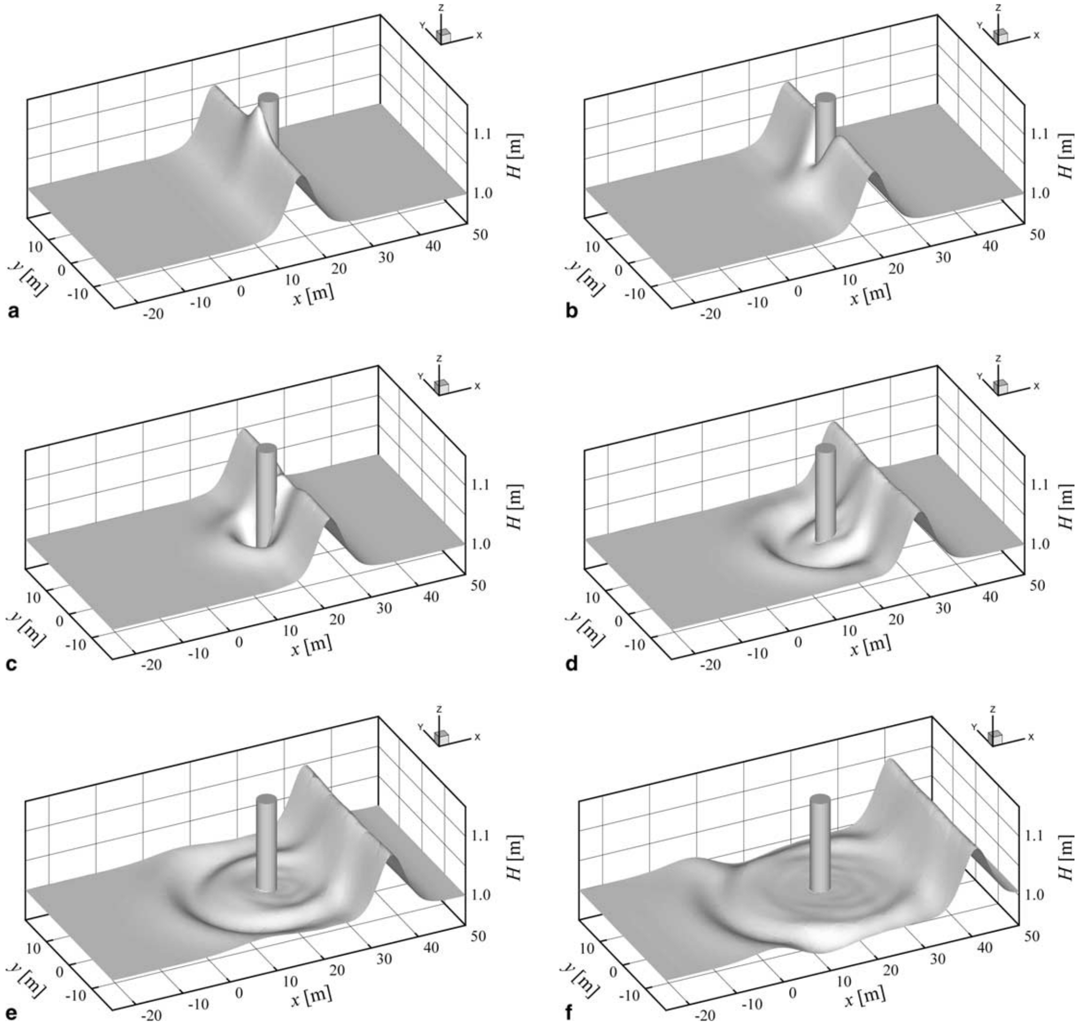

Fig. 9. Solitary wave on a cylinder: (a) $t = 4.5$ s; (b) $t = 5.5$ s; (c) $t = 6.5$ s; (d) $t = 8.5$ s; (e) $t = 10.5$ s; and (f) $t = 12.5$ s.

5.2. Scattering of a solitary wave by a vertical cylinder

The scattering of solitary waves by a vertical cylinder has been computed using Boussinesq models in [44,2,1,47]. We have a rectangular domain $-25 \leqslant x \leqslant 50$ and $-19.2 \leqslant y \leqslant 19.2$ m. A cylinder with a diameter of 4 m is located at $x = 17$ and $y = 0$ m. The domain is discretized into 552 triangles with $P = 5$, see Fig. 8. Note that the edges aligned on the cylinder boundary are curved, see e.g. [22] for a description on the implementation of curved boundaries. The undisturbed water depth is $d = 1.0$ m and the solitary wave (with 0.1 m amplitude) is initially located at $x = 0$ m. The initial solitary wave profile is approximated as in the previous case. The simulation is run for 12.5 s using 2500 time steps.

Fig. 9 shows snapshots of the water depth. At around 4 s the wave starts to run-up on the cylinder and at $t = 6.5$ s the backscattering is evident. Later we see the diffraction and reflection of the scattered waves. We also note that the solitary wave recovers its pre-impact shape.

In Fig. 10, we demonstrate the gain in accuracy by using curved boundaries. Approximating the cylinder with straight sided boundaries generates small scale numerical oscillations. Approximating the cylinder by curved boundaries the solution is smooth.

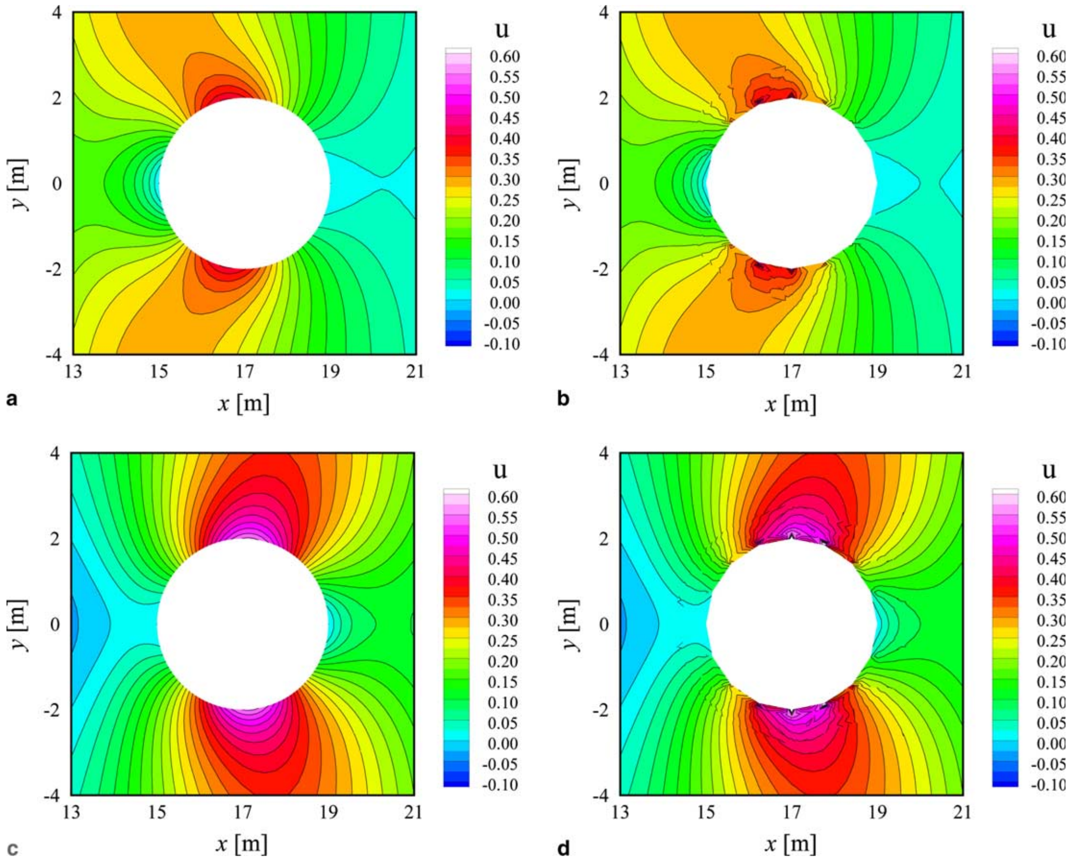

Fig. 10. Contour plots of the velocity u around the cylinder at $t = 4.5$ s (a,b) and $t = 5.5$ s (c,d). Curved edges: (a) and (c). Straight edges: (b) and (d).

5.3. Regular waves over a semicircular shoal

For this case we can compare against experimental data [46]. We have a rectangular domain of size 30×6.096 m with wall boundaries at $y = 0$ and 6.096 m. At $x = 0$ m there is an inflow boundary while at $x = 30$ m we have an open boundary. At the inflow and open boundaries we have applied 3 m wide relaxation zones. The depth in the domain is given by:

$$\Lambda(y) = \sqrt{6.096y - y^2}, \tag{50a}$$

$$d(\mathbf{x}) = \begin{cases} 0.4572, & x \leqslant 10.67 - \Lambda, \\ 0.4572 + 0.04(10.67 - \Lambda - x), & 10.67 - \Lambda < x < 18.29 - \Lambda, \\ 0.1524, & x \geqslant 18.29 - \Lambda. \end{cases} \tag{50b}$$

The incoming linear waves are given inside the relaxation zone as

$$\zeta^{\mathrm{p}}(\mathbf{x}, t) = a\sin(kx - \omega t), \tag{51a}$$

$$u^{\mathrm{p}}(\mathbf{x}, t) = a\frac{\omega}{kd}\sin(kx - \omega t), \tag{51b}$$

$$v^{\mathrm{p}}(\mathbf{x}, t) = 0. \tag{51c}$$

Here the incoming waves have an amplitude of 0.0075 m and a wave period of 2 s.

The domain is decomposed into 386 elements of order $P = 6$. The simulation is run for 50 s using 5000 time steps. In Fig. 11, we show the surface elevation after 50 s. In addition to the obvious shoaling, the semicircular shoal focus the waves to the centre. In Fig. 12, we compare the amplitudes of the first three harmonics obtained from the model with experimental data. The harmonics from the model were computed from time series sampled at vertices aligned on the centreline, using the last five wave periods of the simulation. The harmonics compare fairly well with the experimental data. The computed solutions are similar to results reported in the literature [30,40].

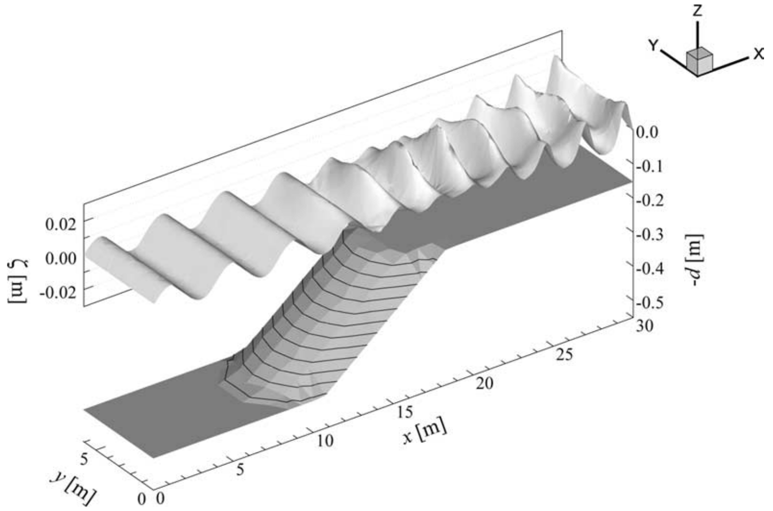

Fig. 11. Snapshot of surface elevation after 50 s shown over the bottom topography (compared to the x and y scales are the surface elevation exaggerated 100 times and the depth 20 times).

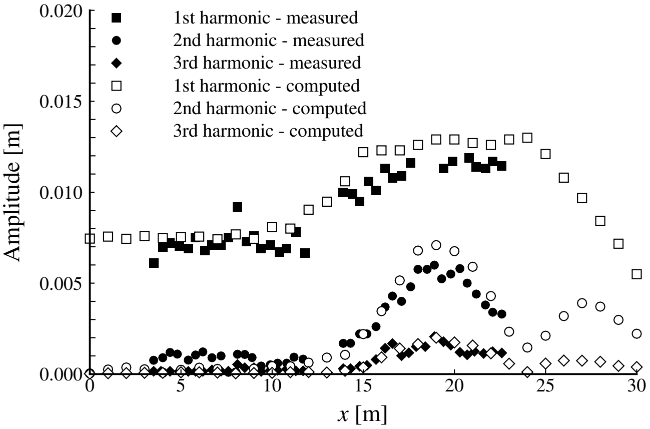

Fig. 12. Wave amplitude for first, second and third harmonic along the centreline.

6. Conclusions

We have presented triangular spectral/*hp* discontinuous Galerkin methods for modelling the propagation and evolution of weakly nonlinear and dispersive water waves over variable bottom topography. We used standard Boussinesq equations, expressed in conservative variables, which are valid for a water depth to deep water wavelength ratio (d/L_0) less than 0.22.

We investigated two different solution approaches:

- A coupled method in which the coupled momentum equations were directly discretized. This results in an implicit matrix of size $2N_{\text{dof}} \times 2N_{\text{dof}}$.
- A scalar method in which the momentum equations are rewritten into a scalar wave continuity equation of advection–diffusion type. This is achieved by introducing the time rate of change of momentum divergence, $z = \nabla \cdot \partial_t(H\mathbf{u})$, as the dependent variable. The resulting implicit matrix becomes $N_{\text{dof}} \times N_{\text{dof}}$. The original variables are recovered in a subsequent step.

We observed that the two approaches gave identical results in terms of accuracy, convergence and restriction on the time step for the DG models. However, the scalar method was more CPU efficient and required less memory to store the implicit matrix. As the polynomial order is increased in the spectral/*hp* discretization, the portion of the total CPU time used for the sparse solve increases. Subsequently, the efficiency of the DG scalar method increases with increasing order as the overhead of the recovery step and numerical flux evaluations becomes less significant.

Three formulations of the dispersive flux were considered and these all behaved as expected. The sBR and LDG fluxes were shown to give optimal convergence, $P + 1$, for both odd and even orders. The BR flux, which lacks a penalty term, sometimes degenerated to sub-optimal convergence for odd P. As the LDG has a smaller stencil, compared to stencils of the BR and sBR fluxes, the LDG requires less storage and less CPU time per time step than the other two fluxes. However, the BR and sBR fluxes are conceptually easier and allow for a larger time step.

Finally, we compared the spectral/*hp* DG model against a finite difference model. It was found that the finite difference model was superior for low-accuracy and short integration times. For long-time integration and for highly accurate results the high-order DG method was the most efficient technique, even for a simple geometry. We also note that the spectral/*hp* element methods permits very general discretization of complex and curved geometries.

References

[1] D. Ambrosi, L. Quartapelle, A Taylor-Galerkin method for simulating nonlinear dispersive water waves, Journal of Computational Physics 146 (1998) 546–569.

[2] J.S. Antunes Do Carmo, F.J. Seabra Santos, E. Barthélemy, Surface waves propagation in shallow water: a finite element model, International Journal for Numerical Methods in Fluids 16 (1993) 447–459.

[3] D.N. Arnold, F. Brezzi, B. Cockburn, L.D. Marini, Unified analysis of discontinuous Galerkin methods for elliptic problems, SIAM Journal of Numerical Analysis 39 (2002) 1749–1779.

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

[5] P. Castillo, B. Cockburn, I. Perugia, D. Schötzau, An a priori error analysis of the local discontinuous Galerkin method for elliptic problems, SIAM Journal on Numerical Analysis 38 (5) (2000) 1676–1706.

[6] P. Castillo, Performance of discontinuous Galerkin methods for elliptic problems, SIAM Journal on Numerical Analysis 24 (2) (2002) 524–547.

[7] B. Cockburn, C.-W. Shu, The local discontinuous Galerkin method for time-dependent convection-diffusion systems, SIAM Journal on Numerical Analysis 35 (6) (1998) 2440–2463.

[8] B. Cockburn, G.E. Karniadakis, C.-W. Shu, The development of discontinuous Galerkin methods, in: B. Cockburn, G.E. Karniadakis, C.-W. Shu (Eds.), Discontinuous Galerkin Methods, Lecture Notes in Computational Science and Engineering, vol. 11, Springer, Berlin, 2000, pp. 3–50.

[9] B. Cockburn, C.-W. Shu, Runge–Kutta discontinuous Galerkin methods for convection-dominated problems, Journal of Scientific Computing 16 (3) (2001) 173–261.

[10] T.A. Davis, UMFPACK Version 4.3 User Guide, Revision of Technical report TR-03-008, University of Florida, 2004.

[11] M. Dubiner, Spectral methods on triangles and other domains, Journal of Scientific Computing 6 (4) (1991) 345–390.

[12] F. Dupont, Comparison of numerical methods for modelling ocean circulation in basins with irregular coasts, PhD thesis, McGill University, 2001.

[13] C. Eskilsson, S.J. Sherwin, A discontinuous spectral element model for Boussinesq-type equations, Journal of Scientific Computing 17 (1–4) (2002) 143–152.

[14] C. Eskilsson, S.J. Sherwin, A triangular spectral/*hp* discontinuous Galerkin method for modelling 2D shallow water equations, International Journal for Numerical Methods in Fluids 45 (2004) 605–623.

[15] C. Eskilsson, S.J. Sherwin, Discontinuous Galerkin spectral/*hp* element modelling of dispersive shallow water systems, Journal of Scientific Computing 22–23 (2005) 269–288.

[16] S. Fagherazzi, P. Rasetarinera, M.Y. Hussaini, D.J. Furbish, Numerical solution of the dam-break problem with a discontinuous Galerkin method, Journal of Hydraulic Engineering 130 (6) (2004) 532–539.

[17] F.X. Giraldo, A spectral element shallow water model on spherical geodesic grids, International Journal for Numerical Methods in Fluids 35 (2001) 869–901.

[18] F.X. Giraldo, J.S. Hesthaven, T. Warburton, Nodal high-order discontinuous Galerkin methods for the spherical shallow water equations, Journal of Computational Physics 181 (2002) 499–525.

[19] M.F. Gobbi, J.T. Kirby, G. Wei, A fully nonlinear Boussinesq model for surface waves. Part 2. Extension to $\mathcal{O}(kh)^4$, Journal of Fluid Mechanics 405 (2000) 181–210.

[20] P.M. Gresho, R.L. Sani, Incompressible Flow and the Finite Element Method, Wiley, New York, 1998.

[21] M. Iskandarani, D.B. Haidvogel, J.P. Boyd, A staggered spectral element model with application to the oceanic shallow water equations, International Journal for Numerical Methods in Fluids 20 (1995) 393–414.

[22] G.Em. Karniadakis, S.J. Sherwin, Spectral/*hp* Element Methods for Computational Fluid Dynamics, second ed., Oxford University Press, Oxford, USA, 2005.

[23] T. Koornwinder, Two-variable analogues of the classical orthogonal polynomials, in: R.A. Askey (Ed.), Theory and Application of Special Functions, Academic Press, New York, 1975, pp. 435–495.

[24] H.O. Kreiss, J. Oliger, Methods for the Approximate Solution of Time Dependent Problems, GARP Publications Series No. 10, 1973.

[25] H.P. Langtangen, G. Pedersen, Computational models for weakley dispersive nonlinear water waves, Computer Methods in Applied Mechanics and Engineering 160 (1998) 337–358.

[26] Y.S. Li, S.-X. Liu, O.W.H. Wai, Y.-X. Yu, Wave concentration by a navigation channel, Applied Ocean Research 22 (2000) 199–213.

[27] D.R. Lynch, W.G. Gray, A wave equation model for finite element tidal computations, Computers and Fluids 7 (1979) 207–228.

[28] H. Ma, A spectral element basin model for the shallow water equations, Journal of Computational Physics 109 (1993) 133–149.

[29] P.A. Madsen, I.R. Murray, O.R. Sørensen, A new form of the Boussinesq equations with improved linear dispersion characteristics, Coastal Engineering 15 (1991) 371–388.

[30] P.A. Madsen, O.R. Sørensen, A new form of the Boussinesq equations with improved linear dispersion characteristics. Part 2. A slowly-varying bathemetry, Coastal Engineering 18 (1992) 183–204.

[31] P.A. Madsen, H.A. Schäffer, Higher-order Boussinesq-type equations for surface gravity waves: derivation and analysis, Philosophical Transactions of the Royal Society London A 356 (1998) 3123–3184.

[32] P.A. Madsen, H.B. Bingham, H.A. Schäffer, Boussineq-type formulations for fully nonlinear and extremely dispersive water waves: derivation and analysis, Proceedings of the Royal Society London A 459 (2003) 1075–1104.

[33] O. Nwogu, Alternative form of Boussinesq equations for nearshore wave propagation, Journal of Waterway, Port, Coastal and Ocean Engineering 119 (1993) 618–638.

[34] D.H. Peregrine, Long waves on a beach, Journal of Fluid Mechanics 27 (1967) 815–827.

[35] J. Proriol, Sur une famile de polynomes á deux variables orthogonax dans un triangle, CR Acadamic Science Paris 245 (1957) 2459–2461.

[36] S.J. Sherwin, R.M. Kirby, J. Peió, R.L. Taylor, O.C. Zienkiewicz, On 2D elliptic discontinuous Galerkin methods, International Journal of Numerical Methods in Engineering (in press).

[37] S.J. Sherwin, Dispersion analysis of the continuous and discontinuous Galerkin formulations, in: B. Cockburn, G.E. Karniadakis, C.-W. Shu (Eds.), Discontinuous Galerkin Methods, Lecture Notes in Computational Science and Engineering, vol. 11, Springer, Berlin, 2000, pp. 425–431.

[38] F. Shi, R.A. Dalrymple, J.T. Kirby, Q. Chen, A. Kennedy, A fully nonlinear Boussinesq model in generalized curvilinear coordinates, Coastal Engineering 42 (2001) 337–358.

[39] F. Shi, J.T. Kirby, R.A. Dalrymple, Q. Chen, Wave simulations in Ponce de Leon inlet using Boussinesq model, Journal of Waterway, Port, Coastal and Ocean Engineering 129 (2003) 124–135.

[40] O.R. Sørensen, H.A. Schäffer, L.S. Sørensen, Boussinesq-type modelling using an unstructured finite element technique, Coastal Engineering 50 (2004) 181–198.

[41] M. Taylor, J. Tribbia, M. Iskandarani, The spectral element method for the shallow water equations on a sphere, Journal of Computational Physics 130 (1997) 92–108.

[42] E.F. Toro, Shock-Capturing Methods for Free-Surface Shallow Flows, Wiley, New York, 2001.

[43] M.A. Walkley, A numerical method for extended Boussinesq shallow-water wave equations, PhD thesis, University of Leeds, UK, 1999.

[44] K.-H. Wang, T.Y. Wu, G.T. Yates, Three-dimensional scattering of solitary waves by vertical cylinder, Journal of Waterway, Port, Coastal and Ocean Engineering 118 (1992) 551–566.

[45] G. Wei, J.T. Kirby, Time-dependent numerical code for extended Boussinesq equations, Journal of Waterway, Port, Coastal and Ocean Engineering 121 (1995) 251–261.

[46] R.W. Whalin, The limit of applicability of linear wave refraction theory in convergence zone, US Army Corps of Engineers, Research Report H-71-3, USA, 1971.

[47] S.-B. Woo, P.L.-F. Liu, Finite-element model for modified Boussinesq equations. I: Model development, Journal of Waterway, Port, Coastal and Ocean Engineering 130 (2004) 1–16.

[48] S.-B. Woo, P.L.-F. Liu, Finite-element model for modified Boussinesq equations. II: Application to nonlinear harbour oscillations, Journal of Waterway, Port, Coastal and Ocean Engineering 130 (2004) 17–28.

[49] Y. Xu, C.-W. Shu, Local discontinuous Galerkin methods for three classes of nonlinear wave equations, Journal of Computational Mathematics 22 (2004) 250–274.

[50] Y. Xu, C.-W. Shu, Local discontinuous Galerkin methods for nonlinear Schrödinger equations, Journal of Computational Physics 205 (2005) 72–97.

[51] J. Yan, C.-W. Shu, A local discontinuous Galerkin method for KdV type equations, SIAM Journal of Numerical Analysis 40 (2002) 769–791.