The Dissertation Committee for Ali Samii
certifies that this is the approved version of the following dissertation:

A Hybridized Discontinuous Galerkin Method for Nonlinear Dispersive Water Waves

Committee:

Clint Dawson, Supervisor

Leszek Demkowicz

Irene Gamba

Ben Hodges

Chad Landis

Laxminarayan Raja

A Hybridized Discontinuous Galerkin Method for Nonlinear Dispersive Water Waves

by

Ali Samii, B.E.; M.E.; Ph.D.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2017

To my parents.

Acknowledgments

First and foremost, I wish to express my love and gratitude to the one who bestowed on me the gift of life. Among all favors, he has given me the opportunity to learn and interact with many great people, who I wish to thank here.

This work was not possible without the help and guidance of Prof. Clint Dawson. The joy of working with a knowledgeable, patient, and humble adviser always kept me encouraged when facing different problems throughout this journey.

I would like to thank the committee members, Profs. Demkowicz, Gamba, Hodges, Landis, and Raja, for their time, input and participation.

During my work as part of the Computational Hydraulics Group (CHG) in the Institute for Computational Engineering and Sciences (ICES), I benefited from the companionship and discussion with my colleagues and friends. I appreciate their help, and will never forget the good times that we spent together.

Lastly, and maybe most important of all, I want to thank my family for their continuous love and support. Their kind attention has always kept me motivated in my whole life.

This work was funded in part by the National Science Foundation grant ACI 1339801. Their support is gratefully acknowledged.

A Hybridized Discontinuous Galerkin Method for Nonlinear Dispersive Water Waves

Publication No. _____

Ali Samii, Ph.D.
The University of Texas at Austin, 2017

Supervisor: Clint Dawson

Simulation of water waves near the coast is an important problem in different branches of engineering and mathematics. For mathematical models to be valid in this region, they should include nonlinear and dispersive properties of the corresponding waves. Here, we study the numerical solution to three equations for modeling coastal water waves using the hybridized discontinuous Galerkin method (HDG). HDG is known to be a more efficient and in certain cases a more accurate alternative to some other discontinuous Galerkin methods, such as local DG.

The first equation that we solve here is the Korteweg-de Vries equation. Similar to common HDG implementations, we first express the approximate variables and numerical fluxes in each element in terms of the approximate traces of the scalar variable, and its first derivative. These traces are assumed to be single-valued on each face. We next impose the conservation of the

numerical fluxes via two sets of equations on the element boundaries. We solve this equation by Newton-Raphson method. We prove the stability of the proposed method for a proper choice of stabilization parameters. Through numerical examples, we observe that for a mesh with kth order elements, the computed variable and its first and second derivatives show optimal convergence at order $k+1$ in both linear and nonlinear cases, which improves upon previously employed techniques.

Next, we consider solving the fully nonlinear irrotational Green-Naghdi equation. This equation is often used to simulate water waves close to the shore, where there are significant dispersive and nonlinear effects involved. To solve this equation, we use an operator splitting method to decompose the problem into a dispersive part and a hyperbolic part. The dispersive part involves an implicit step, which has regularizing effects on the solution of the problem. On the other hand, for the hyperbolic sub-problem, we use an explicit hybridized DG method. Unlike the more common implicit version of the HDG, here we start by solving the flux conservation condition for the numerical traces. Afterwards, we use these traces in the original PDEs to obtain the internal unknowns. This process involves Newton iterations at each time step for computing the numerical traces. Next, we couple this solver with the dispersive solver to obtain the solution to the Green-Naghdi equation. We then solve a set of numerical examples to verify and validate the employed technique. In the first example we show the convergence properties of the numerical method. Next, we compare our results with a set of experimental

data for nonlinear water waves in different situations. We observe close to optimal convergence rates and a good agreement between our numerical results and the experimental data.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Computational modeling of near-coastal water waves is a crucial field of research to understand the complicated behavior of the water flow in these regions. While engineers use such models to design and build effective coastal protection systems against hurricanes and storms, emergency managers use them to improve their preparedness in the case of such hazards. The mathematical models used to describe the fluid flow in such problems are based on the asymptotic expansion of the incompressible Euler's equation and removing higher order terms from the expansion. When studying off-shore waves, we usually consider nondispersive wave models where different wavelengths travel with the same speed. However, near the coast, short waves start to fall behind the longer waves and the shape of the waves start to evolve. This process, which is known as wave shoaling, results in the breaking of waves and dissipation of their energy. Including such phenomena in the simulations using nondispersive wave models results in significant errors. The key fact in choosing the correct wave model is knowing the scales dominating different regions of the problem. Throughout this work, we will follow the shallow water regime assumption, which are characterized by large length scales compared to the typical depth.

1.1 Shallow Water Regime

In order to identify the regime of the water waves, we need to introduce a number of length scales in the problem. These scales are shown in Fig. 1.1. Here, we have denoted the horizontal coordinates with $\mathbf{x} \in \mathbb{R}^d$, where d is the horizontal dimension of the problem. For instance, when we are solving a 2D problem, $\mathbf{x}$ is simply the x-coordinate and for 3D problems $\mathbf{x} = (x, y)$. At the given time t, we use D_t to denote the subset of $\mathbb{R}^{d+1}$ which is filled with water, and any point in D_t is identified by the coordinates $(\mathbf{x}, z) \in \mathbb{R}^{d+1}$. In this dissertation, we make the following assumptions:

- The fluid is inviscid, incompressible, irrotational, with constant and uniform density.
- Water surface and bottom can be presented as graphs, in the forms $z = \zeta(t, \mathbf{x})$ and $z = b(\mathbf{x})$, respectively. Hence, while the water surface can change as a function of time, the bottom boundary is taken constant in time.
- Fluid particles do not cross the top and bottom boundaries.
- The external pressure (P_0) is constant, and there is no surface tension.
- The only forcing applied to the water is the gravity force; hence, the wind and Coriolis forces are not considered in this study.

For reasons that will become evident later, we define four dominant length scales in our problem. As shown in Fig. 1.1 we use H_0 to denote the

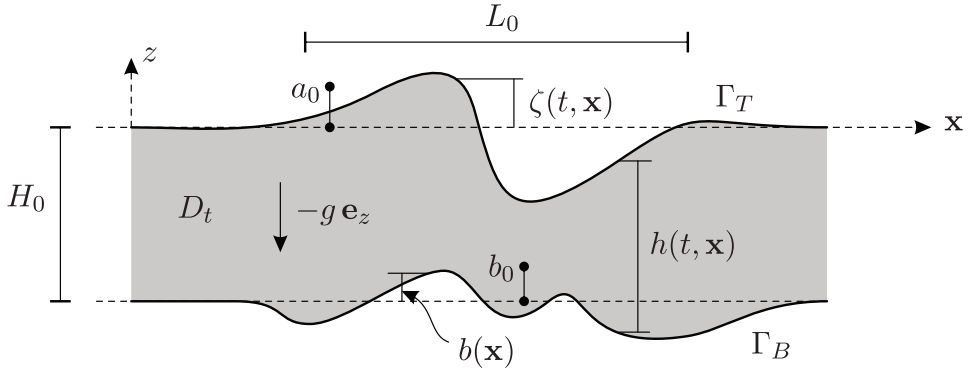


Figure 1.1: Domain of the problem, and the employed notations and length scales.

typical water height, L_0 for the typical horizontal scale, a_0 for the order of wave amplitude, and b_0 to denote the order of topography variation. Based on these length scales, we also define the following dimensionless parameters:

$$\varepsilon = \frac{a_0}{H_0}; \quad \mu = \frac{H_0^2}{L_0^2}; \quad \beta = \frac{b_0}{H_0}. \tag{1.1}$$

It is worthwhile noting that, while our definition of ε and β match the majority of literature, some researchers tend to take $\mu = H_0/L_0$. As we will see in the next chapter, ε controls how much nonlinearity we allow in our formulation, e.g. when $\varepsilon = O(1)$, the regime is known as large amplitude or fully nonlinear; moreover, $0 < \varepsilon \ll 1$ results in the weakly nonlinear regimes. Accordingly, ε is referred to as the *nonlinearity parameter*. Moreover, μ and β are the *shallowness* and *topography* parameters, respectively. In our study, we assume $\mu \ll 1$, but we do not set any special condition on ε and β.

As we will see in Chapter 2, by expressing the solution of the Euler's equation in terms of an asymptotic expansion with respect to μ, ε, β, we

arrive at an equation which is in terms of the increasing powers of these parameters. Now, if we assume $\mu \ll 1$, and for some $n \geq 0$ drop all terms containing μ^m with $m \geq n$, we say that the obtained equation is $O(\mu^n)$ consistent with the original Euler's equation. The *shallow water regime* is the set of all such equations, in which by dropping the higher order terms of μ we can achieve a desirable approximation to the original equation. This dissertation is devoted to develop a hybridized discontinuous Galerkin method for solving three such equations, i.e. Nonlinear Shallow Water Equation (NSWE), Green-Naghdi (GN) equation, and weakly nonlinear Korteweg-de Vries (KdV) equation. The details for derivation of these equations are discussed in the next chapter.

1.2 A Review of Numerical Procedures in Shallow Water Regime

Among the equations in the shallow water regime, NSWE has been studied by many researchers. This equation was first derived in the one-dimensional case, by Saint-Venant [24], assuming a linear variation of the pressure in the vertical direction. This work resulted in an equation with advection, diffusion, and eddy viscosity terms. As will be shown in the next chapter, the NSWE is $O(\mu)$ consistent with the water wave equation. Due to the complex geometries in which we need to solve this equation, the finite element method has been a viable option among other numerical methods. Most of the efforts in this context have been on the development of stable

techniques for advection dominated flows. The first such study was conducted using an implicit continuous Galerkin method with cubic Hermite functions to solve the 1D NSWE [71]. The 2D problem was later solved in Cartesian and Spherical coordinates, and including Coriolis forcing [21]. Most of these methods were based on primitive variables (velocity and surface elevation). It was later shown that approximating the primitive variables at the same nodal points can result in spurious oscillation and the idea of staggered grids for these variables were implemented [73]. Hence, most of the techniques developed later were based on using primitive variables in staggered grids [70]. However, it was known that using the formulation in terms of vorticity and divergence in non-staggered grids does not result in the same issues as primitive variables in such grids [74]. The developments in this context continued with conservative iterative methods [56], selective lumping finite elements [35], and improved stability methods [51]. One of the more popular techniques was based on wave continuity equation instead of water surface evolution equation [46], which was employed in a number of other works [36, 37, 45]. A priori error estimates were also provided for this method [8, 9].

A majority of the methods based on the continuous Galerkin (CG) formulation, have important drawbacks such as violating the local conservation, failing to satisfy the primitive variable continuity, and lack of stability. As a result, a number of important studies were carried out in the framework of discontinuous Galerkin methods (DG). These methods can naturally incorporate some of the important features mentioned above. By a proper choice of nu-

merical fluxes, one can add more stability to the solution, and including slope limiters in their formulations is quite straightforward [7]. They are locally conservative, which makes them appropriate choices when one wants to use them for particle tracking purposes [22]. Different variations of DG have been used to solve the system of nonlinear hyperbolic equations or specifically NSWE. One of the first DG formulations for such equations was proposed in [6]. A major upgrade to accuracy and stability of these methods was the Runge-Kutta local projection DG [12–14, 16], which was later applied to nonlinear convection-diffusion systems [10, 15]. Different studies have used coupled CG-DG methods for the simulation of shallow water systems [23].

One of the drawbacks of the DG methods compared to the CG variants is the large number of degrees of freedom that one needs to deal with when solving large problems. Hence, by writing the DG equations in hybridized form, one can get a comparably smaller global system along with a set of element-wise equations. This technique is known as the hybridized DG (HDG) and has been applied to a variety of problems, including nonlinear hyperbolic systems [11, 53, 54, 57]. The global system of equations in HDG needs to be solved on the skeleton space of the mesh, which results in lower number of degrees of freedom and smaller bandwidth of the global matrix. HDG has been usually used in combination with implicit time integrators; however, there are other implementations such as implicit-explicit methods [68], fully explicit methods [39, 65], and spacetime discretization [59].

A higher order approximation than NSWE to the water wave equation

is the Green-Naghdi equation (also known as Serre or fully nonlinear Bousinesq equation [72]). It was first derived by Serre for a 1D problem [63], and later for the 2D problems with general topography [31]. As we will show later, they are $O(\mu^2)$ consistent with the incompressible Euler's equation. They are usually used to solve the water wave problem in deeper regions compared to NSWE. For example, while inside the ocean the values of μ vary between $10^{-5} - 10^{-4}$, near the coast, μ is of order 10^{-2}. Hence, in the simulation of water waves approaching from the ocean to the coast, if we use an $O(\mu)$ model for the ocean (e.g. NSWE), we need an $O(\mu^2)$ model for the coast to keep the approximation order consistent throughout the computational domain.

An important feature that the Green-Naghdi equation offers is the ability to include dispersive characteristics of water waves. Simulation of dispersive water waves dates back to the 1980s when numerical methods were used to solve the run-up of non-breaking [47] and breaking [77] water waves. However, most of these methods were at best based on weakly nonlinear assumption, i.e. taking $\varepsilon = O(\mu)$ in the asymptotic expansion [50]. The equations in this regime are usually referred to as the *'classical' Boussinesq equations* or *Boussinesq–Peregrine models*. This regime is not suitable in many applications in coastal oceanography, where we need to model large amplitude waves. It was not until the 2000s that the first works on solving the fully nonlinear dispersive waves appeared [29, 30, 48]. Lannes et. al used an operator splitting technique to solve the equations derived in [41], using a combination of finite volume and finite difference methods [4]. These equations were later developed for 2D

simulations [42] and solved using local DG [28]. While most of these works were based on the assumptions that we made in Sec. 1.1, one can also find rotational fluid models employed to obtain such approximate equations [78]. This rotational model equations have been solved in [55] using local DG.

1.3 Contributions

The research work which is reported in this dissertation contains the following contributions:

- We have developed an implicit hybridized DG method for the nonlinear KdV equation. The KdV equation is one of the simplest dispersive equations, which involves third order derivatives. We prove the stability of our numerical scheme for the linear KdV, and show the convergence properties of the proposed method through numerical examples. The results of this part of the work is also reported in [62].

- We have developed a hybridized DG element in the deal.II finite element library. This element can be used in shared and distributed parallel frameworks. To this end, we have implemented a new scheme for numbering of the degrees of freedom and applying the boundary conditions independent of the library. The developed software has been shown to scale well up to 1024 computational cores. The results of using this element in solving the nonhomogeneous and anisotropic diffusion equation is reported in [61].

- We have solved the fully nonlinear irrotational Green-Naghdi equation in two dimensions using the hybridized discontinuous Galerkin method. The proposed method is validated against experimental benchmark tests, where we have found good agreement between our numerical results and the experimental data. This part of our work has also given rise to an explicit hybridized DG for solving nonlinear shallow water equations. To the best of our knowledge, this technique is among the first explicit implementation of HDG for solving a nonlinear conservation law.

1.4 Outline

In the forthcoming chapters, we first discuss the different wave models, which are solved in this dissertation, and explain how they are related to each other. In Chapter 3, we present our method for solving the KdV equation and show the corresponding results. In Chapter 4, we explain our explicit hybridized DG method for the solution of the nonlinear shallow water equation. In Chapter 5, we present our numerical method for solving the irrotational Green-Naghdi equation, and use a set of numerical experiments to verify and validate our proposed method. In Chapter 6, we mention our concluding remarks and a few directions to continue the current work.

Chapter 2

Wave Models for Shallow Water Regime

In this section we derive the governing equations of the water waves based on the assumptions that we introduced in Sec. 1.1. By substituting an asymptotic expansion of the solution into these equations we obtain the main wave equations that we plan to solve in the next three chapters. In this section, we follow the notation used by many other researchers in the water wave theory [40].

2.1 Free Surface Bernoulli Equations

At a given time t, let D_t denote the subset of $\mathbb{R}^{d+1}$, which is filled with water (refer to Fig. 2.1). At a given point $(\mathbf{x}, z) \in D_t$, let $\mathbf{U}(t, \mathbf{x}, z) \in \mathbb{R}^{d+1}$ denote the velocity of a fluid particle. Meanwhile $\mathbf{u}(t, \mathbf{x}, z) \in \mathbb{R}^d$ and $w(t, \mathbf{x}, z) \in \mathbb{R}$ are the horizontal and vertical components of the velocity. At this point $P(t, \mathbf{x}, z)$ denotes the pressure. The acceleration of gravity, which acts in the vertical direction $(-g\mathbf{e}_z)$, is taken constant everywhere. In what follows, we use ∇ to denote the gradient in the horizontal direction and $\underset{\sim}{\nabla}$ to denote $(\nabla, \partial_z)^T$. In order to maintain consistency, we also use Δ and $\underset{\sim}{\Delta}$ to denote ∇^2 and $\nabla^2 + \partial_z^2$, respectively.

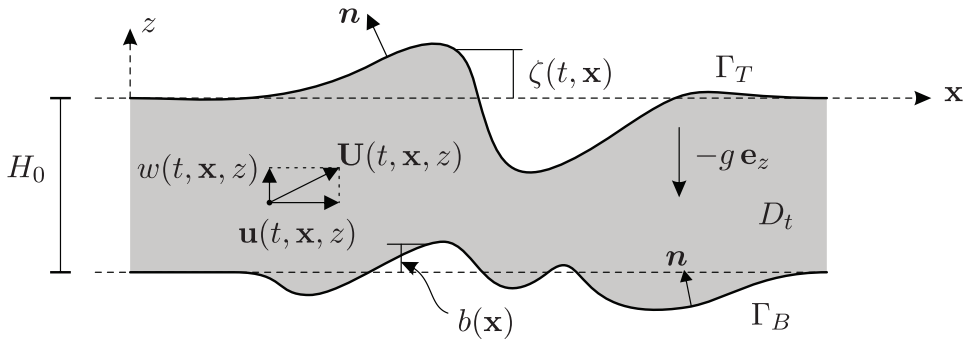


Figure 2.1: Domain of the problem, and the employed notation.

Based on the assumptions introduced in Sec. 1.1, the Euler's equation governs the flow, and the velocity field is both irrotational and solenoidal:

$$\partial_t \mathbf{U} + (\mathbf{U} \cdot \underset{\sim}{\nabla})\mathbf{U} = -\frac{1}{\rho}\underset{\sim}{\nabla} P - g\mathbf{e}_z \qquad \text{in } D_t, \tag{2.1a}$$

$$\underset{\sim}{\nabla} \cdot \mathbf{U} = 0 \qquad \text{in } D_t, \tag{2.1b}$$

$$\underset{\sim}{\nabla} \times \mathbf{U} = 0 \qquad \text{in } D_t. \tag{2.1c}$$

Meanwhile on Γ_T, we have $P = P_0$. The boundary conditions on Γ_T and Γ_B can be derived using the fact that the fluid particles do not cross these boundaries. Let $(\mathbf{x}_p(t), z_p(t))$ denote the position of a fluid particle on Γ_T. Since, the equation of Γ_T is $-z + \zeta(t, \mathbf{x}) = 0$, we have $z_p(t) - \zeta(t, \mathbf{x}_p(t)) = 0$. Moreover, the particle stays on Γ_T, and we have:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left[-z_p(t) + \zeta(t, \mathbf{x}(t))\right] = 0 \implies -\partial_t z_p + \partial_t \zeta + \nabla\zeta\, \partial_t \mathbf{x}_p = 0.$$

Knowing that $\partial_t z_p = w$, and $\partial_t \mathbf{x}_p = \mathbf{u}$,

$$\partial_t \zeta + \nabla\zeta \cdot \mathbf{u} - w = 0 \quad \text{ on } \Gamma_T. \tag{2.2}$$

This is the kinematic boundary condition on water surface. Now, the normal vector to Γ_T can be obtained by taking the gradient of the equation of this surface, i.e. $\boldsymbol{n} = (-\nabla\zeta, 1)/\sqrt{1+|\nabla\zeta|^2}$. Hence, (2.2) can be written as:

$$\partial_t\zeta - \sqrt{1+|\nabla\zeta|^2}\,\mathbf{U}\cdot\boldsymbol{n} = 0 \quad \text{ on } \Gamma_T. \tag{2.3}$$

similarly on Γ_B, we have:

$$\mathbf{U}\cdot\boldsymbol{n} = 0 \quad \text{ on } \Gamma_B. \tag{2.4}$$

Next, we use (2.1c), to express $\mathbf{U}$ in the form $\mathbf{U} = \nabla\Phi$, and obtain the free surface Bernoulli equations:

$$\partial_t\Phi + \frac{1}{2}|\underset{\sim}{\nabla}\Phi|^2 + gz = -\frac{1}{\rho}(P - P_0) \qquad \text{in } D_t, \tag{2.5a}$$

$$\underset{\sim}{\nabla}^2\Phi = 0 \qquad \text{in } D_t, \tag{2.5b}$$

$$\partial_t\zeta - \sqrt{1+|\nabla\zeta|^2}\partial_n\Phi = 0 \qquad \text{on } \Gamma_T, \tag{2.5c}$$

$$\partial_n\Phi = 0 \qquad \text{on } \Gamma_B. \tag{2.5d}$$

2.2 Dirichlet–Neumann (D–N) Operator

Here, we plan to transfer our d+1-dimensional problem to a d-dimensional problem. In the literature, this is usually done by a process of integrating the equations over the depth of the domain [44]. Basically, we want to remove any dependence on the vertical direction. To this end, we choose a more mathematically established approach. First we combine Eqs. (2.5b-d) into a single equation, which will be only in the horizontal direction. Second, we replace

the derivatives in the z-direction, using the chain rule. This scheme was first introduced by Zakharov [76] and later formulated in a robust procedure in [19, 20].

2.2.1 Definition of the D–N Operator

Let us define ψ as the trace of Φ at the water surface, i.e.: $\psi = \Phi|_{\Gamma_T}$. We use the well-developed theory of Laplace's equation to define an operator, which takes ψ as its input and having ζ, b, uses the following boundary value problem to solve for Φ:

$$\underset{\sim}{\Delta}\Phi = 0 \qquad \text{in } D_t, \tag{2.6a}$$

$$\Phi = \psi \qquad \text{on } \Gamma_T \text{ (i.e. } z = \zeta(t, \mathbf{x})), \tag{2.6b}$$

$$\partial_n \Phi = 0 \qquad \text{on } \Gamma_B \text{ (i.e. } z = -H_0 + b(\mathbf{x})). \tag{2.6c}$$

Under proper regularity assumptions, the above equation has a unique solution for a given ψ. We identify the following map:

$$G[\zeta, b] : \psi \mapsto \sqrt{1 + |\nabla\zeta|^2}\, \partial_n \Phi|_{\Gamma_T}, \tag{2.7}$$

or the form, which is more useful for practical purposes (compare (2.2) and (2.3)):

$$G[\zeta, b] : \psi \mapsto -\nabla\zeta \cdot \nabla\Phi|_{\Gamma_T} + \partial_z \Phi|_{\Gamma_T}, \tag{2.8}$$

which is known as the *Dirichlet–Neumann operator*. A rigorous definition of the D–N operator is beyond the scope of this dissertation. Here, we have assumed enough regularity for the unknowns, so that the solution of the above

boundary value problem exists in the distributional sense of the derivatives. We refer the interested reader to the original papers [19, 20], where the operator is defined in its appropriate functional settings, and is proved to be well-defined and continuous.

Now, we can write (2.5b – d) as a single equation:

$$\partial_t \zeta - G[\zeta, b]\psi = 0 \tag{2.9}$$

In the next step, we want to remove Φ and its z-derivatives from (2.5a). For this purpose, we obtain the three derivatives: $\partial_t \Phi$, $\partial_z \Phi$, and $\nabla \Phi$ at the water surface. For $\partial_t \Phi$ and $\nabla \Phi$, we use $\psi(t, \mathbf{x}) = \Phi(t, \mathbf{x}, \zeta(t, \mathbf{x}))$ and the chain rule to get:

$$\partial_t \Phi = \partial_t \psi - \partial_z \Phi \, \partial_t \zeta \quad \text{ on } \Gamma_T, \tag{2.10a}$$

$$\nabla \Phi = \nabla \psi - \partial_z \Phi \, \nabla \zeta \quad \text{ on } \Gamma_T. \tag{2.10b}$$

For $\partial_z \Phi$, we start from (2.2) and use (2.9) and (2.10) to obtain:

$$\partial_z \Phi = \frac{G[\zeta, b]\psi + \nabla \zeta \cdot \nabla \psi}{1 + |\nabla \zeta|^2} \quad \text{ on } \Gamma_T. \tag{2.10c}$$

Next, substituting (2.9) and (2.10) in (2.5a) results in the following equation on Γ_T:

$$\begin{aligned}
\partial_t \psi + g\zeta &= -\frac{1}{2}(\nabla \psi - \partial_z \nabla \zeta) \cdot (\nabla \psi - \partial_z \nabla \zeta) - \frac{1}{2}(\partial_z)^2 + \partial_z \Phi \, \partial_t \zeta \\
&= -\frac{1}{2}|\partial \psi|^2 - \frac{1}{2}\left(1 + |\nabla \zeta|^2\right)(\partial_z \Phi)^2 + (\partial_z \Phi)^2 + (\partial_z \Phi)^2 |\nabla \zeta|^2 \\
&= -\frac{1}{2}|\nabla \psi|^2 + \frac{1}{2}(1 + |\nabla \zeta|^2)(\partial_z \Phi)^2 \\
&= -\frac{1}{2}|\nabla \psi|^2 + \frac{(G[\zeta, b]\psi + \nabla \zeta \cdot \nabla \psi)^2}{2(1 + |\nabla \zeta|^2)}.
\end{aligned} \tag{2.11}$$

In the above derivation, from first to second step, we have used (2.2), i.e. $\partial_t\zeta = -\nabla\zeta\cdot\nabla\Phi + \partial_z\Phi$ and substituted $\nabla\Phi$ and $\partial_z\Phi$ from (2.10).

Thus we arrive at the following two equations, which are written on the water surface (a d-dimensional manifold) and only contain horizontal derivatives:

$$\begin{cases} \partial_t\zeta - G[\zeta,b]\psi = 0, \\ \partial_t\psi + g\zeta + \dfrac{1}{2}|\nabla\psi|^2 - \dfrac{(G[\zeta,b]\psi + \nabla\zeta\cdot\nabla\psi)^2}{2(1+|\nabla\zeta|^2)} = 0. \end{cases} \tag{2.12}$$

2.2.2 Relationship between D–N Map and the Depth Averaged Momentum

In order to establish a relationship between D–N map and the depth averaged velocity ($\bar{\mathbf{u}}$), we start from the definition:

$$\bar{\mathbf{u}}(t,\mathbf{x}) = \frac{1}{h(t,\mathbf{x})}\int_{-H_0+b(t,\mathbf{x})}^{\zeta(t,\mathbf{x})} \nabla\Phi(t,\mathbf{x},z)\,dz,$$

and take the divergence of the integral in the horizontal direction, i.e.:

$$\begin{aligned} \nabla\cdot(h\bar{\mathbf{u}}) &= \nabla\cdot\int_{-H_0+b}^{\zeta}\nabla\Phi\,dz \\ &= \int_{-H_0+b}^{\zeta}\nabla^2\Phi\,dz + \nabla\zeta\cdot\nabla\Phi|_{\Gamma_T} - \nabla b\cdot\nabla\Phi|_{\Gamma_B}. \end{aligned}$$

Due to the irrotational flow, $\underset{\sim}{\Delta}\Phi = \nabla^2\Phi + \partial_z^2\Phi = 0$, and the term inside the above integral is $-\partial_z^2\Phi$. Furthermore, $\partial_n\Phi|_{\Gamma_B} = 0$, which can be written as $\nabla b\cdot\nabla\Phi|_{\Gamma_B} - \partial_z\Phi|_{\Gamma_B} = 0$. Hence, the last term in the above relation can be substituted with $\partial_z\Phi|_{\Gamma_B}$. As a result:

$$\begin{aligned} \nabla\cdot(h\bar{\mathbf{u}}) &= \int_{-H_0+b}^{\zeta} -\partial_z^2\Phi\,dz + \nabla\zeta\cdot\nabla\Phi|_{\Gamma_T} - \partial_z\Phi|_{\Gamma_B} \\ &= -\partial_z\Phi|_{\Gamma_T} + \nabla\zeta\cdot\nabla\Phi|_{\Gamma_T} \end{aligned} \tag{2.13}$$

Comparing this with (2.8), we find that:

$$G[\zeta, b]\psi = -\nabla \cdot (h\bar{\mathbf{u}}). \tag{2.14}$$

Substituting this relation into the first equation of (2.12), one gets the familiar form of kinematic condition, i.e. $\partial_t \zeta + \nabla \cdot (h\bar{\mathbf{u}}) = 0$. Since, we mainly use the velocity potential to derive the asymptotic equations, we prefer to keep our equations in terms of this variable instead of velocity and water height, for now.

2.3 Nondimensionalization and Dominant Scales

We have already defined the length scales: L_0, H_0, a_0, b_0 (refer to Fig. 1.1, and Sec. 1.1). Now, we want to define a time scale. The way we do this is by first obtaining a wave speed scale and then dividing a length scale by it to get the time scale. The velocity scale that we use here is the phase speed of the solution of the linearized equation about the rest state on a flat bottom, i.e. when $\zeta = 0, b = 0$. In this case (2.6) becomes:

$$\underset{\sim}{\Delta}\Phi = 0 \qquad \text{in } D_t, \tag{2.15a}$$

$$\Phi = \psi \qquad \text{on } \Gamma_T \text{ (i.e. } z = 0), \tag{2.15b}$$

$$\partial_z \Phi = 0 \qquad \text{on } \Gamma_B \text{ (i.e. } z = -H_0). \tag{2.15c}$$

We also linearize (2.12) about $\zeta = 0, b = 0$, to get:

$$\begin{cases} \partial_t \zeta - G[0,0]\psi = 0, \\ \partial_t \psi + g\zeta = 0. \end{cases} \tag{2.16}$$

Which can be combined to form the following equation for ζ:

$$\partial_t^2 \zeta + gG[0,0]\zeta = 0. \tag{2.17}$$

For our purpose, we take the Fourier transform of (2.15) with respect to the horizontal variable $\mathbf{x}$:

$$\frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} \left[\partial_z^2 \Phi + \nabla^2 \Phi(\xi, z)\right] e^{-i\mathbf{x}\cdot\xi}\, dx = 0$$
$$\therefore \quad \partial_z^2 \hat{\Phi}(\xi, z) - |\xi|^2 \hat{\Phi}(\xi, z) = 0$$

With the boundary conditions $\partial_z \hat{\Phi} = 0$ on $z = -H_0$, and $\hat{\Phi} = \hat{\psi}$ on $z = 0$. The solution to this equation is $\hat{\Phi}(\xi, z) = C_1(\xi)\sinh(|\xi|z) + C_2(\xi)\cosh(|\xi|z)$, and after applying the boundary conditions: $C_1(\xi) = \tanh(|\xi|H_0)\hat{\psi}(\xi), C_2(\xi) = \hat{\psi}(\xi)$. This results in the following form for $\hat{\Phi}$:

$$\hat{\Phi}(\xi, z) = \frac{\cosh\left[(z + H_0)|\xi|\right]}{\cosh(H_0|\xi|)} \hat{\psi}(\xi). \tag{2.18}$$

Hence, the formulation for the Dirichlet-Neumann map in the wavenumber domain becomes:

$$(\partial_z \hat{\Phi})(\xi, 0) = |\xi| \tanh(H_0|\xi|)\hat{\psi}(\xi)$$

And (2.17) becomes:

$$\partial_t^2 \hat{\zeta}(\xi) + g|\xi| \tanh(H_0|\xi|)\hat{\zeta}(\xi) = 0.$$

Next we take the Fourier transform of the above equation with respect to time to obtain the variation of the frequency (ω) as a function of the wave number$|\xi|$:

$$\omega(\xi) = \sqrt{g|\xi| \tanh(H_0|\xi|)},$$

which is the *dispersion relation* for linear waves. It gives the frequency of oscillation as a function of the magnitude of the wavenumber vector ($|\xi|$). The wavelength of the corresponding waves is given by $\lambda = 2\pi/|\xi|$. Next, by calculating the ratio $\omega(\xi)/|\xi|$, one obtains the *phase speed* (or *celerity*) of linear water waves:

$$c_P(|\xi|H_0) = \sqrt{gH_0}\frac{\sqrt{\tanh(|\xi|H_0)}}{\sqrt{|\xi|H_0}}, \tag{2.19}$$

and we use ν_ξ to denote:

$$\nu_\xi(|\xi|H_0) = \frac{\tanh(|\xi|H_0)}{|\xi|H_0} = \frac{\tanh(2\pi H_0/\lambda)}{2\pi(H_0/\lambda)}.$$

Now, having (2.19), we can define our wave speed scale c_0 by substituting ν_ξ with ν as follows:

$$c_0 = \sqrt{gH_0\nu} \quad \text{, with} \quad \nu = \frac{\tanh(2\pi\sqrt{\mu})}{2\pi\sqrt{\mu}}. \tag{2.20}$$

Here, μ is the shallowness parameter and is defined according to (1.1). Intuitively, μ characterizes the typical wavenumber in the problem. The variation of ν as a function of μ is shown in Fig. 2.2. When we are dealing with the shallow water regime or long waves ($\mu \ll 1$), we can assume that $\nu \simeq 1$. In other words by a small change in λ, ν_ξ does not vary significantly, and $c_0 = \sqrt{gH_0}$ is the traveling speed of the majority of wavelengths. Thus the water waves are nondispersive in very shallow waters. One can also refer to Fig. 2.3, where we have shown the variation of normalized phase velocity ($c_P/\sqrt{gH_0}$) as a function of $|\xi|H_0$. On the other hand, when we have intermediate values for μ, the

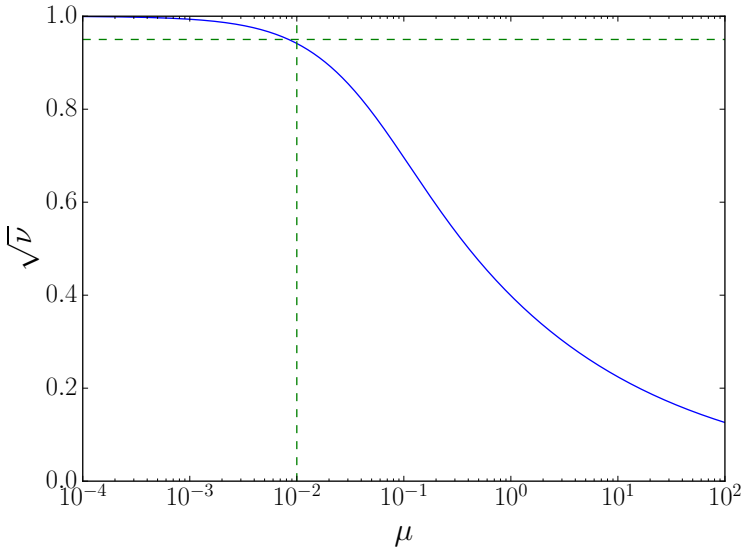


Figure 2.2: Variation of $\sqrt{\nu}$, with respect to shallowness parameter μ; the dashed lines represent $\mu = 10^{-2}$ and $\sqrt{\nu} = 0.95$.

longer waves travel faster than the shorter waves and the water waves show dispersive properties.

For future reference, we also derive the formula for the *group velocity* of the linear waves. The group velocity characterizes the speed at which the wave packet travels. The overall shape of the wave packet is formed by the superposition of different wavelengths with different frequencies. If the dispersion relation is linear (i.e. the wave is nondispersive) then the wave packet and the energy travel with the same speed as the global phase velocity. However, in dispersive waves, the wave packet travels at a lower speed compared to its wave constituents. The group velocity can be obtained as the gradient of $\omega(\xi)$

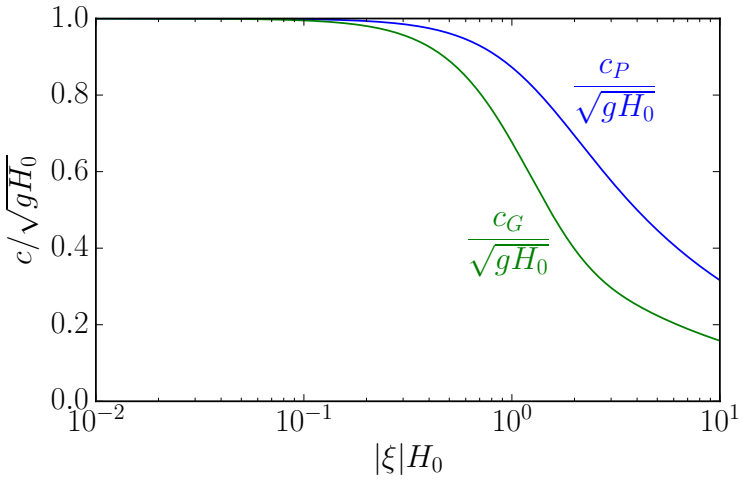


Figure 2.3: Normalized group and phase velocity of linear waves for different wavenumbers.

with respect to ξ; hence:

$$c_G(|\xi|H_0) = \sqrt{gH_0}\frac{|\xi|H_0\,\mathrm{sech}^2(|\xi|H_0) + \tanh(|\xi|H_0)}{2\sqrt{|\xi|H_0\tanh(|\xi|H_0)}} \tag{2.21}$$

The graphs of the phase and group velocity for different wavenumbers are plotted in Fig. 2.3.

Finally, we can define our time scale based on the length scale (L_0) and the characteristic speed c_0:

$$t_0 = \frac{L_0}{\sqrt{gH_0\nu}} \tag{2.22}$$

2.3.1 Scaling the Variables and Operators

In order to obtain the nondimensionalized equations, we first define the scaled space and time coordinates:

$$\mathbf{x}' = \frac{\mathbf{x}}{L_0}, \quad z' = \frac{z}{H_0}, \quad t' = \frac{t}{t_0}, \tag{2.23}$$

and their corresponding derivatives:

$$\nabla' = L_0\nabla, \quad \underset{\sim}{\nabla}' = L_0(\nabla, \sqrt{\mu}\,\partial_z)^T, \quad \underset{\sim}{\Delta}' = L_0^2(\nabla^2 + \mu\,\partial_z^2), \quad \partial_{t'} = t_0\partial_t. \tag{2.24}$$

For future reference, we also recall $\underset{\sim}{\nabla}$ and $\underset{\sim}{\Delta}$ in terms of nondimensionalized coordinates:

$$\underset{\sim}{\nabla} = \frac{1}{H_0}(\sqrt{\mu}\,\nabla', \partial_{z'})^T, \quad \underset{\sim}{\Delta} = \frac{1}{H_0^2}(\mu\,\nabla'^2 + \partial_{z'}^2). \tag{2.25}$$

The scaling of ζ, b, and $h = \zeta + H_0 - b$ is quite straightforward:

$$\zeta' = \frac{\zeta}{a_0}, \quad b' = \frac{b}{b_0}, \quad h' = \frac{h}{H_0}. \tag{2.26}$$

To find the order of magnitude of Φ and ψ, we first refer to (2.18), which states that Φ and ψ should have the same order of magnitude. Next, we look at the second equation of (2.16) to get:

$$\frac{1}{t_0}\partial_{t'}\psi_0\psi' + ga_0\zeta' = 0 \implies \psi_0 = ga_0t_0 = \frac{a_0}{H_0}L_0\sqrt{\frac{gH_0}{\nu}}.$$

Hence, we define the scaled velocity potential as:

$$\Phi' = \frac{1}{\Phi_0}\Phi, \quad \text{with} \quad \Phi_0 = \frac{a_0}{H_0}L_0\sqrt{\frac{gH_0}{\nu}}. \tag{2.27}$$

Now, since we have defined Φ_0, we define a velocity scale $\mathbf{u}_0$ based on the identity: $\mathbf{u}_0 = \nabla\Phi_0$, or in nondimensional form: $\mathbf{u}_0\mathbf{u}' = \Phi_0/L_0\nabla'\Phi'$. Hence, the dimensionless horizontal velocity finds the form:

$$\mathbf{u}' = \frac{1}{\mathbf{u}_0}\mathbf{u}, \quad \text{with} \quad \mathbf{u}_0 = \frac{\Phi_0}{L_0} = \frac{\varepsilon}{\sqrt{\nu}}\sqrt{gH_0}. \tag{2.28}$$

Before obtaining the nondimensionalized forms of the equations, we consider scaling the Dirichlet-Neumann operator. According to (2.2), we know that $G[\zeta, b]\psi = \partial_z\Phi - \nabla\Phi\cdot\nabla\zeta$ on the water surface. By substituting the derivatives and variables from (2.24) – (2.27), we have:

$$G[\zeta, b]\psi = \frac{\Phi_0}{H_0}(\partial_{z'}\Phi' - \mu\nabla'(\varepsilon\zeta')\cdot\nabla\Phi')|_{z'=\varepsilon\zeta}.$$

Thus we define:

$$G'[\varepsilon\zeta', \beta b']\psi' \coloneqq (\partial_{z'}\Phi' - \mu\nabla'(\varepsilon\zeta')\cdot\nabla\Phi')|_{z'=\varepsilon\zeta}, \tag{2.29}$$

to get:

$$G[\zeta, b]\psi = \frac{\Phi_0}{H_0}G'[\varepsilon\zeta', \beta b']\psi'. \tag{2.30}$$

2.3.2 Nondimensionalization of the Equations

We first obtain the nondimensionalized version of the boundary value problem (2.6). Using the definitions in the previous section, this equation takes the form:

$$\mu(\nabla')^2\Phi' + \partial_{z'}^2\Phi' = 0 \qquad \text{in } -1 + \beta b' \le z' \le \varepsilon\zeta', \tag{2.31a}$$

$$\Phi' = \psi' \qquad \text{on } z' = \varepsilon\zeta'(t, \mathbf{x}), \tag{2.31b}$$

$$\partial_{z'}\Phi' - \mu\nabla'(\beta b')\cdot\nabla'\Phi' = 0 \qquad \text{on } z = -1 + \beta b'(\mathbf{x}). \tag{2.31c}$$

Next, we write Eqs. (2.12) in the nondimensionalized form. We show the procedure for the first equation, and only write the result for the second equation. As we have already defined, $\zeta = a_0\zeta'$, $\partial_t = (1/t_0)\partial_{t'}$, and $G[\zeta, \beta]\psi = (\Phi_0/H_0)G'[\varepsilon\zeta', \beta b']\psi'$. Hence, the first equation in (2.12) becomes:

$$\frac{1}{t_0}\partial_{t'}(a_0\zeta') - \frac{\Phi_0}{H_0}G'[\varepsilon\zeta', \beta b']\psi' = 0 \implies \partial_{t'}\zeta' - \frac{t_0\Phi_0}{a_0H_0}G'[\varepsilon\zeta', \beta b']\psi' = 0.$$

One can simply check the identity $t_0\Phi_0/a_0H_0 = 1/\mu\nu$. The process for the second equation is similar, but requires a little more work. We leave that part to the reader and write the final nondimensionalized equations:

$$\begin{cases} \partial_{t'}\zeta' - \dfrac{1}{\mu\nu}G'[\varepsilon\zeta', \beta b']\psi' = 0, \\ \partial_{t'}\psi' + \zeta' + \dfrac{\varepsilon}{2\nu}|\nabla\psi'|^2 \\ \qquad\qquad - \dfrac{\varepsilon\mu}{\nu}\dfrac{\left(\frac{1}{\mu}G'[\varepsilon\zeta', \beta b']\psi' + \nabla'(\varepsilon\zeta')\cdot\nabla'\psi'\right)^2}{2(1+\varepsilon^2\mu|\nabla'\zeta'|^2)} = 0. \end{cases} \tag{2.32}$$

It will be also useful to write (2.14) in the dimensionless form. This will be a straightforward application of definitions for ∇', Φ', h', and $\mathbf{u}'$:

$$G'[\varepsilon\zeta', \beta b']\psi' = -\mu\,\nabla'\cdot(h'\bar{\mathbf{u}}') \tag{2.33}$$

2.4 Shallow Water Models

In the previous sections we defined four dimensionless parameters μ, ε, β, and ν. We mentioned that $\mu \ll 1$ characterizes the shallow water regime and in this case $\nu = 1$. If we do not make any assumption on the typical magnitude of topography and wave amplitude, we can take $\varepsilon \sim 1, \beta \sim 1$. The

equations which will be derived by this assumptions are called fully nonlinear equations. As we will see both Nonlinear Shallow Water (NSW) and Green-Naghdi equations belong to this group.

Since $\mu \ll 1$, we consider approximating the velocity potential based on the following asymptotic expansion:

$$\Phi'(t, \mathbf{x}, z) = \sum_{n=0}^{N} \mu^n \Phi'_n(t, \mathbf{x}, z) + O(\mu^{N+1}) \tag{2.34}$$

Hence, by including only the first term in the above sum, we approximate Φ up to $O(\mu^{N+1})$. Thus such a model is called $O(\mu^{N+1})$-consistent with the solution to original water wave problem. Here, we only consider $O(\mu)$ and $O(\mu^2)$ models.

Substituting (2.34) to the boundary value problem (2.31), and arranging the terms with the same power of μ, one gets:

$$\partial^2_{z'} \Phi'_n = \begin{cases} 0, & \text{for } n = 0, \\ -(\nabla')^2 \Phi'_{n-1}, & \text{otherwise.} \end{cases} \tag{2.35}$$

Meanwhile, we let Φ'_0 satisfy the boundary condition on the top and set the homogeneous boundary condition for other Φ'_n's. Thus the boundary conditions find the form:

$$\Phi'_n = \begin{cases} \psi', & \text{for } n = 0, \\ 0, & \text{otherwise,} \end{cases} \qquad \text{for} \quad z' = \varepsilon\zeta', \tag{2.36}$$

$$\partial_{z'} \Phi'_n = \begin{cases} 0, & \text{for } n = 0, \\ \beta \nabla' b' \cdot \nabla' \Phi'_{n-1} & \text{otherwise,} \end{cases} \qquad \text{for} \quad z' = -1 + \beta b'. \tag{2.37}$$

We have to solve a simple ODE to obtain the solution to Φ'_0. Afterwards, the solution to Φ'_1 will be obtained by substituting Φ'_0 in the above

equations and solving another ODE. The process is straightforward, and can be done using a computer algebra software. Thus, we will have:

$$\Phi'_0 = \psi' \tag{2.38}$$

$$\begin{aligned}\Phi'_1 =& -\frac{\nabla'^2\psi'}{2}z'^2 + \left[(-1+\beta b')\nabla'^2\psi' + \beta\nabla b'\cdot\nabla\psi\right] z \\ &+ \frac{\nabla'^2\psi}{2}\varepsilon^2\zeta'^2 - \varepsilon\zeta'(-1+\beta b')\nabla'^2\psi' - \varepsilon\beta\zeta'\nabla' b'\cdot\nabla'\psi' \\ =& -\frac{\nabla'^2\psi'}{2}z'^2 + \left[(-1+\beta b')\nabla'^2\psi' + \beta\nabla b'\cdot\nabla\psi\right] z \\ &+ \frac{\nabla'^2\psi}{2}\left[h'^2 - (1-\beta b')^2\right] - \beta(h'-1+\beta b')\nabla' b'\cdot\nabla'\psi'\end{aligned} \tag{2.39}$$

It is worthwhile noting that for an $O(\mu)$ model, the velocity potential is constant in depth. This means, the velocity field does not depend on the z-coordinate in the $O(\mu)$ models (e.g. NSWE). Also, the vertical component of the velocity, i.e. $w' = \partial_{z'}\Phi'$ vanishes in these models. On the other hand, in $O(\mu^2)$ models, such as Green-Naghdi equation, the velocity varies quadratically in depth.

Next, let us obtain the velocity variation corresponding to Φ'_0 and Φ'_1. In the nondimensionalized coordinates we have:

$$\bar{\mathbf{u}}'_n = \frac{1}{h'}\int_{-1+\beta b'}^{\zeta'} \nabla\Phi'_n\, dz'.$$

Hence, we can obtain $\bar{\mathbf{u}}_0$ and $\bar{\mathbf{u}}_1$ by some algebraic manipulations. Here, we use Wolfram Mathematica to obtain the following formula for these averaged velocities:

$$\bar{\mathbf{u}}'_0 = \nabla'\psi', \tag{2.40}$$

$$\bar{\mathbf{u}}'_1 = -\mu T'[h', b']\nabla'\psi', \tag{2.41}$$

where,

$$T'[h',b']\mathbf{w} = R_1'[h',b'](\nabla' \cdot \mathbf{w}) + \beta R_2'[h,b'](\nabla b' \cdot \mathbf{w}), \tag{2.42}$$

and,

$$R_1'[h',b']w = -\frac{1}{3h'}\nabla'(h'^3 w) - \beta\frac{h'}{2}w\nabla' b', \tag{2.43a}$$

$$R_2'[h',b']w = \frac{1}{2h'}\nabla'(h'^2 w) + \beta w\nabla' b'. \tag{2.43b}$$

Now, we can write the average velocity as:

$$\bar{\mathbf{u}}' = \nabla'\psi' - \mu T'[h',b']\nabla'\psi' + O(\mu^2) \tag{2.44}$$

Therefore, $\nabla'\psi' = \bar{\mathbf{u}}' + \mu T'\nabla\psi' + O(\mu^2)$. Substituting $\nabla'\psi'$ from this relation into itself, will result in:

$$\begin{aligned}\nabla'\psi' &= \bar{\mathbf{u}}' + \mu T'[h',b']\bar{\mathbf{u}}' + \mu^2 T'[h',b']\left(T'[h',b']\nabla'\psi'\right) + O(\mu^2)\\ \therefore \quad \nabla'\psi' &= \bar{\mathbf{u}}' + \mu T'[h',b']\bar{\mathbf{u}}' + O(\mu^2).\end{aligned} \tag{2.45}$$

This relation is our last piece of machinery to derive asymptotic water wave equations. In the remainder of this chapter we obtain three main equations in the water wave theory. In Table 2.1 we have summarized the main features of a number of shallow water models based on the considered range of the dimensionless parameters.

2.4.1 Nonlinear Shallow Water Equation

In order to obtain the nonlinear shallow water equation, we start with (2.32) and use (2.33) to substitute $G'\psi'$ with $-\mu\nabla' \cdot (h'\bar{\mathbf{u}}')$. We also take the

Table 2.1: Range of nonlinearity parameter (ε), topography parameter (β), and the spatial dimension for different shallow water models and the corresponding precision order

Class	Model	ε	β	d	Precision
Fully Nonlinear	Saint-Venant (NSWE)	$O(1)$	$O(1)$	1,2	$O(\mu t)$
	Green-Naghdi	$O(1)$	$O(1)$	1,2	$O(\mu^2 t)$
Moderately Nonlinear	Camassa-Holm	$O(\sqrt{\mu})$	0	1	$O(\mu^2 t)$
Weakly Nonlinear	Bossinesq-Peregrine	$O(\mu)$	$O(\mu)$	1,2	$O(\mu^2 t)$
	KdV	$O(\mu)$	0	1	$O(\mu^2 t)$

gradient of the second equation of (2.32), and use (2.33) and $\nabla'\psi' = \bar{\mathbf{u}}'$ to replace $G'\psi'$ and $\nabla'\psi'$ in terms of $\bar{\mathbf{u}}'$. We also drop all terms of order $O(\mu^N)$, with $N \geq 1$. Thus, we will get:

$$\begin{cases} \partial_{t'}\zeta' + \nabla' \cdot (h'\bar{\mathbf{u}}') = 0, \\ \partial_{t'}\bar{\mathbf{u}}' + \nabla'\zeta' + \varepsilon\bar{\mathbf{u}}' \cdot \nabla\bar{\mathbf{u}}' = 0. \end{cases} \tag{2.46}$$

We usually prefer the equations to be in terms of the conserved variables, i.e. $h, h\bar{\mathbf{u}}$. Hence, we use $\partial_{t'}h' = \varepsilon\partial_{t'}\zeta'$, and $\partial_{t'}\mathbf{u}' = [\partial_{t'}(h'\bar{\mathbf{u}}') + \varepsilon\bar{\mathbf{u}}'\nabla' \cdot (h'\bar{\mathbf{u}}')]/h$ in the second equation to obtain:

$$\begin{cases} \partial_{t'}h' + \varepsilon\nabla' \cdot (h'\bar{\mathbf{u}}') = 0, \\ \partial_{t'}(h'\bar{\mathbf{u}}') + \varepsilon\nabla' \cdot (h'\bar{\mathbf{u}}' \otimes \bar{\mathbf{u}}') + \dfrac{1}{\varepsilon}h'(\nabla'(h' + \beta b')) = 0. \end{cases} \tag{2.47}$$

Finally, we can write the equations with dimensions:

$$\begin{cases} \partial_t h + \nabla \cdot (h\bar{\mathbf{u}}) = 0, \\ \partial_t(h\bar{\mathbf{u}}) + \nabla \cdot (h\bar{\mathbf{u}} \otimes \bar{\mathbf{u}}) + gh\nabla h + gh\nabla b = 0. \end{cases} \tag{2.48}$$

We will explain the solution technique for this equation in chapter 4.

2.4.2 Green-Naghdi Equation

The process for obtaining Green-Naghdi equation is similar to NSWE; however, in the final step, instead of dropping all terms containing powers of μ, we drop the terms of order $O(\mu^N)$, with $N > 1$. This derivation is a little more involved than NSWE, and we only give the final equations. In order to avoid excessive verbosity, we denote such operators as $T'[h', b']$ without their arguments, i.e. T'. Then, the Green-Naghdi equations in terms of the dimensionless variables reads as:

$$\begin{cases} \partial_{t'}\zeta' + \nabla' \cdot (h'\bar{\mathbf{u}}') = 0, \\ (I + \mu T')(\partial_{t'}\bar{\mathbf{u}}') + \nabla'\zeta' + \varepsilon(\bar{\mathbf{u}}' \cdot \nabla')\bar{\mathbf{u}}' + \varepsilon\mu Q'(\bar{\mathbf{u}}') = 0. \end{cases} \tag{2.49}$$

With T' defined in (2.42), and Q' is defined in terms of R_1' and R_2', which were introduced in (2.43):

$$Q'(\mathbf{w}) = R_1' \left(\nabla' \cdot (\mathbf{w}\nabla' \cdot \mathbf{w}) - 2(\nabla' \cdot \mathbf{w})^2\right) + \beta R_2' \left((\mathbf{w} \cdot \nabla')^2 b'\right) \tag{2.50}$$

It is easy to check that Q' contains third order derivatives of the velocity field, which can be avoided by introducing a new operator Q_1' as follows:

$$Q_1'[h', b'](\mathbf{w}) = T'[h', b'] \left((\mathbf{w} \cdot \nabla)\mathbf{w}\right) - Q'[h', b'](\mathbf{w}). \tag{2.51}$$

Now, Q_1' contains up to second derivatives, and has the form:

$$\begin{aligned} Q_1'(\mathbf{w}) = &- 2R_1' \left(\partial_{x'}\mathbf{w} \cdot \partial_{y'}\mathbf{w}^\perp + (\nabla' \cdot \mathbf{w})^2\right) \\ &+ \beta R_2' \left(\mathbf{w} \cdot (\mathbf{w} \cdot \nabla')\nabla' b'\right) \end{aligned} \tag{2.52}$$

Here, $\mathbf{w}^\perp = (-\mathbf{w}_2, \mathbf{w}_1)^T$; meanwhile, $\partial_{x'}$, and $\partial_{y'}$ are the partial derivatives with respect to x', and y' respectively. Using this definition, the equation

(2.49) becomes:

$$\begin{cases} \partial_{t'}\zeta' + \nabla' \cdot (h'\bar{\mathbf{u}}') = 0, \\ (I + \mu T') \left(\partial_{t'}\bar{\mathbf{u}}' + \varepsilon(\bar{\mathbf{u}}' \cdot \nabla')\bar{\mathbf{u}}'\right) + \nabla'\zeta' + \varepsilon\mu Q_1'(\bar{\mathbf{u}}') = 0. \end{cases} \tag{2.53}$$

Similar to the previous section, we prefer the equations in terms of $h', h'\bar{\mathbf{u}}'$:

$$\begin{cases} \partial_{t'}h' + \varepsilon\nabla' \cdot (h'\bar{\mathbf{u}}') = 0, \\ \left(I + \mu h' T' \frac{1}{h'}\right) \left(\partial_{t'}(h'\bar{\mathbf{u}}') + \varepsilon\nabla' \cdot (h'\bar{\mathbf{u}}' \otimes \bar{\mathbf{u}}')\right) + h'\nabla'\zeta' + \varepsilon\mu h' Q_1'(\bar{\mathbf{u}}') = 0. \end{cases} \tag{2.54}$$

As we will see in chapter 5, solving this equation can be simplified, if we apply the inverse operator $\left(I + \mu h T' \frac{1}{h}\right)^{-1}$ on the second equation. Afterwards, we go back to the unknowns with dimensions and the above system becomes:

$$\begin{cases} \partial_t h + \nabla \cdot (h\bar{\mathbf{u}}) = 0, \\ \partial_t(h\bar{\mathbf{u}}) + \nabla \cdot (h\bar{\mathbf{u}} \otimes \bar{\mathbf{u}}) + \left(I + \mu h T \frac{1}{h}\right)^{-1} (gh\nabla\zeta + hQ_1(\bar{\mathbf{u}})) = 0. \end{cases} \tag{2.55}$$

The operators T and Q_1 with dimensions are according to (2.42) and (2.52) with $\beta = 1$, respectively. We will solve this equation in chapter 5, with a small modification which improves its dispersive properties.

2.4.3 Korteweg–de Vries Equation (KdV)

The KdV equation was first derived by Korteweg and De Vries [38], and the existence of its solution was proved in [18]. Unlike the previous two equations which were categorized as fully nonlinear models, the KdV equation is a representative of weakly nonlinear models, i.e. $\mu \ll 1$ and $\varepsilon \sim \mu$. In the special case of KdV equation, we also take $\beta = 0$. As a result, we can write equation (2.32) in a 1D setup as follows:

$$\begin{cases} \partial_{t'}\zeta' + \partial_{x'}(h'\bar{\mathbf{u}}') = 0, \\ \left(1 - \frac{\mu}{3}\partial_{x'}^2\right) \left(\partial_{t'}\bar{\mathbf{u}}'\right) + \partial_{x'}\zeta' + \frac{\varepsilon}{2}\bar{\mathbf{u}}'\partial_{x'}\bar{\mathbf{u}}' = 0. \end{cases} \tag{2.56}$$

A possible solution to this equation satisfying it up to order $O(\mu^2)$ is given by [17]:

$$\zeta' = \bar{\mathbf{u}}' + \frac{\varepsilon}{4}\bar{\mathbf{u}}'^2 + \frac{\mu}{6}\bar{\partial}_{x't'}\bar{\mathbf{u}}', \quad \bar{\mathbf{u}}' = u + \frac{\mu}{12}x'\partial_{x'}u.$$

Here u solves the Benjamin-Bona-Mahony equation [2], which is a weakly nonlinear equation ($\varepsilon \sim O(\sqrt{\mu})$). It is shown in [41] that by taking $\tau = \varepsilon t', \eta = x' - t', u = v(\varepsilon t', x' - t'), S = \varepsilon/\mu$, one can obtain a unidirectional wave equation, which is known as the KdV equation:

$$\partial_\tau v + \frac{3}{2}v\partial_\eta v + \frac{1}{6S}\partial_\eta^3 v = 0 \tag{2.57}$$

We will solve this equation using an implicit hybridized DG method in the next chapter.

Chapter 3

A Hybridized Discontinuous Galerkin Technique for the KdV Equation[1]

In this chapter we introduce a hybridized discontinuous Galerkin (HDG) method for solving nonlinear Korteweg-de Vries (KdV) type equations. Similar to a standard HDG implementation, we first express the approximate variables and numerical fluxes inside each element in terms of the approximate traces of the scalar variable (u), and its first derivative ($\partial_x u$). These traces are assumed to be single-valued on each face. Next, we impose the conservation of numerical fluxes via two extra sets of equations. Using these global flux conservation conditions and applying the Newton-Raphson method, we construct a system of equations that can be solely expressed in terms of the increments of approximate traces in each iteration. Afterwards, we solve these equations, and substitute the approximate traces back into local equations over each element to obtain local approximate solutions. As for the time stepping scheme, we use the backward difference formulae. The method is proved to be stable

[1]This chapter is based on the article by Ali Samii, Nishant Panda, Craig Michoski and Clint Dawson, entitled: "A hybridized discontinuous Galerkin method for the nonlinear Korteweg–de Vries equation" [62]. Samii prepared the computer code for this article and wrote most of the manuscript except its introduction. Panda helped with implementing the numerical approach, Michoski wrote the introduction of the article, and Dawson provided mathematical proofs.

for a proper choice of stabilization parameters. Through numerical examples, we observe that for a mesh with kth order elements, the computed variable and its first and second derivatives show optimal convergence at order $k+1$ in both linear and nonlinear cases, which improves upon previously employed techniques.

3.1 Problem Statement and Space Discretization

We explained the derivation process of KdV equation in the previous chapter. Here, we write this equation in a more general form:

$$\partial_t u + \partial_x(\alpha_1 u^2 + \alpha_2 \partial_x^2 u) = f(x,t), \quad x \in \Omega \subset \mathbb{R},\, t \in (0,T], \tag{3.1}$$

with $\Omega \coloneqq [x_L, x_R]$, and the following initial and boundary data:

$$\begin{aligned} & u = u_0 && \text{in } \Omega \text{ for } t = 0, \\ \text{either of } & \begin{cases} u = g_u & \text{on } x_L, \text{ and } x_R, \\ u = g_u & \text{on } x_L, \text{ and } \partial_x^2 u\, \mathsf{n} = g_p \quad \text{on } x_R, \\ \partial_x^2 u\, \mathsf{n} = g_p & \text{on } x_L, \text{ and } u = g_u \quad \text{on } x_R, \end{cases} \\ & \partial_x u = g_q && \text{on } x_L \text{ or } x_R. \end{aligned} \tag{3.2}$$

Here u represents the wave amplitude, and n is the outward unit normal on the corresponding face (c.f. Fig. 3.1). Since, we are working in a one-dimensional setup, we look at n as a scalar, which is equal to ± 1 on x_L and x_R. α_2 is also equal to ± 1, and signifies the wave propagation direction. Moreover, we use α_1 to switch between a linear problem, where $\alpha_1 = 0$, and the nonlinear case with $\alpha_1 = 3$. When we take $\alpha_2 = 1$, the boundary condition on $\partial_x u$ should be

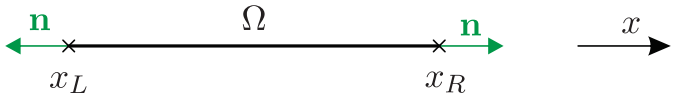


Figure 3.1: The domain of the problem and its left and right boundary.

applied on x_R, and when $\alpha_2 = -1$, the boundary condition should be applied on x_L. The well-definedness of the above problem has been studied in detail in [34], and it is known that the above set of boundary conditions results in a well-posed initial-boundary value problem for the KdV equation.

Next, we introduce the mixed forms $q = \partial_x u$ and $p = q_x$, and form the first order system of equations corresponding to (3.1):

$$\begin{aligned} &\partial_t u + \partial_x(\alpha_1 u^2 + \alpha_2 p) = f(x,t), \\ &\qquad p - \partial_x q = 0, \qquad\qquad x \in \Omega,\ t \in (0,T], \\ &\qquad q - \partial_x u = 0, \end{aligned} \tag{3.3}$$

with initial and boundary conditions:

$$\begin{aligned} &u = u_0 \qquad\quad \text{in } \Omega \text{ for } t = 0, \\ &\text{either of } \begin{cases} u = g_u & \text{on } x_L, \text{ and } x_R, \\ u = g_u & \text{on } x_L, \text{ and } p\,\mathsf{n} = g_p \quad \text{on } x_R, \\ p\,\mathsf{n} = g_p & \text{on } x_L, \text{ and } u = g_u \quad \text{on } x_R, \end{cases} \\ &q = g_q \qquad\quad \text{on } x_L \text{ or } x_R. \end{aligned} \tag{3.4}$$

For the purposes of analyzing the stability of the method, we will also consider periodic boundary conditions in place of (3.4).

3.1.1 Mesh Notation

We will partition $\Omega \subset \mathbb{R}$, by a finite collection of disjoint elements $\mathcal{T}_h := \{K_j\}$. The domain of each element K_j is considered to be: $K_j = [x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}]$. Since, we will work on a 1D domain, the left and right faces of K_j are each comprised of just one point. However, to maintain the generality, we use $\partial\mathcal{T}_h$ to denote the collection of the faces of all of the elements, i.e. $\partial\mathcal{T}_h = \{\partial K : K \in \mathcal{T}_h\}$. Let us denote by $\mathcal{E}_h^0$ the set of interior faces and $\mathcal{E}_h^\partial$ the set of boundary faces; meanwhile $\mathcal{E}_h = \mathcal{E}_h^\partial \cup \mathcal{E}_h^0$.

For any two neighboring elements K^+ and K^-, with nonempty $\partial K^+ \cap \partial K^-$, we will assign n^+ and n^- the outward pointing normals of ∂K^+ and ∂K^- respectively. The values of (u, q, p) on the common face of these elements will be denoted by $u^\pm, q^\pm, p^\pm$. We also denote $u^\pm, q^\pm, p^\pm$ on $x_{j\mp\frac{1}{2}}$, with $u^\pm_{j\mp\frac{1}{2}}, q^\pm_{j\mp\frac{1}{2}}, p^\pm_{j\mp\frac{1}{2}}$. For instance, $u^-_{j+\frac{1}{2}}$ means the value of u on the left side of a face located at $x_{j+\frac{1}{2}}$. Hence, $\mathsf{n}^-_{j+\frac{1}{2}} = +1$ and $\mathsf{n}^+_{j-\frac{1}{2}} = -1$, for all j. The mean $\{\!\{\cdot\}\!\}$ and jump $[\![\cdot]\!]$ of the information v on a given face $e \in \mathcal{E}_h^0$ are defined as:

$$\{\!\{v\}\!\} = (v^+ + v^-)/2, \quad \text{and} \quad [\![v\,\mathsf{n}]\!] = v^+\,\mathsf{n}^+ + u^-\,\mathsf{n}^-.$$

For boundary faces in $\mathcal{E}_h^\partial$ where the information (v) is single valued, the mean and jump are defined as:

$$\{\!\{v\}\!\} = v, \quad \text{and} \quad [\![v\,\mathsf{n}]\!] = v\,\mathsf{n}.$$

Furthermore, the boundary faces with available boundary data on u, q, and p will be denoted by Γ_u, Γ_q, and Γ_p respectively. It is worthwhile to mention $\mathcal{E}_h^\partial = \Gamma_u \cup \Gamma_p$.

3.1.2 Approximation Spaces

Let $\mathcal{P}^k(G)$ be the set of polynomials of degree at most k on the domain G. The discontinuous finite element spaces we use are

$$W_h^k = \{w \in L^2(\Omega) : w|_K \in \mathcal{P}^k(K),\ \forall K \in \mathcal{T}_h\}.$$

The trace finite element space (or skeleton space) is defined by:

$$M_h^k = \{\mu \in L^2(\mathcal{E}_h) : \mu|_e \in \mathcal{P}^k(e),\ \forall e \in \mathcal{E}_h\}.$$

We also characterize the following spaces, with the built-in boundary conditions:

$$M_h^k(\ell) = \{\mu \in M_h^k : \mu = \Pi\ell \ \text{ on } \Gamma_u\}, \qquad \bar{M}_h^k(\ell) = \{\mu \in M_h^k : \mu = \Pi\ell \ \text{ on } \Gamma_q\},$$

with Π being the L^2 projection into the skeleton space restricted to the boundary.

For the scalar product of functions v and w we will use the convention $(v,w)_G = \int_G vw\,\mathrm{d}x$, for $G \subset \Omega$. Moreover, $\langle v, w\rangle_{\partial K_j}$ which is commonly denoting the integration on the faces of $K_j \in \mathcal{T}_h$, may be simply written as $v^-_{j+\frac{1}{2}} w^-_{j+\frac{1}{2}} + v^+_{j-\frac{1}{2}} w^+_{j-\frac{1}{2}}$. Nevertheless, we might use any of these notations to keep the expressions concise and clear.

When we sum inner products over the entire mesh we use the notation:

$$\begin{aligned}
(v,w)_{\mathcal{T}_h} &= \sum_{K\in\mathcal{T}_h} (v,w)_K,\\
\langle \zeta, \rho\rangle_{\partial\mathcal{T}_h} &= \sum_{K\in\mathcal{T}_h} \langle \zeta, \rho\rangle_{\partial K},\\
\langle \mu, \omega\rangle_{\mathcal{E}_h} &= \sum_{e\in\mathcal{E}_h} \langle \mu, \omega\rangle_e,
\end{aligned}$$

where v, w are defined on $\mathcal{T}_h$, ζ, ρ are defined on $\partial\mathcal{T}_h$, and μ, ω are defined on $\mathcal{E}_h$.

3.2 Solution Method

Since the solution method for the nonlinear equation is closely related to that of the linear problem, and the latter can be explained more clearly, we will first look at the technique for the linear case. Without loss of generality, we take $\alpha_2 = -1$ in (3.3).

3.2.1 Linear Problem Solver

Considering eq. (3.3) with $\alpha_2 = -1$ and $\alpha_1 = 0$, we want to find the piecewise polynomial solutions $u, q, p \in W_h^k$, such that for all test functions $v, w, z \in W_h^k$,

$$\begin{aligned}
(\partial_t u, v)_K + (p, \partial_x v)_K - \langle \overset{*}{p}\,\mathsf{n}, v\rangle_{\partial K} &= (f, v)_K, \\
(p, w)_K + (q, \partial_x w)_K - \langle \overset{*}{q}\,\mathsf{n}, w\rangle_{\partial K} &= 0, \\
(q, z)_K + (u, \partial_x z)_K - \langle \hat{u}, z\,\mathsf{n}\rangle_{\partial K} &= 0,
\end{aligned} \tag{3.5}$$

for all $K \in \mathcal{T}_h$. Here $\overset{*}{p}$, $\overset{*}{q}$, are *numerical fluxes* and $\hat{u}$ is the *numerical trace* on ∂K. Similar to other numerical methods, numerical fluxes are approximations to p, q, and we choose them in a way to result in a stable and accurate method. On the other hand, in our hybrid method, we keep the numerical trace $\hat{u}$ as a new unknown on the skeleton space. We take $\hat{u}$ from $M_h^k(g_u)$, which means $\hat{u}$ is single valued on $\mathcal{E}_h$ by construction. To ensure the conservativeness of the method, we require that the normal components of $\overset{*}{q}$ and $\overset{*}{p}$ be continuous

across element edges. This continuity in our 1D problem means that these fluxes should be single-valued on each face. In regular discontinuous Galerkin methods, we can apply this single-valuedness by using the same flux on each face for the two elements connected to that face. In our hybrid technique we maintain the conservation of the flux via extra sets of equations. As a first step, we define $\overset{*}{q}$, and $\overset{*}{p}$ in the following forms:

$$\begin{aligned} \overset{*}{q} &= \hat{q} + \sigma(q - \hat{q})\,\mathsf{n}, \\ \overset{*}{p} &= p + \tau(u - \hat{u})\,\mathsf{n}. \end{aligned} \tag{3.6}$$

Here, we have introduced a new numerical trace $\hat{q} \in \bar{M}_h^k(g_q)$ and expressed the flux $\overset{*}{q}$ in terms of this trace. Similar to $\hat{u}$, we are going to keep $\hat{q}$ as a global unknown in the equations. Meanwhile, $\overset{*}{p}$ is also defined in terms of u, $\hat{u}$, p, which are among the current unknowns of the problem. Moreover, σ and τ are *stabilization parameters*. We will obtain the required condition for these parameters to make the method stable in the next section.

Next, we want to include the boundary data g_u and g_q into our solution. These boundary data are included by defining $\hat{u}$ and $\hat{q}$ on Γ_u and Γ_q, respectively. Hence, we set:

$$\hat{u} = \begin{cases} g_u, & \text{on } \partial K \cap \Gamma_u, \\ \lambda, & \text{on } \partial K \backslash \Gamma_u. \end{cases} \qquad \hat{q} = \begin{cases} g_q, & \text{on } \partial K \cap \Gamma_q, \\ \psi, & \text{on } \partial K \backslash \Gamma_q. \end{cases} \tag{3.7}$$

With $(\lambda, \phi) \in M_h^k(0) \times \bar{M}_h^k(0)$. In other words, on the faces where we have boundary data on u (Γ_u), we exclude $\hat{u}$ from our set of unknowns. On other faces, we substitute $\hat{u}$ with λ. We also eliminate $\hat{q}$ on Γ_q, and substitute it with ψ on all other faces.

So far, we have three equations (3.5), in the domain of each element. These three equations will be used to compute the internal unknowns: u, p, q. Solving these three equations in each element for u, p, q forms our *local problem*. In other words, for a given element $K \in \mathcal{T}_h$, we assume $\hat{u}, \overset{*}{p}, \overset{*}{q}$ are known on ∂K, and we want to solve (3.5) for u, p, q. Since, the fluxes $\overset{*}{q}, \overset{*}{p}$ are defined through numerical traces $\hat{u}, \hat{q}$, we can solve the local problem, provided that $\hat{u}, \hat{q}$ are known. Unlike, u, p, q, the traces $\hat{u}, \hat{q}$ are global unknowns. In order to find them, we need two extra global equations. These global equations are obtained by enforcing the conservation of the numerical fluxes on the element edges. Hence, we require that, on a given face $e \in \mathcal{E}_h$:

$$\left[\!\left[\overset{*}{q}\, \mathsf{n} \right]\!\right] = \begin{cases} q\, \mathsf{n}, & e \in \mathcal{E}_h^{\partial} \backslash \Gamma_q, \\ 0, & e \in \mathcal{E}_h^0. \end{cases} \qquad \left[\!\left[\overset{*}{p}\, \mathsf{n} \right]\!\right] = \begin{cases} g_p, & e \in \Gamma_p, \\ 0, & e \in \mathcal{E}_h^0. \end{cases} \tag{3.8}$$

It should be noted that, by setting $\left[\!\left[\overset{*}{q}\, \mathsf{n} \right]\!\right] = q\, \mathsf{n}$ we are not applying any boundary condition on $\hat{q}$. Instead, we want to emphasize that on the outflow face, where we have no boundary data on $\hat{q}$, the normal component of $\overset{*}{q}$ should be equal to the normal component of q from the upwind element. For the case of $\alpha_2 = -1$, $\mathcal{E}_h^{\partial} \backslash \Gamma_q$ in the above relation is equivalent to $x = x_R$. Since on x_R, $\overset{*}{q}\, \mathsf{n}$ is single-valued, we set its value equal to $q\, \mathsf{n}$ from the only contributing element. Meanwhile, by applying (3.8) on interior faces, we make sure that the fluxes on all of the element edges are conserved.

Before we continue to the final formulation, let us review our unknowns and the equations we use to solve them. We have three unknowns in the domain of each element $K \in \mathcal{T}_h$, i.e. u, p, q. Our local problem is solving (3.5)

for these internal unknowns, assuming that $\hat{u}, \hat{q}$ are known on ∂K. We also have two sets of global equations (3.8), which we use to compute $\hat{u}, \hat{q}$. These equations are the conservation of the flux across element edges. For interior faces, these global equations are simply $\left[\!\left[\overset{*}{q}\, \mathsf{n} \right]\!\right] = 0$ and $\left[\!\left[\overset{*}{p}\, \mathsf{n} \right]\!\right] = 0$. The boundary conditions on u, q are applied on $\hat{u}, \hat{q}$, through (3.7). The boundary condition on p is applied via $\left[\!\left[\overset{*}{p}\, \mathsf{n} \right]\!\right] = g_p$ on Γ_p. It should be noted that $\hat{u}$ is unknown on every face in $\mathcal{E}_h \backslash \Gamma_u$, and we have an equation for $\left[\!\left[\overset{*}{p}\, \mathsf{n} \right]\!\right]$ on every face in $\mathcal{E}_h^0 \cup \Gamma_p$. Since, $\mathcal{E}_h \backslash \Gamma_u = \mathcal{E}_h^0 \cup \Gamma_p$ the number of unknown $\hat{u}$ is equal to the number of equations on $\left[\!\left[\overset{*}{p}\, \mathsf{n} \right]\!\right]$. Similarly, one can see that, the number of unknown $\hat{q}$ is equal to the number of equations on $\left[\!\left[\overset{*}{q}\, \mathsf{n} \right]\!\right]$. Since, we introduce $\hat{u}, \hat{q}$ as extra unknowns on the mesh skeleton, and compute them using two constraint equations, i.e. the flux continuity conditions, this method can be classified as a *hybrid method* [3, 25].

As a special case of the above discussion, one can apply periodic boundary conditions by setting $\hat{u}|_{x_R} = \hat{u}|_{x_L}$, and $\hat{q}|_{x_R} = \hat{q}|_{x_L}$. These two will guarantee that the numerical traces are the same at x_L and x_R. Also, in order to apply the flux conservation conditions on the two ends of the domain, we set $\overset{*}{q}\,|_{x_R} = \overset{*}{q}\,|_{x_L}$, and $\overset{*}{p}\,|_{x_R} = \overset{*}{p}\,|_{x_L}$. These two conditions are actually obtained by assuming all faces are interior faces in (3.8).

Ultimately, we want to find $u, q, p \in W_h^k$, and traces $(\lambda, \psi) \in M_h^k(0) \times \bar{M}_h^k(0)$, such that $\forall v, w, z \in W_h^k$, (3.5), and (3.8) are satisfied. In this process we will apply the boundary conditions (3.7) and the flux definitions (3.6). Before looking at the implementation, we substitute the fluxes from (3.6) into

equation (3.5). Hence, for all $K \in \mathcal{T}_h$:

$$\begin{aligned}
(\partial_t u, v)_K + (p, \partial_x v)_K - \langle p\,\mathsf{n}, v\rangle_{\partial K} - \langle \tau u, v\rangle_{\partial K} + \langle \tau \hat{u}, v\rangle_{\partial K} &= (f, v)_K,\\
(p, w)_K + (q, \partial_x w)_K - \langle \sigma q, w\rangle_{\partial K} - \langle (\mathsf{n} - \sigma)\hat{q}, w\rangle_{\partial K} &= 0,\\
(q, z)_K + (u, \partial_x z)_K - \langle \hat{u}, z\,\mathsf{n}\rangle_{\partial K} &= 0.
\end{aligned} \tag{3.9}$$

As mentioned before, for a given $K \in \mathcal{T}_h$, we will use these equations to obtain u, q, p, assuming that $\hat{u}$ and $\hat{q}$ are known on ∂K.

By inserting the boundary data (3.7) into these equations, one gets:

$$\begin{aligned}
(\partial_t u, v)_K - (\partial_x p, v)_K - \langle \tau\, u, v\rangle_{\partial K} & \\
+ \langle \tau\, \lambda, v\rangle_{\partial K} &= (f, v)_K - \langle \tau\, g_u, v\rangle_{\partial K\cap\Gamma_u},\\
(p, w)_K + (q, \partial_x w)_K - \langle \sigma q, w\rangle_{\partial K} & \\
- \langle (\mathsf{n} - \sigma)\psi, w\rangle_{\partial K} &= \langle (\mathsf{n} - \sigma) g_q, w\rangle_{\partial K\cap\Gamma_q},\\
(q, z)_K + (u, z_x)_K - \langle \lambda, z\,\mathsf{n}\rangle_{\partial K} &= \langle g_u, z\,\mathsf{n}\rangle_{\partial K\cap\Gamma_u}.
\end{aligned} \tag{3.10}$$

Also substitute (3.6) into (3.8) to obtain the following global equations:

$$\begin{aligned}
\langle \hat{q}\,\mathsf{n} + \sigma(q - \hat{q}), \mu\rangle_{\partial\mathcal{T}_h} &= 0,\\
\langle p\,\mathsf{n} + \tau(u - \hat{u}), \eta\rangle_{\partial\mathcal{T}_h} &= \langle g_p, \eta\rangle_{\partial\mathcal{T}_h\cap\Gamma_p},
\end{aligned} \tag{3.11}$$

for all $(\mu, \eta) \in \bar{M}_h^k(0) \times M_h^k(0)$.

Next, we want to solve the system of equations (3.10) and (3.11) by the hybridized DG technique.

3.2.1.1 Implementation

Let us assemble the local equations (3.10) and write them in terms of the following bilinear operators:

$$\begin{aligned} \mathscr{a}(\partial_t u, v) - \mathscr{b}^{\mathrm{T}}(p, v) + \mathscr{c}_1(\lambda, v) - \mathscr{d}_1(u, v) &= \mathscr{f}(v) - \mathscr{g}(v), \\ \mathscr{a}(p, w) + \mathscr{b}(q, w) - \mathscr{c}_2(\psi, w) - \mathscr{d}_2(q, w) &= \mathscr{h}(w), \\ \mathscr{a}(q, z) + \mathscr{b}(u, z) - \mathscr{c}_3(\lambda, z) &= \mathscr{k}(z), \end{aligned} \tag{3.12}$$

and for the global equation (3.11):

$$\begin{aligned} \mathscr{c}_4^{\mathrm{T}}(q, \mu) + \mathscr{e}_1(\psi, \mu) &= 0, \\ \mathscr{c}_5^{\mathrm{T}}(u, \eta) + \mathscr{c}_6^{\mathrm{T}}(p, \eta) - \mathscr{e}_2(\lambda, \eta) &= \mathscr{s}(\eta), \end{aligned} \tag{3.13}$$

with the following definitions:

$$\begin{aligned} \mathscr{a}(u, v) &= (u, v)_{\mathcal{T}_h}, \quad \mathscr{b}(q, w) = (q, \partial_x w)_{\mathcal{T}_h}, \\ \mathscr{d}_1(u, v) &= \langle \tau u, v \rangle_{\partial \mathcal{T}_h}, \quad \mathscr{d}_2(q, w) = \langle \sigma q, w \rangle_{\partial \mathcal{T}_h}, \\ \mathscr{c}_1(\lambda, v) &= \langle \tau \lambda, v \rangle_{\partial \mathcal{T}_h}, \quad \mathscr{c}_2(\psi, w) = \langle (\mathsf{n} - \sigma)\psi, w \rangle_{\partial \mathcal{T}_h}, \\ \mathscr{c}_3(\lambda, z) &= \langle \lambda, z\,\mathsf{n} \rangle_{\partial \mathcal{T}_h}, \quad \mathscr{c}_4^{\mathrm{T}}(q, \mu) = \langle \sigma q, \mu \rangle_{\partial \mathcal{T}_h} - \langle q\,\mathsf{n}, \mu \rangle_{\partial \mathcal{T}_h \setminus \Gamma_q}, \\ \mathscr{c}_5^{\mathrm{T}}(u, \eta) &= \langle \tau u, \eta \rangle_{\partial \mathcal{T}_h}, \quad \mathscr{c}_6^{\mathrm{T}}(p, \eta) = \langle p\,\mathsf{n}, \eta \rangle_{\partial \mathcal{T}_h}, \\ \mathscr{e}_1(\psi, \mu) &= \langle (\mathsf{n} - \sigma)\psi, \mu \rangle_{\partial \mathcal{T}_h}, \quad \mathscr{e}_2(\lambda, \mu) = \langle \tau \lambda, \mu \rangle_{\partial \mathcal{T}_h}, \\ \mathscr{f}(v) &= (f, v)_{\mathcal{T}_h}, \quad \mathscr{g}(v) = \langle \tau\, g_u, v \rangle_{\partial \mathcal{T}_h \cap \Gamma_u}, \\ \mathscr{h}(w) &= \langle (\mathsf{n} - \sigma) g_q, w \rangle_{\partial \mathcal{T}_h \cap \Gamma_q}, \quad \mathscr{k}(z) = \langle g_u, z\,\mathsf{n} \rangle_{\partial \mathcal{T}_h \cap \Gamma_u}, \\ \mathscr{s}(\eta) &= \langle g_p, \eta \rangle_{\partial \mathcal{T}_h \cap \Gamma_p}, \end{aligned} \tag{3.14}$$

for all $v, w, z \in W_h^k$, and $(\mu, \eta) \in \bar{M}_h^k(0) \times M_h^k(0)$.

Next, we write (3.12) and (3.13) in the discretized form. As for the time integration scheme, we choose a backward Euler approach with time-step Δt^n at time-level t^n. One may appropriately use higher order BDF or an implicit Runge-Kutta method. As a result, the corresponding matrix equations at time t^n would become:

$$\frac{1}{\Delta t}AU - B^{\mathrm{T}}P + C_1\Lambda - D_1U = F - G + \frac{1}{\Delta t}AU^{n-1}, \tag{3.15a}$$

$$AP + BQ - C_2\Psi - D_2Q = H, \tag{3.15b}$$

$$AQ + BU - C_3\Lambda = K, \tag{3.15c}$$

$$C_4^{\mathrm{T}}Q + E_1\Psi = 0, \tag{3.15d}$$

$$C_5^{\mathrm{T}}U + C_6^{\mathrm{T}}P - E_2\Lambda = S, \tag{3.15e}$$

where U^{n-1} stands for U from the previous time-level, and all other variables are calculated at the current time-level.

As mentioned before, we are not going to assemble eqs. (3.15a-c) and solve them globally. We will apply a process of *condensation* on the internal unknowns U, P and Q and express them in terms of the trace unknowns Λ and Ψ. Then we solve global equations (3.15d,e) for Λ and Ψ. To this end, we do a local solve on (3.15c) and obtain Q in terms of the other unknowns, and the supported boundary data:

$$Q = A^{-1}(K - BU + C_3\Lambda). \tag{3.16a}$$

Then, we substitute Q from the above relation into (3.15b), to obtain an

expression for P in terms of U, Λ, Ψ, and the boundary information:

$$\begin{aligned} P =& A^{-1}(B-D_2)A^{-1}BU - A^{-1}(B-D_2)A^{-1}C_3\Lambda + A^{-1}C_2\Psi \\ &+ A^{-1}H - A^{-1}(B-D_2)A^{-1}K. \end{aligned} \tag{3.16b}$$

Finally, we put P from the above relation into (3.15a) to obtain U in terms of Λ, Ψ and the boundary data:

$$\begin{aligned} &\left[\frac{1}{\Delta t}A - D_1 - B^{\mathrm{T}}A^{-1}(B-D_2)A^{-1}B\right]U \\ &\quad = \frac{1}{\Delta t}AU^{n-1} + \left[-C_1 - B^{\mathrm{T}}A^{-1}(B-D_2)A^{-1}C_3\right]\Lambda + B^{\mathrm{T}}A^{-1}C_2\Psi \\ &\qquad - B^{\mathrm{T}}A^{-1}(B-D_2)A^{-1}K + B^{\mathrm{T}}A^{-1}H + F - G. \end{aligned} \tag{3.16c}$$

The solution procedure to implement this technique, can be summarized in three steps:

1. Obtain U in the local equation (3.16c) in terms of Λ and Ψ, and use this U to obtain Q in terms of Λ and Ψ via (3.16a); also obtain P in terms of Λ and Ψ via (3.16b).

2. Assemble the U, Q, and P from the previous step for each element, along with Λ and Ψ into the global equations (3.15-d,e), to form the global matrix equation and solve it for Λ and Ψ.

3. Use the globally solved Λ and Ψ from the previous step, to solve the local equations (3.16c,a,b) for U, Q, and P. As explained in the first step, one starts with (3.16c) to compute U, then use this U in (3.16a) and (3.16b) to obtain Q and P.

In this scheme, the first and third steps are local on each element and can be done in parallel. The only global solve step is the second step. Moreover, the number of skeleton unknowns in these global equations are $\mathcal{O}(k^{d-1}/h)$, compared to the internal unknowns which are $\mathcal{O}(k^d/h)$. Hence, we can expect an improved performance from the proposed hybridized scheme.

3.2.1.2 Stability of the Method

In this section we prove the stability of the proposed method in the continuous time case. We first look at the simplest case of periodic boundary conditions. Then, we discuss the stability for other types of boundary conditions.

Theorem 3.2.1. *If the stabilization parameters in* (3.6) *satisfy:* $\sigma \neq 0$, $\sigma > \frac{1}{2}\mathsf{n}$, *and* $\tau < 0$, *then the proposed method with periodic boundary conditions is stable and the solution to* (3.5) *exists and is unique.*

Proof. We consider (3.5), with the zero source term and expand the boundary terms to obtain:

$$(\partial_t u, v)_{K_j} + (p, \partial_x v)_{K_j} - \overset{*}{p}{}^{-}_{j+\frac{1}{2}} v^{-}_{j+\frac{1}{2}} + \overset{*}{p}{}^{+}_{j-\frac{1}{2}} v^{+}_{j-\frac{1}{2}} = 0, \tag{3.17a}$$

$$(p, w)_{K_j} + (q, \partial_x w)_{K_j} - \overset{*}{q}{}^{-}_{j+\frac{1}{2}} w^{-}_{j+\frac{1}{2}} + \overset{*}{q}{}^{+}_{j-\frac{1}{2}} w^{+}_{j-\frac{1}{2}} = 0, \tag{3.17b}$$

$$(q, z)_{K_j} + (u, \partial_x z)_{K_j} - \hat{u}_{j+\frac{1}{2}} z^{-}_{j+\frac{1}{2}} + \hat{u}_{j-\frac{1}{2}} z^{+}_{j-\frac{1}{2}} = 0. \tag{3.17c}$$

Setting $v = u$, $w = -q$, $z = p$, would yield:

$$\begin{aligned}
(\partial_t u, u)_{K_j} + (p, \partial_x u)_{K_j} - \overset{*}{p}{}^{-}_{j+\frac{1}{2}} u^{-}_{j+\frac{1}{2}} + \overset{*}{p}{}^{+}_{j-\frac{1}{2}} u^{+}_{j-\frac{1}{2}} = 0, \\
-(p, q)_{K_j} - (q, \partial_x q)_{K_j} + \overset{*}{q}{}^{-}_{j+\frac{1}{2}} q^{-}_{j+\frac{1}{2}} - \overset{*}{q}{}^{+}_{j-\frac{1}{2}} q^{+}_{j-\frac{1}{2}} = 0, \\
(q, p)_{K_j} + (u, \partial_x p)_{K_j} - \hat{u}_{j+\frac{1}{2}} p^{-}_{j+\frac{1}{2}} + \hat{u}_{j-\frac{1}{2}} p^{+}_{j-\frac{1}{2}} = 0.
\end{aligned}$$

Then we add these equations together:

$$\begin{aligned}
(\partial_t u, u)_{K_j} + \int_{K_j} \partial_x (pu) \,\mathrm{d}x - \frac{1}{2} \int_{K_j} \partial_x (q^2) \,\mathrm{d}x \\
- \overset{*}{p}{}^{-}_{j+\frac{1}{2}} u^{-}_{j+\frac{1}{2}} + \overset{*}{p}{}^{+}_{j-\frac{1}{2}} u^{+}_{j-\frac{1}{2}} + \overset{*}{q}{}^{-}_{j+\frac{1}{2}} q^{-}_{j+\frac{1}{2}} - \overset{*}{q}{}^{+}_{j-\frac{1}{2}} q^{+}_{j-\frac{1}{2}} \\
- \hat{u}_{j+\frac{1}{2}} p^{-}_{j+\frac{1}{2}} + \hat{u}_{j-\frac{1}{2}} p^{+}_{j-\frac{1}{2}} = 0.
\end{aligned}$$

Which may be written as:

$$\begin{aligned}
\frac{1}{2} \partial_t (u, u)_{K_j} + p^{-}_{j+\frac{1}{2}} u^{-}_{j+\frac{1}{2}} - p^{+}_{j-\frac{1}{2}} u^{+}_{j-\frac{1}{2}} - \frac{1}{2} \left[(q^2)^{-}_{j+\frac{1}{2}} - (q^2)^{+}_{j-\frac{1}{2}} \right] \\
- \overset{*}{p}{}^{-}_{j+\frac{1}{2}} u^{-}_{j+\frac{1}{2}} + \overset{*}{p}{}^{+}_{j-\frac{1}{2}} u^{+}_{j-\frac{1}{2}} + \overset{*}{q}{}^{-}_{j+\frac{1}{2}} q^{-}_{j+\frac{1}{2}} - \overset{*}{q}{}^{+}_{j-\frac{1}{2}} q^{+}_{j-\frac{1}{2}} \\
- \hat{u}_{j+\frac{1}{2}} p^{-}_{j+\frac{1}{2}} + \hat{u}_{j-\frac{1}{2}} p^{+}_{j-\frac{1}{2}} = 0.
\end{aligned}$$

By reordering the terms, we get:

$$\frac{1}{2} \frac{\partial}{\partial t} \|u\|^2_{K_j} + \Theta^q_{K_j} + \Theta^p_{K_j} = 0, \tag{3.18}$$

with

$$\begin{aligned}
\Theta^q_{K_j} &= \left[\left(\overset{*}{q}{}^{-}_{j+\frac{1}{2}} - \tfrac{1}{2} q^{-}_{j+\frac{1}{2}} \right) q^{-}_{j+\frac{1}{2}} - \left(\overset{*}{q}{}^{+}_{j-\frac{1}{2}} - \tfrac{1}{2} q^{+}_{j-\frac{1}{2}} \right) q^{+}_{j-\frac{1}{2}} \right], \\
\Theta^p_{K_j} &= \left[\left(p^{-}_{j+\frac{1}{2}} - \overset{*}{p}{}^{-}_{j+\frac{1}{2}} \right) u^{-}_{j+\frac{1}{2}} + \left(\overset{*}{p}{}^{+}_{j-\frac{1}{2}} - p^{+}_{j-\frac{1}{2}} \right) u^{+}_{j-\frac{1}{2}} - \hat{u}_{j+\frac{1}{2}} p^{-}_{j+\frac{1}{2}} + \hat{u}_{j-\frac{1}{2}} p^{+}_{j-\frac{1}{2}} \right].
\end{aligned}$$

Now, let us rewrite $\overset{*}{p}$ from (3.6), as below:

$$\begin{aligned}
\overset{*}{p}{}^{-}_{j+\frac{1}{2}} &= p^{-}_{j+\frac{1}{2}} + \tau^{-}_{j+\frac{1}{2}}(u^{-}_{j+\frac{1}{2}} - \hat{u}_{j+\frac{1}{2}})\mathsf{n}^{-}_{j+\frac{1}{2}} = p^{-}_{j+\frac{1}{2}} + \tau^{-}_{j+\frac{1}{2}}(u^{-}_{j+\frac{1}{2}} - \hat{u}_{j+\frac{1}{2}}),\\
\overset{*}{p}{}^{+}_{j-\frac{1}{2}} &= p^{+}_{j-\frac{1}{2}} + \tau^{+}_{j-\frac{1}{2}}(u^{+}_{j-\frac{1}{2}} - \hat{u}_{j-\frac{1}{2}})\mathsf{n}^{+}_{j-\frac{1}{2}} = p^{+}_{j-\frac{1}{2}} - \tau^{+}_{j-\frac{1}{2}}(u^{+}_{j-\frac{1}{2}} - \hat{u}_{j-\frac{1}{2}}).
\end{aligned} \tag{3.19}$$

By substituting $p^{-}_{j+\frac{1}{2}}$ and $p^{+}_{j+\frac{1}{2}}$ from (3.19) into $\Theta^{p}_{K_j}$, we have:

$$\begin{aligned}
\Theta^{p}_{K_j} =& -\tau^{-}_{j+\frac{1}{2}}(u^{-}_{j+\frac{1}{2}} - \hat{u}_{j+\frac{1}{2}})u^{-}_{j+\frac{1}{2}} - \tau^{+}_{j-\frac{1}{2}}(u^{+}_{j-\frac{1}{2}} - \hat{u}_{j-\frac{1}{2}})u^{+}_{j-\frac{1}{2}}\\
&- \hat{u}_{j+\frac{1}{2}}\left(\overset{*}{p}{}^{-}_{j+\frac{1}{2}} - \tau^{-}_{j+\frac{1}{2}}(u^{-}_{j+\frac{1}{2}} - \hat{u}_{j+\frac{1}{2}})\right)\\
&+ \hat{u}_{j-\frac{1}{2}}\left(\overset{*}{p}{}^{+}_{j-\frac{1}{2}} + \tau^{+}_{j-\frac{1}{2}}(u^{+}_{j-\frac{1}{2}} - \hat{u}_{j-\frac{1}{2}})\right)\\
=& -\tau^{-}_{j+\frac{1}{2}}\left((u^{-}_{j+\frac{1}{2}})^2 - \hat{u}_{j+\frac{1}{2}}u^{-}_{j+\frac{1}{2}}\right) - \tau^{+}_{j-\frac{1}{2}}\left((u^{+}_{j-\frac{1}{2}})^2 - \hat{u}_{j-\frac{1}{2}}u^{+}_{j-\frac{1}{2}}\right)\\
&- \hat{u}_{j+\frac{1}{2}}\overset{*}{p}{}^{-}_{j+\frac{1}{2}} + \tau^{-}_{j+\frac{1}{2}}\left(\hat{u}_{j+\frac{1}{2}}u^{-}_{j+\frac{1}{2}} - (\hat{u}_{j+\frac{1}{2}})^2\right)\\
&+ \hat{u}_{j-\frac{1}{2}}\overset{*}{p}{}^{+}_{j-\frac{1}{2}} + \tau^{+}_{j-\frac{1}{2}}\left(\hat{u}_{j-\frac{1}{2}}u^{+}_{j-\frac{1}{2}} - (\hat{u}_{j-\frac{1}{2}})^2\right).
\end{aligned} \tag{3.20}$$

Now, we sum over all elements, and apply the conservation of the flux. For the current 1D problem, the flux conservation condition, i.e. $\left[\!\left[\overset{*}{p}\,\mathsf{n}\right]\!\right] = 0$, simply becomes: $\overset{*}{p}{}^{+} = \overset{*}{p}{}^{-}$. Hence, we get:

$$\Theta^{p}_{\mathcal{T}_h} = \sum_{K_j\in\mathcal{T}_h}\Theta^{p}_{K_j} = \sum_{j} -\tau^{-}_{j+\frac{1}{2}}(u^{-}_{j+\frac{1}{2}} - \hat{u}_{j+\frac{1}{2}})^2 - \tau^{+}_{j-\frac{1}{2}}(u^{+}_{j-\frac{1}{2}} - \hat{u}_{j-\frac{1}{2}})^2. \tag{3.21}$$

which is nonnegative, for all $\tau^{\mp}_{j\pm\frac{1}{2}} < 0$, or simply $\tau < 0$.

Next, let us consider $\Theta^{q}_{K_j}$ in (3.18), and choose $\overset{*}{q}$ similar to (3.6):

$$\begin{aligned}
\overset{*}{q}{}^{+}_{j-\frac{1}{2}} &= \hat{q}_{j-\frac{1}{2}} + \sigma^{+}_{j-\frac{1}{2}}(q^{+}_{j-\frac{1}{2}} - \hat{q}_{j-\frac{1}{2}})\mathsf{n}^{+}_{j-\frac{1}{2}} = \hat{q}_{j-\frac{1}{2}} - \sigma^{+}_{j-\frac{1}{2}}(q^{+}_{j-\frac{1}{2}} - \hat{q}_{j-\frac{1}{2}}),\\
\overset{*}{q}{}^{-}_{j+\frac{1}{2}} &= \hat{q}_{j+\frac{1}{2}} + \sigma^{-}_{j+\frac{1}{2}}(q^{-}_{j+\frac{1}{2}} - \hat{q}_{j+\frac{1}{2}})\mathsf{n}^{-}_{j+\frac{1}{2}} = \hat{q}_{j+\frac{1}{2}} + \sigma^{-}_{j+\frac{1}{2}}(q^{-}_{j+\frac{1}{2}} - \hat{q}_{j+\frac{1}{2}}).
\end{aligned} \tag{3.22}$$

These fluxes should be used along with the flux conservation condition $[\![\overset{*}{q}\, \mathsf{n}]\!] = \overset{*}{q}^{+}\mathsf{n}^{+} + \overset{*}{q}^{-}\mathsf{n}^{-} = 0$. Assuming $\sigma \neq 0$, $q^{-}_{j+\frac{1}{2}}$ and $q^{+}_{j-\frac{1}{2}}$ may be written as:

$$q^{+}_{j-\frac{1}{2}} = \frac{(1+\sigma^{+}_{j-\frac{1}{2}})\hat{q}_{j-\frac{1}{2}} - \overset{*}{q}^{+}_{j-\frac{1}{2}}}{\sigma^{+}_{j-\frac{1}{2}}}, \qquad q^{-}_{j+\frac{1}{2}} = \frac{\overset{*}{q}^{-}_{j+\frac{1}{2}} - (1-\sigma^{-}_{j+\frac{1}{2}})\hat{q}_{j+\frac{1}{2}}}{\sigma^{-}_{j+\frac{1}{2}}}. \quad (3.23)$$

Next, we consider $\Theta^{q}_{K_j}$ from (3.18), and substitute the above $q^{\pm}$, to obtain:

$$\begin{aligned}
\Theta^{q}_{K_j} &= -\frac{(\overset{*}{q}^{-}_{j+\frac{1}{2}})^2 + (1-\sigma^{-}_{j+\frac{1}{2}})^2(\hat{q}_{j+\frac{1}{2}}) - 2(1-\sigma^{-}_{j+\frac{1}{2}})\overset{*}{q}^{-}_{j+\frac{1}{2}}\hat{q}_{j+\frac{1}{2}}}{2(\sigma^{-}_{j+\frac{1}{2}})^2} \\
&\quad + \frac{(\overset{*}{q}^{-}_{j+\frac{1}{2}})^2 - (1-\sigma^{-}_{j+\frac{1}{2}})\overset{*}{q}^{-}_{j+\frac{1}{2}}\hat{q}_{j+\frac{1}{2}}}{\sigma^{-}_{j+\frac{1}{2}}} - \frac{(1+\sigma^{+}_{j-\frac{1}{2}})\overset{*}{q}^{+}_{j-\frac{1}{2}}\hat{q}_{j-\frac{1}{2}} - (\overset{*}{q}^{+}_{j-\frac{1}{2}})^2}{\sigma^{+}_{j-\frac{1}{2}}} \\
&\quad + \frac{(\overset{*}{q}^{+}_{j-\frac{1}{2}})^2 + (1+\sigma^{+}_{j-\frac{1}{2}})^2(\hat{q}_{j-\frac{1}{2}})^2 - 2(1+\sigma^{+}_{j-\frac{1}{2}})\overset{*}{q}^{+}_{j-\frac{1}{2}}\hat{q}_{j-\frac{1}{2}}}{2(\sigma^{+}_{j-\frac{1}{2}})^2} \\
&= \frac{(2\sigma^{-}_{j+\frac{1}{2}}-1)}{2(\sigma^{-}_{j+\frac{1}{2}})^2}\left[(\overset{*}{q}^{-}_{j+\frac{1}{2}})^2 + (\hat{q}_{j+\frac{1}{2}})^2 - 2\overset{*}{q}^{-}_{j+\frac{1}{2}}\hat{q}_{j+\frac{1}{2}}\right] - \frac{1}{2}(\hat{q}_{j+\frac{1}{2}})^2 + \overset{*}{q}^{-}_{j+\frac{1}{2}}\hat{q}_{j+\frac{1}{2}} \\
&\quad + \frac{(2\sigma^{+}_{j-\frac{1}{2}}+1)}{2(\sigma^{+}_{j-\frac{1}{2}})^2}\left[\overset{*}{q}^{+}_{j-\frac{1}{2}} - \hat{q}_{j-\frac{1}{2}}\right]^2 + \frac{1}{2}(\hat{q}_{j-\frac{1}{2}})^2 - \overset{*}{q}^{+}_{j-\frac{1}{2}}\hat{q}_{j-\frac{1}{2}}. \qquad (3.24)
\end{aligned}$$

By summing over all elements, and applying the flux conservation and periodic boundary condition, we get:

$$\begin{aligned}
\Theta^{q}_{\mathcal{T}_h} &= \sum_{K_j \in \mathcal{T}_h} \Theta^{q}_{K_j} \\
&= \sum_{j} \frac{(2\sigma^{-}_{j+\frac{1}{2}}-1)}{2(\sigma^{-}_{j+\frac{1}{2}})^2}\left[\overset{*}{q}^{-}_{j+\frac{1}{2}} - \hat{q}_{j+\frac{1}{2}}\right]^2 + \frac{(2\sigma^{+}_{j-\frac{1}{2}}+1)}{2(\sigma^{+}_{j-\frac{1}{2}})^2}\left[\overset{*}{q}^{+}_{j-\frac{1}{2}} - \hat{q}_{j-\frac{1}{2}}\right]^2. \quad (3.25)
\end{aligned}$$

Which is non-negative for $\sigma^{-} > \frac{1}{2}$, and $\sigma^{+} > -\frac{1}{2}$, or simply $\sigma > \frac{1}{2}\mathsf{n}$, and also $\sigma \neq 0$.

Eventually, if we sum (3.18) over all elements, we get:

$$\frac{1}{2}\frac{\partial}{\partial t}\|u\|^2_{\mathcal{T}_h} + \Theta^q_{\mathcal{T}_h} + \Theta^p_{\mathcal{T}_h} = 0,$$

with $\Theta^q_{\mathcal{T}_h}$ and $\Theta^p_{\mathcal{T}_h}$ obtained in (3.21) and (3.25). According to the assumptions on σ and τ, these two are nonnegative. Hence, $\partial\|u\|^2_{\mathcal{T}_h}/\partial t \leq 0$, and the only solution to the problem with zero source term and zero initial condition is $u = 0$. By putting $u = 0$ in (3.21), and knowing that $\Theta^p_{\mathcal{T}_h} = 0$, one gets $\hat{u} = 0$. Next, set $z = q$ in (3.17c) and use $u = 0$ and $\hat{u} = 0$ to conclude $q = 0$. Also, we know $\Theta^q_{\mathcal{T}_h} = 0$, which implies $\overset{*}{q}^+ = \overset{*}{q}^- = \hat{q}$. Comparing this with the relationship of $\overset{*}{q}$ and $\hat{q}$ obtained from (3.23) and setting $q = 0$, one can see that $\hat{q} = \overset{*}{q} = 0$. Finally, set $w = p$ in (3.17b), and use $q = 0$ and $\overset{*}{q} = 0$ to deduce that $p = 0$. Consequently, the only solution to the problem with periodic boundary condition, zero initial condition and zero source term is the trivial solution.

Since we are working in a linear and finite dimensional setting, the trivial null-space implies existence and uniqueness of the solution. Therefore, the theorem follows. □

Corollary 3.2.2. *The proposed method with the boundary conditions $u = 0$ on x_R, x_L, and $q = 0$ on x_L, and the stabilization parameters: $\sigma > \frac{1}{2}\mathsf{n}$, $\tau < 0$, is stable and has a unique solution.*

Proof. The proof follows similar steps as Theorem 3.1. However, in (3.20) the two terms $-\hat{u}_{j+\frac{1}{2}}\overset{*}{p}{}^-_{j+\frac{1}{2}}$ and $\hat{u}_{j-\frac{1}{2}}\overset{*}{p}{}^+_{j-\frac{1}{2}}$ will not cancel at the boundaries of the

domain. Instead, by taking $g_u = 0$ both of these terms become zero. Moreover, in (3.24), by setting $g_q = 0$, one can get $\overset{*}{q}^{+}\hat{q} = (\hat{q})^2 = 0$ at $x = x_L$. On $x = x_R$, using (3.23) with $\overset{*}{q}^{-} = q^{-}$, results in $\overset{*}{q}^{-} = \hat{q}$. Hence, we get:

$$\frac{1}{2}\frac{\partial}{\partial t}\|u\|^2_{\mathfrak{T}_h} + \bar{\Theta}^q_{\mathfrak{T}_h} + \Theta^p_{\mathfrak{T}_h} = 0.$$

with, $\bar{\Theta}^q_{\mathfrak{T}_h} = \Theta^q_{\mathfrak{T}_h} + \frac{1}{2}(\hat{q}|_{x=x_R})^2$, which is non-negative. Therefore, based on the same logic as Theorem 3.1, the only solution to the problem with zero source term, zero initial condition, and zero boundary conditions is the trivial solution. Hence, we have stability. The existence and uniqueness will also follow. □

Corollary 3.2.3. *The proposed method with the boundary conditions $u = 0$ on x_L, $p = 0$ on x_R, and $q = 0$ on x_L, and the stabilization parameters: $\sigma > \frac{1}{2}\mathsf{n}$, $\tau < 0$, is stable and has a unique solution. The same is true for $u = 0$ on x_R, $p = 0$ on x_L, and $q = 0$ on x_L.*

Proof. Similar to Corollary 3.2, one can show that in (3.20) the two terms $-\hat{u}_{j+\frac{1}{2}}\overset{*}{p}^{-}_{j+\frac{1}{2}}$ and $\hat{u}_{j-\frac{1}{2}}\overset{*}{p}^{+}_{j-\frac{1}{2}}$ are zero for $g_u = g_p = 0$. Hence, for zero initial condition, zero boundary condition and zero source term, the only solution to the problem is the trivial solution. Therefore, we have stability, existence, and uniqueness of the solution. □

In the proof of Corollary 3.2.2, it is worthwhile noting that, supporting the boundary data for q on x_R instead of x_L can result in an unstable scheme. In that case, instead of $\bar{\Theta}^q_{\mathfrak{T}_h} = \Theta^q_{\mathfrak{T}_h} + \frac{1}{2}(\hat{q}|_{x=x_R})^2$, we get $\bar{\Theta}^q_{\mathfrak{T}_h} = \Theta^q_{\mathfrak{T}_h} - \frac{1}{2}(\hat{q}|_{x=x_L})^2$, which is not necessarily non-negative, and the stability cannot be inferred.

3.2.2 Nonlinear Solver

Let us consider (3.3), with $\alpha_2 = -1$ and $\alpha_1 = 3$. We want to find the approximations $u, q, p \in W_h^k$, such that for all test functions $v, w, z \in W_h^k$,

$$\begin{aligned}(\partial_t u, v)_K - (3u^2 - p, \partial_x v)_K + \langle \mathcal{H}\,\mathsf{n}, v\rangle_{\partial K} &= (f, v)_K,\\ (p, w)_K + (q, \partial_x w)_K - \langle \overset{*}{q}\,\mathsf{n}, w\rangle_{\partial K} &= 0,\\ (q, z)_K + (u, \partial_x z)_K - \langle \hat{u}, z\,\mathsf{n}\rangle_{\partial K} &= 0,\end{aligned} \tag{3.26}$$

for all $K \in \mathcal{T}_h$. Here, the numerical flux $\mathcal{H}$ is an approximation to $3u^2 - p$. In this nonlinear problem, we define $\mathcal{H}$ and $\overset{*}{q}$ as follows:

$$\begin{aligned}\overset{*}{q} &= \hat{q} + \sigma(q - \hat{q})\,\mathsf{n},\\ \mathcal{H} &= 3\hat{u}^2 - p + \tau(u - \hat{u})\,\mathsf{n}.\end{aligned} \tag{3.27}$$

In order to apply the boundary conditions on u, q, we use the same scheme as linear case, i.e. (3.7). Furthermore, we require our fluxes to be conserved across the element faces. This flux conservation will be enforced explicitly through a set of global equations, which can be written for a given face $e \in \mathcal{E}_h$:

$$\left[\!\left[\overset{*}{q}\,\mathsf{n} \right]\!\right] = \begin{cases} q\,\mathsf{n}, & e \in \mathcal{E}_h^{\partial}\backslash\Gamma_q,\\ 0, & e \in \mathcal{E}_h^0. \end{cases} \qquad \begin{cases} p\,\mathsf{n} = g_p, & e \in \Gamma_p,\\ [\![\, \mathcal{H}\,\mathsf{n} \,]\!] = 0, & e \in \mathcal{E}_h^0. \end{cases} \tag{3.28}$$

The difference of the above equations with (3.8) is the way that we apply the boundary condition on p. Since this boundary condition is applied through the numerical flux, and the flux is nonlinear, the boundary condition at Γ_p is applied via $p\,\mathsf{n} = g_p$. It should be noted that we use local equations to derive p in terms of $\hat{u}$ and $\hat{q}$; therefore, the corresponding global equation will be in terms of the numerical traces.

Using the flux conservation relations together with (3.26) we can form our nonlinear system of equations. Moreover, in order to discretize in time, we use backward Euler time-stepping scheme with time-step Δt at time-level t_n. Hence, we are looking for $u, q, p \in W_h^k$ and $\hat{u}, \hat{q} \in M_h^k(g_u) \times \bar{M}_h^k(g_q)$, such that:

$$\begin{aligned}
\frac{1}{\Delta t}(u,v)_K - (p_x,v)_K - (3u^2,\partial_x v)_K + \langle 3\hat{u}^2\,\mathsf{n}, v\rangle_{\partial K} \\
+\langle \tau u, v\rangle_{\partial K} - \langle \tau \hat{u}, v\rangle_{\partial K} = (f,v)_K + \frac{1}{\Delta t}(u^{n-1},v)_K, \\
(p,w)_K + (q,\partial_x w)_K - \langle \sigma q, w\rangle_{\partial K} - \langle (\mathsf{n}-\sigma)\hat{q}, w\rangle_{\partial K} = 0, \\
(q,z)_K + (u,z_x)_K - \langle \hat{u}, z\,\mathsf{n}\rangle_{\partial K} = 0, \\
\langle \hat{q}\,\mathsf{n} + \sigma(q-\hat{q}), \mu\rangle_{\partial \mathcal{T}_h} = 0, \\
\langle 3\hat{u}^2\,\mathsf{n} + \tau(u-\hat{u}), \eta\rangle_{\partial \mathcal{T}_h \setminus \Gamma_p} - \langle p\,\mathsf{n}, \eta\rangle_{\partial \mathcal{T}_h} = -\langle g_p, \eta\rangle_{\Gamma_p},
\end{aligned} \tag{3.29}$$

For all $v, w, z \in W_h^k$, and $(\mu, \eta) \in \bar{M}_h^k(0) \times M_h^k(0)$.

3.2.2.1 Choice of the Numerical Fluxes

Theorem 3.2.1 gives the sufficient conditions on the stabilization parameters to make the linear solver stable. For the nonlinear solver, we use the same σ as we suggested for the linear one. However, choosing a constant τ will not result in the best approximation in nonlinear problems. Therefore, we split τ into τ_0 and τ_1 which are corresponding to the linear and nonlinear parts of total flux. Hence, in (3.27), $\mathcal{H} = 3\hat{u}^2 - p + (\tau_0 + \tau_1)(u - \hat{u})\,\mathsf{n}$. Based on Theorem 3.2.1, $-\tau_0$ should be a constant negative real value; hence, $\tau_0 > 0$. For τ_1, one option can be based on a Lax-Friedrichs type of flux [54, 57]; however, according to Theorem 3.2.1, we still want $\tau_1 < 0$. Hence, we choose

$\tau_1 = -|\partial(3\hat{u}^2)/\partial\hat{u}|$, and finally τ can be written as:

$$\tau(\hat{u}) = -\left|\frac{\partial(3\hat{u}^2)}{\partial\hat{u}}\right| + \tau_0 = -6\,|\hat{u}| + \tau_0,$$

with τ_0 being a constant positive real value.

3.2.2.2 Implementation

To implement the nonlinear solver, we apply the Newton-Raphson method to the system of equations (3.29). Hence, having the current iteration $\bar{u}, \bar{q}, \bar{p} \in W_h^k$ and $(\bar{\hat{u}}, \bar{\hat{q}}) \in M_h^k(g_u) \times \bar{M}_h^k(g_q)$, and denoting $\tau(\bar{\hat{u}})$ with $\bar{\tau}$, we want to find the increments $\delta u, \delta q, \delta p \in W_h^k$ and $(\delta\hat{u}, \delta\hat{q}) \in M_h^k(0) \times \bar{M}_h^k(0)$ such that:

$$\begin{aligned}
\tilde{\mathscr{a}}(\delta u, v) - \mathscr{b}^{\mathrm{T}}(\delta p, v) + \tilde{\mathscr{c}}_1(\delta\hat{u}, v) &= \tilde{\mathscr{f}}(v),\\
\mathscr{a}(\delta p, w) + \mathscr{b}(\delta q, w) - \mathscr{c}_2(\delta\hat{q}, w) - \mathscr{d}_2(\delta q, w) &= \tilde{\mathscr{h}}(w),\\
\mathscr{a}(\delta q, z) + \mathscr{b}(\delta u, z) - \mathscr{c}_3(\delta\hat{u}, z) &= \tilde{\mathscr{k}}(z),\\
\mathscr{c}_4^{\mathrm{T}}(\delta q, \mu) + \mathscr{e}_1(\delta\hat{q}, \mu) &= \tilde{\mathscr{r}}(\mu),\\
\bar{\mathscr{c}}_5^{\mathrm{T}}(\delta u, \eta) - \mathscr{c}_6^{\mathrm{T}}(\delta p, \eta) + \tilde{\mathscr{e}}_2(\delta\hat{u}, \eta) &= \tilde{\mathscr{s}}(\eta),
\end{aligned} \tag{3.30}$$

with $v, w, z \in W_h^k$, and $(\mu, \eta) \in \bar{M}_h^k(0) \times M_h^k(0)$. The bilinear forms are similar to (3.14), except the following:

$$
\begin{aligned}
\tilde{a}(\delta u, v) &= \frac{1}{\Delta t}(\delta u, v)_{\mathcal{T}_h} - (6\,\bar{u}\,\delta u, \partial_x v)_{\mathcal{T}_h} + \langle \bar{\tau}\,\delta u, v\rangle_{\partial\mathcal{T}_h}\,, \\
\tilde{c}_1(\delta\hat{u}, v) &= \left\langle 6\,\bar{\hat{u}}\,\delta\hat{u}\,\mathsf{n}, v\right\rangle_{\partial\mathcal{T}_h} + \left\langle \left[\frac{\partial\bar{\tau}}{\partial\bar{\hat{u}}}(\bar{u}-\bar{\hat{u}}) - \bar{\tau}\right]\delta\hat{u}, v\right\rangle_{\partial\mathcal{T}_h}, \\
\tilde{e}_2(\delta\hat{u}, \eta) &= \left\langle 6\,\bar{\hat{u}}\,\delta\hat{u}\,\mathsf{n}, \eta\right\rangle_{\partial\mathcal{T}_h} + \left\langle \left[\frac{\partial\bar{\tau}}{\partial\bar{\hat{u}}}(\bar{u}-\bar{\hat{u}}) - \bar{\tau}\right]\delta\hat{u}, \eta\right\rangle_{\partial\mathcal{T}_h}, \\
\tilde{\ell}(v) &= (f, v)_{\mathcal{T}_h} - \frac{1}{\Delta t}\left(\bar{u} - u^{n-1}, v\right)_{\mathcal{T}_h} + (\bar{p}_x, v)_{\mathcal{T}_h} \\
&\qquad + \left(3\bar{u}^2, \partial_x v\right)_{\mathcal{T}_h} - \left\langle 3\bar{\hat{u}}^2 + \bar{\tau}(\bar{u}-\bar{\hat{u}})\,\mathsf{n}, v\right\rangle_{\partial\mathcal{T}_h}, \\
\tilde{h}(w) &= -(\bar{p}, w)_{\mathcal{T}_h} - (\bar{q}, \partial_x w)_{\mathcal{T}_h} + \langle\sigma\bar{q}, w\rangle_{\partial\mathcal{T}_h} + \langle(\mathsf{n}-\sigma)\bar{\hat{q}}, w\rangle_{\partial\mathcal{T}_h}, \\
\tilde{k}(z) &= -(q, z)_{\mathcal{T}_h} - (u, z_x)_{\mathcal{T}_h} + \langle\hat{u}, z\,\mathsf{n}\rangle_{\partial\mathcal{T}_h}, \\
\tilde{r}(\mu) &= -\langle\bar{\hat{q}}\,\mathsf{n} + \sigma(\bar{q} - \bar{\hat{q}}), \mu\rangle_{\partial\mathcal{T}_h}, \\
\tilde{s}(\eta) &= \langle g_p, \eta\rangle_{\Gamma_p} - \left[\langle 3\hat{u}^2\,\mathsf{n} + \tau(u-\hat{u}), \eta\rangle_{\partial\mathcal{T}_h\setminus\Gamma_p} - \langle 3p\,\mathsf{n}, \eta\rangle_{\partial\mathcal{T}_h}\right].
\end{aligned}
\tag{3.31}
$$

Next, let us discretize system of equations (3.30), to obtain the following matrix equations:

$$
\begin{aligned}
\tilde{A}\,\delta U - B^{\mathrm{T}}\,\delta P + \tilde{C}_1\,\delta\Lambda &= \tilde{F}, \\
A\,\delta P + (B - D_2)\,\delta Q - C_2\,\delta\Psi &= \tilde{H}, \\
A\,\delta Q + B\,\delta U - C_3\,\delta\Lambda &= \tilde{K}, \\
C_4^{\mathrm{T}}\,\delta Q + E_1\,\delta\Psi &= \tilde{R}, \\
C_5^{\mathrm{T}}\,\delta U - C_6^{\mathrm{T}}\,\delta P + \tilde{E}_2\,\delta U &= \tilde{S}.
\end{aligned}
\tag{3.32}
$$

The process of solving this system of equations is similar to the linear case. We will use the first three equations to obtain δU, δQ, and δP in terms of $\delta\Lambda$ and $\delta\Psi$; then we solve for $\delta\Lambda$ and $\delta\Psi$ using the last two equations. It is

worth noting that in the process of condensing the interior unknowns on the numerical traces, we just solve a series of independent local equations, which can be done simultaneously.

3.3 Numerical Experiments

In this section we will solve a number of simple examples to study the accuracy and capability of the proposed method. In all of these experiments we will use a first order backward difference scheme for time discretization. We will first examine the convergence rate of the computed u, q, p for linear and nonlinear problems. Afterwards, we use the method to solve a few well-known problems in the context of dispersive wave problems.

Example 1: In the first example, we solve the equation (3.3), with $\alpha_2 = -1$, $\alpha_1 = 0$, and $f(x) = 0$ in the domain $\Omega = [0, \pi]$. The initial condition is taken $u_0 = \sin x$, and the boundary conditions are $u(0,t) = -u(\pi,t) = -\sin(t)$, and $q(0,t) = \cos t$. Obviously, the exact solution to this problem is $u = \sin(x-t)$. Meanwhile, the boundary conditions are not periodic, but would result in a well-posed problem. We use a constant and appropriately small time step with different mesh sizes. The values of τ, and σ are equal to -10, and 10, respectively. Hence, the convergence rate of the computed solutions u, q, p at the final time level $T = 0.1$ is calculated. These rates are listed in Table 3.1 for different number of cells and different orders of the polynomial approximation. As one can observe, all of the approximate solutions u, q, p are converging with

$k+1$.

In the second part of this example we consider a similar problem, except at the right side of the domain we apply the boundary condition on p, leading to the boundary data: $u(0,t) = -\sin t$, $p(\pi,t) = -\sin t$, and $q(0,t) = \cos t$. It is worth noting that the boundary condition on p is applied through the flux conservation condition (3.11). Since this set of boundary conditions provide a well-posed problem, one might expect the proposed numerical method to converge at the optimal rates for each $k \geq 0$. We list the corresponding rates of convergence in Table 3.2. Note that unlike the previous case where, we obtain optimal convergence for all $k \geq 0$, in this example, when boundary data on p is set in tandem with no information on the variation of the solution *within* the interior of the cell, i.e. the case of $k = 0$, we see a loss of uniform optimal convergence. Here, the solution of q will be unique up to a constant, and although we can satisfy all of the boundary conditions, we lose the optimal convergence. Nevertheless, the uniform optimal convergence is still observed for each u, q, p whenever $k \geq 1$.

Example 2 In this sample problem, we examine the convergence of the method in solving a nonlinear problem. In Eq. (3.3) we let $\alpha_2 = -1$, $\alpha_1 = 3$, and $f(x) = 7\cos(2x-t) + 6\sin(4x-2t)$, and solve the equation in the domain $\Omega = [0, \pi]$. As for the initial condition we apply $u(x,0) = \sin(2x)$, and the boundary conditions are $u(0,t) = u(\pi,t) = -\sin t$, and $q(0,t) = 2\cos t$. Thus, the exact solution of the problem is $u = \sin(2x-t)$. The stabilization param-

Table 3.1: Convergence rates of the solution of the linear problem (example 1), with the right side boundary condition on u. The analytical solutions are denoted by u_{e}, q_{e}, and p_{e}.

Order (k)	Num. of cells	$\|u - u_{\mathrm{e}}\|_{L^2(\Omega)}$	$\|q - q_{\mathrm{e}}\|_{L^2(\Omega)}$	$\|p - p_{\mathrm{e}}\|_{L^2(\Omega)}$
0	10	5.95E-02	8.14E-02	2.16E-01
	20	3.06E-02	3.74E-02	1.12E-01
	40	1.55E-02	1.79E-02	5.73E-02
	Convergence rate	0.98	1.09	0.95
1	10	1.12E-03	3.95E-03	5.89E-02
	20	1.97E-04	7.60E-04	1.48E-02
	40	4.12E-05	1.39E-04	3.72E-03
	Convergence rate	2.38	2.41	1.99
2	10	2.48E-04	1.16E-03	4.37E-03
	20	3.18E-05	1.80E-04	5.71E-04
	40	3.47E-06	2.32E-05	5.22E-05
	Convergence rate	3.08	2.82	3.19
3	10	3.36E-06	5.23E-06	3.34E-05
	20	2.09E-07	3.01E-07	2.04E-06
	40	1.59E-08	2.17E-08	1.33E-07
	Convergence rate	3.86	3.96	3.98

Table 3.2: Convergence rates of the solution of linear problem (example 1), with the right side boundary condition on p. The analytical solutions are denoted by u_e, q_e, and p_e.

Order (k)	Num. of cells	$\|u - u_e\|_{L^2(\Omega)}$	$\|q - q_e\|_{L^2(\Omega)}$	$\|p - p_e\|_{L^2(\Omega)}$
1	10	1.13E-03	4.04E-03	5.89E-02
	20	1.98E-04	7.60E-04	1.48E-02
	40	4.12E-05	1.39E-04	3.72E-03
	Convergence rate	2.39	2.43	1.99
2	10	2.46E-04	1.16E-03	4.23E-03
	20	3.17E-05	1.79E-04	5.58E-04
	40	3.47E-06	2.32E-05	5.15E-05
	Convergence rate	3.07	2.83	3.18
3	10	3.36E-06	5.25E-06	3.35E-05
	20	2.09E-07	3.01E-07	2.03E-06
	40	1.59E-08	2.18E-08	1.33E-07
	Convergence rate	3.86	3.96	3.99

eters are $\sigma = 1$, and $\tau_0 = 20$. In this problem, the stabilization parameters have a noticeable effect on the accuracy of the solution, and need to be chosen carefully to result in optimal convergence. The convergence rates of the approximate solutions are presented in Table 3.3. It is worthwhile noting that the approximate solutions u, p, and q are converging with optimal convergence for polynomial orders $0 \leq k \leq 3$.

Next, we test the method for the case where the boundary condition on the right side of the domain is applied on p instead of u. The convergence results for this problem are listed in Table 3.4. Similar to the linear case with the boundary condition on p, the method results in the optimal convergence for polynomial orders $k \geq 1$.

Table 3.3: Convergence rates of the solution of nonlinear problem (example 2), with the right side boundary condition on u. The analytical solutions are denoted by u_{e}, q_{e}, and p_{e}.

Order (k)	Num. of cells	$\|u - u_{\mathrm{e}}\|_{L^2(\Omega)}$	$\|q - q_{\mathrm{e}}\|_{L^2(\Omega)}$	$\|p - p_{\mathrm{e}}\|_{L^2(\Omega)}$
0	10	2.22E-01	5.92E-01	1.71E+00
	20	1.23E-01	2.98E-01	9.86E-01
	40	6.49E-02	1.49E-01	5.33E-01
	Convergence rate	0.93	1.00	0.84
1	10	1.08E-02	5.04E-02	3.46E-01
	20	2.46E-03	1.21E-02	8.83E-02
	40	5.97E-04	2.99E-03	2.22E-02
	Convergence rate	2.04	2.04	1.98
2	10	1.62E-03	3.94E-03	2.61E-02
	20	2.06E-04	4.79E-04	3.17E-03
	40	2.59E-05	5.93E-05	3.93E-04
	Convergence rate	2.99	3.03	3.03
3	10	7.88E-05	1.56E-04	1.39E-03
	20	4.99E-06	9.28E-06	8.72E-05
	40	3.15E-07	5.76E-07	5.50E-06
	Convergence rate	3.99	4.04	3.99

Table 3.4: Convergence rates of the solution of nonlinear problem (example 2), with the right side boundary condition on p. The analytical solutions are denoted by u_{e}, q_{e}, and p_{e}.

Order (k)	Num. of cells	$\|u-u_{\mathrm{e}}\|_{L^2(\Omega)}$	$\|q-q_{\mathrm{e}}\|_{L^2(\Omega)}$	$\|p-p_{\mathrm{e}}\|_{L^2(\Omega)}$
1	10	1.09E-02	5.11E-02	3.46E-01
	20	2.46E-03	1.21E-02	8.83E-02
	40	5.97E-04	2.99E-03	2.22E-02
	Convergence rate	2.04	2.02	1.99
2	10	1.59E-03	3.99E-03	2.64E-02
	20	2.06E-04	4.79E-04	3.17E-03
	40	2.59E-05	5.93E-05	3.93E-04
	Convergence rate	2.99	3.01	3.01
3	10	7.88E-05	1.56E-04	1.38E-03
	20	4.99E-06	9.28E-06	8.72E-05
	40	3.15E-07	5.79E-07	5.50E-06
	Convergence rate	3.99	4.00	3.99

Example 3 In the previous examples, we have shown the convergence properties of the method; in this example, we are solving for the classical solution of (3.3) with $\alpha_2 = 1$, $\alpha_1 = 3$, and $f(x,t) = 0$. We solve this equation taking $(x,t) \in [-10,0] \times (0,2]$, with initial condition $u(x,0) = 2\,\mathrm{sech}^2(x+4)$. An exact solution to this problem is $u(x,t) = 2\,\mathrm{sech}(x-4t+4)$; therefore, we extract the relevant boundary data for $u(-10,t)$, $u(0,t)$, and $q(0,t)$ and include them in IBVP definition. Since, $\alpha_2 = 1$, we have applied the q-boundary condition at the right end of the domain. Results are computed using 100 elements with 3rd order polynomials. The time-step size in this example is $\Delta t = 10^{-3}$.

The space-time graphs of the computed solution are shown in Figs. 3.2–3.4. Moreover, the relevant analytical solutions are also shown, for a side by side comparison. Although, we have not chosen our time-step small enough

for error calculation, one can still observe a good match between the computed and analytical solutions.

Example 4 In this experiment, we examine the interaction of two solitary waves with different propagation speeds [75]. We want to solve equation (3.3), with $\alpha_2 = 1$, $\alpha_1 = 3$, and $f(x,t) = 0$ with $(x,t) \in [-20, 0] \times (0, 2]$. The initial condition is assumed as:

$$u_0(x) = 5\,\frac{4.5\,\mathrm{csch}^2\left[1.5(x+14.5)\right] + 2\,\mathrm{sech}^2(x+12)}{\left\{3\coth\left[1.5(x+14.5)\right] - 2\tanh(x+12)\right\}^2}.$$

Next, we use the following exact solution to extract three required boundary data, to complete the problem definition:

$$u(x,t) = 5\,\frac{4.5\,\mathrm{csch}^2\left[1.5(x-9t+14.5)\right] + 2\,\mathrm{sech}^2(x-4t+12)}{\left\{3\coth\left[1.5(x-9t+14.5)\right] - 2\tanh(x-4t+12)\right\}^2}.$$

Similar to the previous example, $\alpha_2 = 1$ in (3.3); hence we apply the boundary condition on p at the right end of the domain. The results are computed using 50 elements with 4th order polynomials. The time-step size is also taken as: $\Delta t = 10^{-5}$.

Figs. 3.5–3.7 shows the space-time graphs of the two solitons interacting with each other. In the first phase, the two waves are approaching, and around $t = 0.5$, they overlap each other. Afterwards, the faster soliton continues to propagate and leaves the domain of analysis. The analytical solutions are also presented in this figure, for comparison purposes. It is worthwhile to note that, one can achieve a better accuracy by using a smaller time-step; nevertheless,

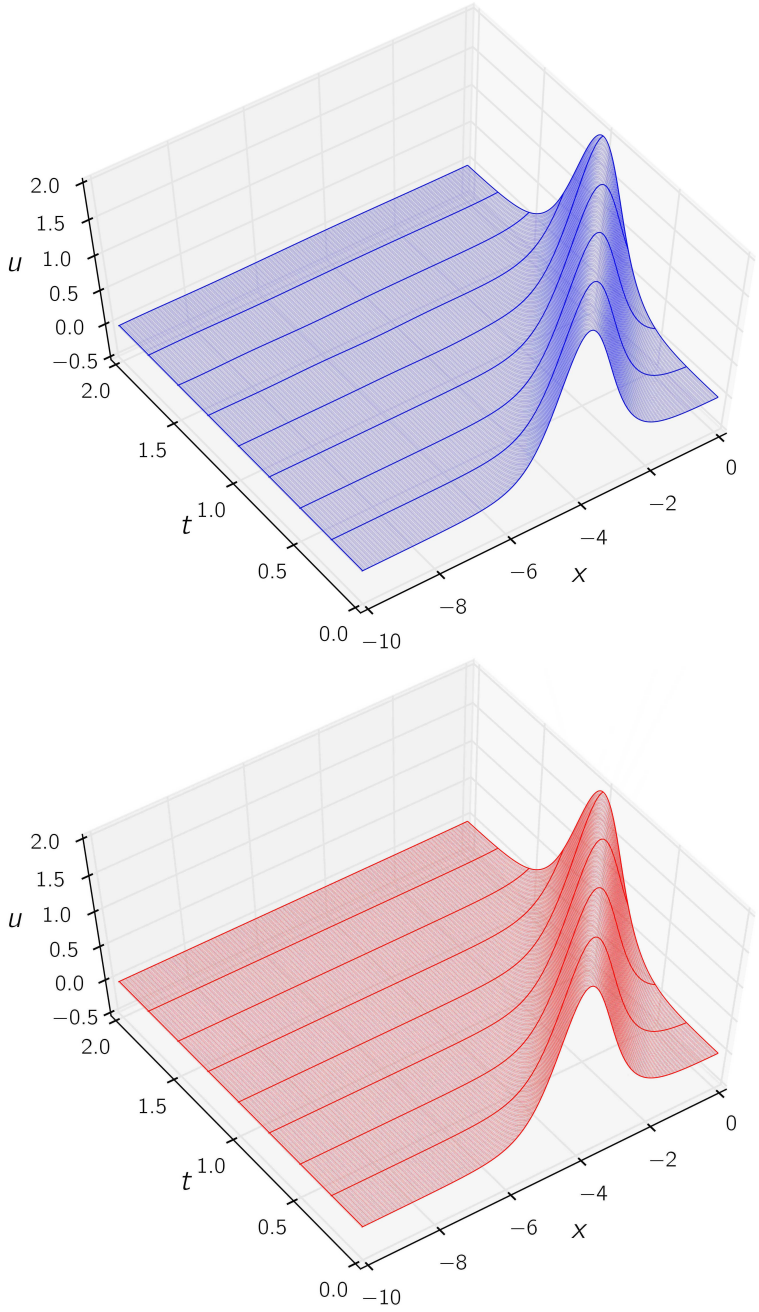


Figure 3.2: Space-time graphs of one soliton in the domain $(x, t) \in [-10, 0] \times (0, 2]$. Evolution of the computed solution (top) and analytical solution (bottom) of u.

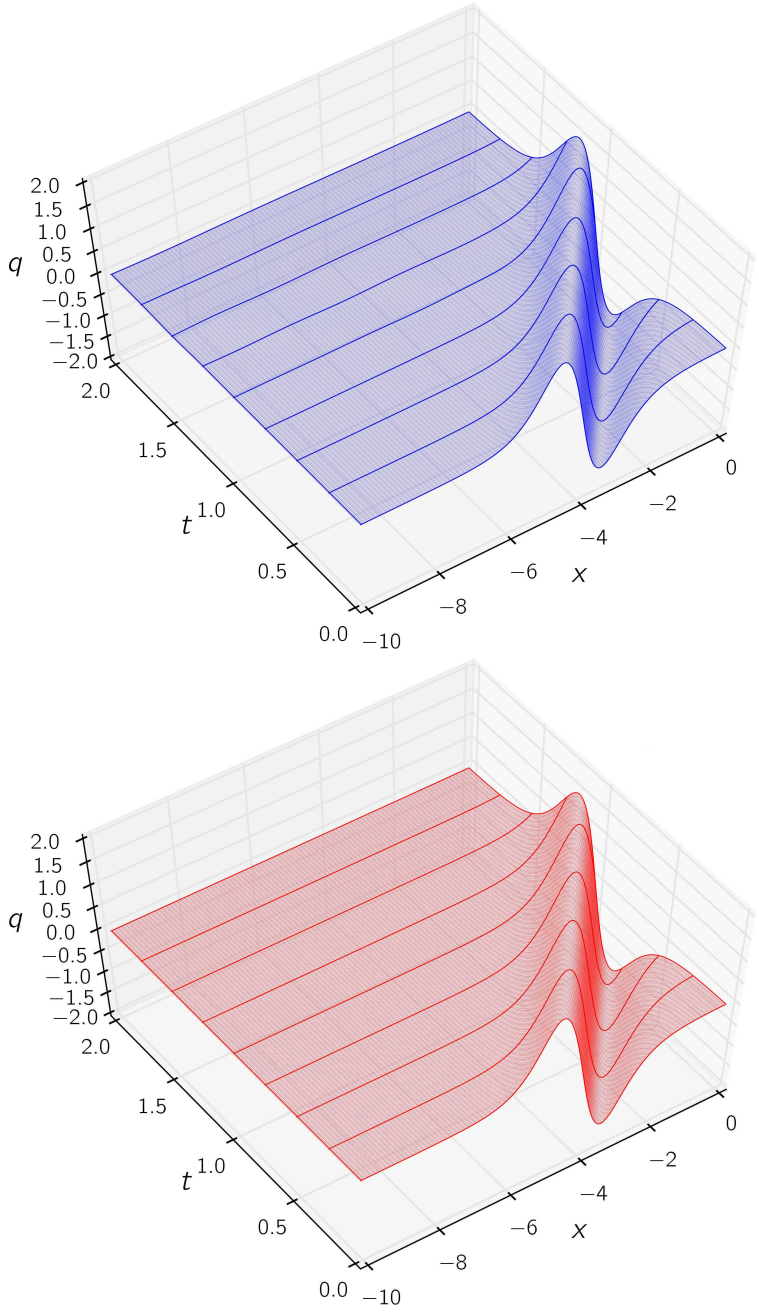


Figure 3.3: Space-time graphs of one soliton in the domain $(x, t) \in [-10, 0] \times (0, 2]$. Evolution of the computed solution (top) and analytical solution (bottom) of q.

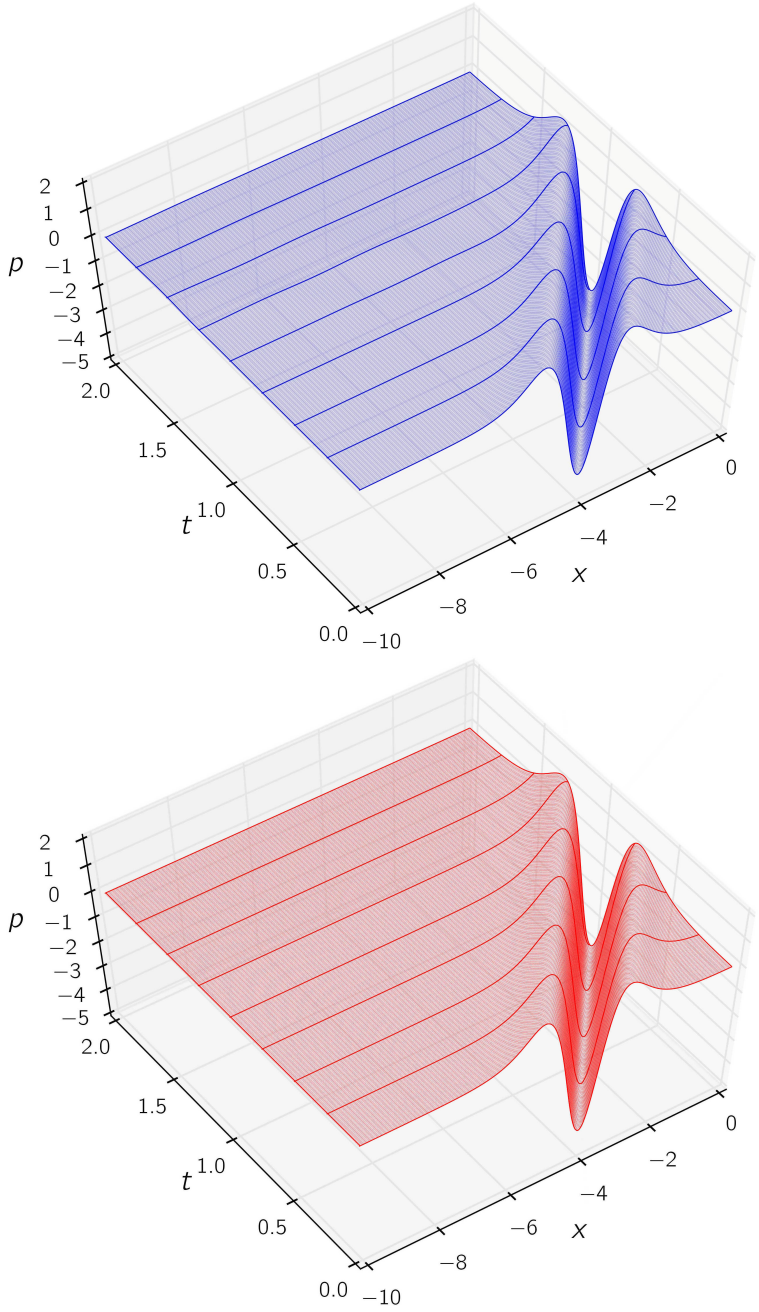


Figure 3.4: Space-time graphs of one soliton in the domain $(x, t) \in [-10, 0] \times (0, 2]$. Evolution of the computed solution (top) and analytical solution (bottom) of p.

even with the current time-step size, we have obtained a very good match between the analytical and computed solutions.

Based on the presented results, we have shown the accuracy of the proposed HDG method. It is worth noting that this technique may be extended to higher order equations. This can perfectly fit into the optimal convergence of HDG in high order derivatives, and give rise to a family of accurate methods for higher order differential equations.

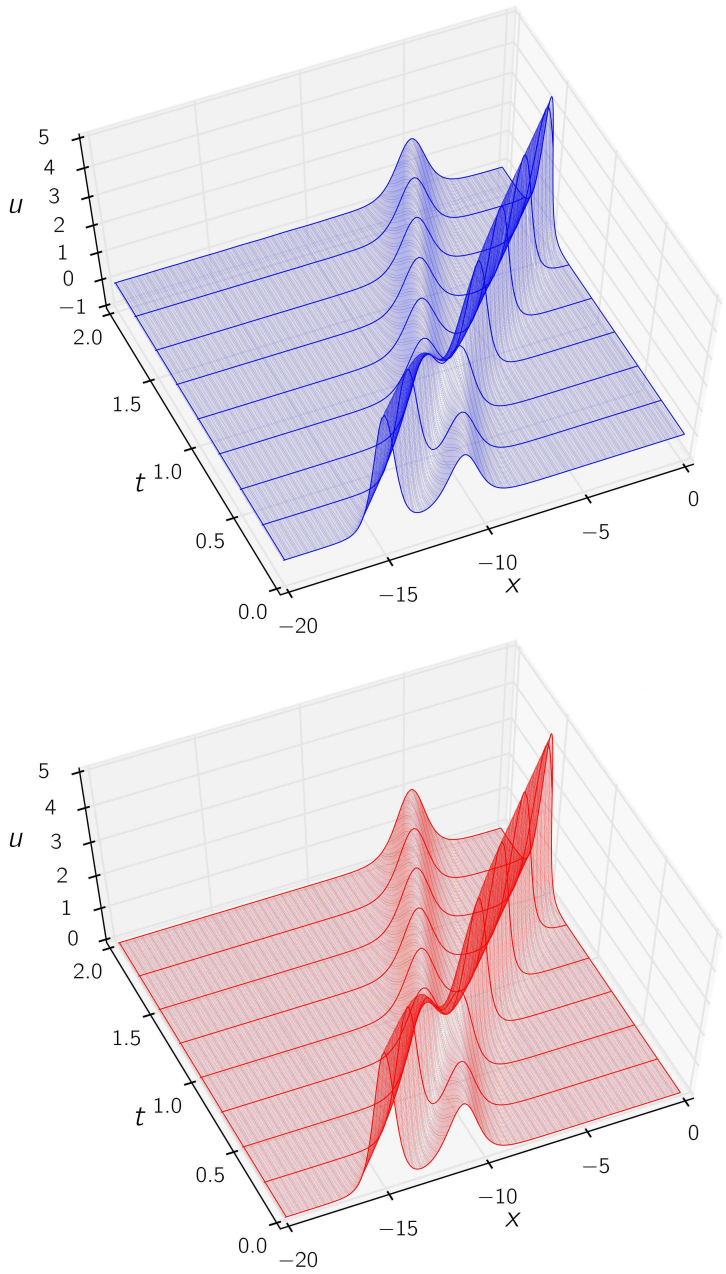


Figure 3.5: Space-time graphs of two solitons in the domain $(x, t) \in [-10, 0] \times (0, 2]$. Evolution of the computed solution (top) and analytical solution (bottom) of u.

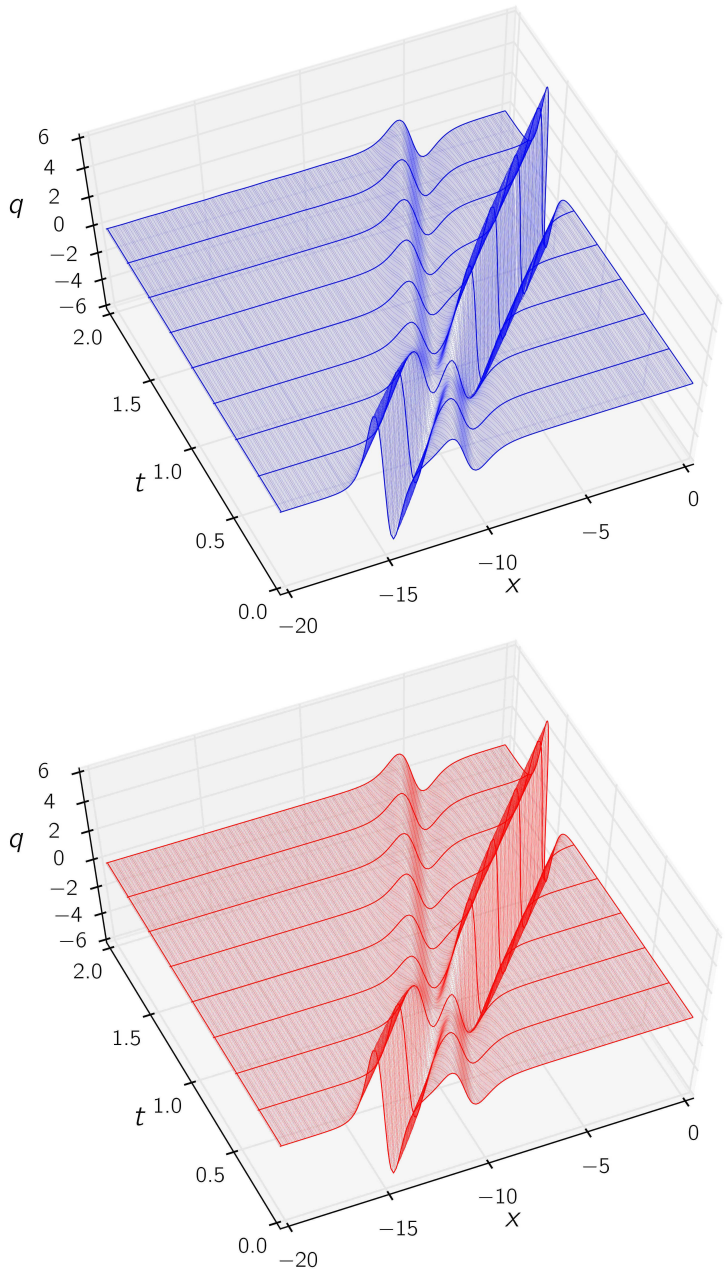


Figure 3.6: Space-time graphs of two solitons in the domain $(x, t) \in [-10, 0] \times (0, 2]$. Evolution of the computed solution (top) and analytical solution (bottom) of q.

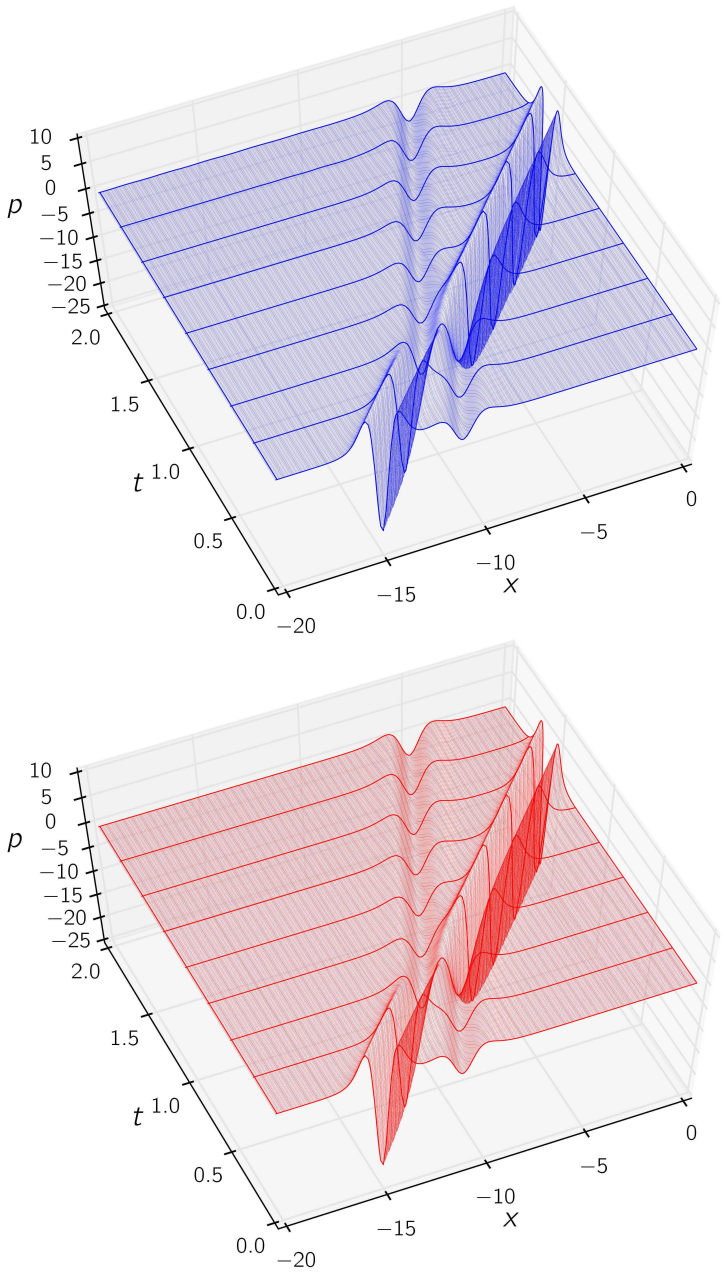


Figure 3.7: Space-time graphs of two solitons in the domain $(x, t) \in [-10, 0] \times (0, 2]$. Evolution of the computed solution (top) and analytical solution (bottom) of p.

Chapter 4

Hybridized Discontinuous Galerkin Method for Nonlinear Shallow Water Equation[1]

In this chapter we present an explicit implementation of the hybridizable discontinuous Galerkin (HDG) method for solving the nonlinear shallow water equation (NSWE). We first follow the common construction of the implicit HDG for nonlinear conservation laws, and then explain the differences between the explicit formulation and the implicit version. For the implicit implementation, we use the approximate traces of the conserved variables $(\hat{h}, \widehat{h\mathbf{u}})$ to express the internal variables and numerical fluxes in each element. Next, we impose the conservation of the numerical fluxes via a global system of equations. Using the Newton-Raphson method, this global system can be solely expressed in terms of the increments of the approximate traces in each iteration. On the other hand, for the explicit method, having $(h, h\mathbf{u})$ at each time level, we first obtain $(\hat{h}, \widehat{h\mathbf{u}})$, such that the conservation of the numerical fluxes is satisfied. This will result in a nonlinear system of equations, which is local to each face of the mesh. Having the solution $((h, h\mathbf{u}),\hat{h}, \widehat{h\mathbf{u}})$ for the previous time step, we use the Runge-Kutta time integration method to ob-

[1]This chapter is based on the paper which will be submitted to Computer Methods in Applied Mechanics and Engineering [60].

tain $(\hat{h}, \widehat{h\mathbf{u}})$ in the next time step. Hence, the introduced explicit technique is based on local operations over the faces and elements of the mesh. Using different numerical examples, we show the optimal convergence of the solution of the explicit approach in the L^2 norm. Finally, through numerical experiments, we discuss the advantages of the implicit and explicit techniques from the computational costs point of view.

4.1 Statement of the Problem

We showed in Chapter 2 that the water wave problem can be approximated by the NSWE up to a good accuracy [40, 41]. This equation reads as follows:

$$\partial_t \boldsymbol{q} + \nabla \cdot \boldsymbol{F}(\boldsymbol{q}) = \boldsymbol{L} \quad \text{in } \Omega \subset \mathbb{R}^d, \tag{4.1}$$

with $\boldsymbol{L}$ being the source term, and

$$\boldsymbol{q} = \begin{Bmatrix} h \\ h\mathbf{u} \end{Bmatrix}, \quad \boldsymbol{F}(\boldsymbol{q}) = \begin{Bmatrix} h\mathbf{u} \\ h\mathbf{u} \otimes \mathbf{u} + \frac{1}{2} g h^2 \mathbf{I} \end{Bmatrix}. \tag{4.2}$$

Here, g is the gravitational acceleration and h is the water depth, i.e. $h(\mathbf{x}, t) = \zeta(\mathbf{x}, t) + H_0 - b(\mathbf{x})$ (refer to Fig. 2.1). Meanwhile, $\mathbf{u}$ is the velocity vector in d spacial dimensions, and $\mathbf{I}$ is the $d \times d$ identity matrix. We assume that h is bounded from below by a minimum positive value.

On the domain boundary ($\partial\Omega$) we can employ different types of boundary conditions, such as periodic conditions, inflow or outflow boundaries, solid wall or any other type of boundary condition which can accompany Eq. (4.1)

and result in a well-posed problem. We postpone the matter of boundary conditions to section 4.2.2, where we discuss the formulation of the method.

4.1.1 Notation

In order to solve Eq. (4.1) with discontinuous finite elements, we define $\mathcal{T}_h = \{K\}$ as a finite collection of disjoint elements partitioning Ω. Also, let $\partial\mathcal{T}_h$ denote all of the faces of the elements in $\mathcal{T}_h$, and $\mathcal{E}_h$ be the set of faces in the mesh. It is worthwhile mentioning that, while in $\mathcal{E}_h$, we count the common faces between two elements only once, the same common face is counted twice when we form $\partial\mathcal{T}_h$. Now, assume e is a common face between two elements K^+ and K^-, i.e. $e = \partial K^+ \cap \partial K^-$. We denote by $\mathbf{n}^\pm$ the unit normals of $K^\pm$ at e and use $[\![\cdot]\!]$ to show the jump of the information across e, e.g. $[\![\boldsymbol{F}\cdot\mathbf{n}]\!] = \boldsymbol{F}^+\cdot\mathbf{n}^+ + \boldsymbol{F}^-\cdot\mathbf{n}^-$, with $\boldsymbol{F}^\pm$ being the values of $\boldsymbol{F}$ corresponding to $K^\pm$. For the faces on the boundary of the domain, where $e \in \partial\mathcal{T}_h \cap \partial\Omega$, we define the jump based on the only contributing face, i.e. $[\![\boldsymbol{F}\cdot\mathbf{n}]\!] = \boldsymbol{F}\cdot\mathbf{n}$.

Throughout this chapter, we mainly use vector notation, with bold italic symbols denoting tuples with $d+1$ components (such as $\boldsymbol{q}$ in Eq. (4.1)). For certain relations, the index notation can provide a more clear description. In those cases, we denote derivatives with respect to spatial coordinates with subscripts, i.e. $q_{i,j}$ denotes the derivative of ith component of $\boldsymbol{q}$ with respect to the jth spatial coordinate. We also use $(v,w)_G$ to denote the inner product of functions v and w in $G \subset \mathbb{R}^d$, i.e. $(v,w)_G = \int_G vw \,\mathrm{d}G$. Furthermore, $\langle v,w\rangle_\Gamma$ denotes $\int_\Gamma vw \,\mathrm{d}\Gamma$, when $\Gamma \subset \mathbb{R}^{d-1}$.

4.1.2 Functional setting

For each element $K \in \mathcal{T}_h$ and $p \geq 0$, let $\mathcal{Q}^p(K)$ denote the space of polynomials of degree at most p in each spatial direction. We choose our trial solution and test spaces, as the set of square integrable functions over $\mathcal{T}_h$, such that their restriction to the domain of K belongs to $\mathcal{Q}^p(K)$; i.e.

$$\mathbf{V}_h^p \coloneqq \{\boldsymbol{q} \in (L^2(\mathcal{T}_h))^{d+1} : \boldsymbol{q}|_K \in (\mathcal{Q}^p(K))^{d+1} \quad \forall K \in \mathcal{T}_h\}. \tag{4.3a}$$

The approximation space over the mesh skeleton ($\mathcal{E}_h$) is defined as:

$$\mathbf{M}_h^p \coloneqq \{\boldsymbol{\mu} \in (L^2(\mathcal{E}_h))^{d+1} : \boldsymbol{\mu}|_e \in (\mathcal{Q}^p(e))^{d+1} \quad \forall e \in \mathcal{E}_h\}. \tag{4.3b}$$

We also define the L^2-projection operator Π_∂, which maps a given $\boldsymbol{\xi} \in (L^2(\mathcal{E}_h))^{d+1}$ to the set of functions whose restriction to $e \in \mathcal{E}_h$ is in $(\mathcal{Q}^p(e))^{d+1}$, and Π_∂ satisfies:

$$\langle \Pi_\partial \boldsymbol{\xi} - \boldsymbol{\xi}, \boldsymbol{\mu} \rangle_e = 0, \quad \forall \boldsymbol{\mu} \in (\mathcal{Q}^p(e))^{d+1}.$$

4.2 Variational formulation

We are looking for a piecewise polynomial solution $\boldsymbol{q}_h \in \mathbf{V}_h^p$ which satisfies Eq. (4.1) in the variational sense. Hence, for all $\boldsymbol{p} \in \mathbf{V}_h^p$ and every $K \in \mathcal{T}_h$, we want $\boldsymbol{q}_h$ to satisfy:

$$(\partial_t \boldsymbol{q}_h, \boldsymbol{p})_K + \langle \boldsymbol{F}_h^*, \boldsymbol{p} \rangle_{\partial K} - (\boldsymbol{F}(\boldsymbol{q}_h), \nabla \boldsymbol{p})_K - \boldsymbol{L}_h(\boldsymbol{p}) = 0. \tag{4.4}$$

Here, $\boldsymbol{F}_h^*$ is the *numerical flux*, an approximation to $\boldsymbol{F}(\boldsymbol{q}) \cdot \mathbf{n}$ over the faces of the element K. Similar to the finite volume method, we can obtain a

stable and convergent solution by a proper choice of $\boldsymbol{F}_h^*$. In hybridizable DG formulation, the numerical flux is defined through the numerical trace ($\widehat{\boldsymbol{q}}_h$), which is an approximation to $\boldsymbol{q}$ on the skeleton space ($\mathcal{E}_h$). Here, we consider the following form for $\boldsymbol{F}_h^*$:

$$\boldsymbol{F}_h^* = \boldsymbol{F}(\widehat{\boldsymbol{q}}_h) \cdot \mathbf{n} + \boldsymbol{\tau}(\boldsymbol{q}_h - \widehat{\boldsymbol{q}}_h), \tag{4.5}$$

where $\boldsymbol{\tau}$ is the stabilization parameter and its choice is important for obtaining a convergent and stable method. We briefly discuss some of the well-known choices for this parameter in section 4.2.1. It is also worth mentioning that $\widehat{\boldsymbol{q}}_h$ is assumed to be single-valued on any given face in $\mathcal{E}_h$.

Next, we also want to satisfy the flux conservation condition across the element faces. Since, the numerical flux is the only means of communication between elements, in all of the internal faces, we require that the projection of the jump of $\boldsymbol{F}_h^*$ onto $\mathbf{M}_h^p$ vanishes, i.e. $\Pi_\partial \llbracket \boldsymbol{F}_h^* \rrbracket = 0$. On the other hand, over the domain boundary ($\partial\Omega$), we apply the boundary condition through the boundary operator $\boldsymbol{B}_h$. Hence, $\forall \boldsymbol{\mu} \in \mathbf{M}_h^p$, we want to have:

$$\langle \boldsymbol{F}_h^*, \boldsymbol{\mu} \rangle_{\partial\mathcal{T}_h \setminus \partial\Omega} + \langle \boldsymbol{B}_h, \boldsymbol{\mu} \rangle_{\partial\mathcal{T}_h \cap \partial\Omega} = 0 \tag{4.6}$$

Here, $\boldsymbol{B}_h$ is the boundary operator, and should be defined according to the applied conditions on $\partial\Omega$. The extended form of $\boldsymbol{B}_h$ for a few boundary condition types is represented in subsection 4.2.2.

We should solve (4.4) and (4.6) to obtain the unknowns of the problem. We can substitute $\boldsymbol{F}_h^*$ from (4.5) into these two equations, and assemble (4.4)

over all of the elements. Thus, the problem may be summarized as finding the approximate solution $(\boldsymbol{q}_h, \widehat{\boldsymbol{q}}_h) \in \mathbf{V}_h^p \times \mathbf{M}_h^p$, such that, for all $(\boldsymbol{p}, \boldsymbol{\mu}) \in \mathbf{V}_h^p \times \mathbf{M}_h^p$:

$$(\partial_t \boldsymbol{q}_h, \boldsymbol{p})_{\mathcal{T}_h} - (\boldsymbol{F}(\boldsymbol{q}_h), \nabla \boldsymbol{p})_{\mathcal{T}_h} + \langle \boldsymbol{\tau} \boldsymbol{q}_h, \boldsymbol{p} \rangle_{\partial \mathcal{T}_h} + \langle \boldsymbol{F}(\widehat{\boldsymbol{q}}_h) \cdot \mathbf{n}, \boldsymbol{p} \rangle_{\partial \mathcal{T}_h} - \langle \boldsymbol{\tau} \widehat{\boldsymbol{q}}_h, \boldsymbol{p} \rangle_{\partial \mathcal{T}_h} - \boldsymbol{L}_h(\boldsymbol{p}) = 0, \tag{4.7a}$$

$$\langle \boldsymbol{F}(\widehat{\boldsymbol{q}}_h) \cdot \mathbf{n}, \boldsymbol{\mu} \rangle_{\partial \mathcal{T}_h \setminus \partial\Omega} + \langle \boldsymbol{\tau} \boldsymbol{q}_h, \boldsymbol{\mu} \rangle_{\partial \mathcal{T}_h \setminus \partial\Omega} - \langle \boldsymbol{\tau} \widehat{\boldsymbol{q}}_h, \boldsymbol{\mu} \rangle_{\partial \mathcal{T}_h \setminus \partial\Omega} + \langle \boldsymbol{B}_h, \boldsymbol{\mu} \rangle_{\partial \mathcal{T}_h \cap \partial\Omega} = 0. \tag{4.7b}$$

Before we explain the solution approach for this problem, a brief discussion on the stabilization parameter and the employed boundary conditions is worthwhile.

4.2.1 Stabilization parameter

As commonly considered in the literature [52], we employ two choices for the stabilization parameter ($\boldsymbol{\tau}$) which are motivated by Lax-Friedrichs and Roe solvers. As for the Lax-Friedrichs approach, we first find $\lambda_{\max}$, the maximum absolute value of the eigenvalues of the Jacobian matrix of the system (denoted by $\boldsymbol{A}$):

$$\boldsymbol{A} = \partial \boldsymbol{F}(\widehat{\boldsymbol{q}}) / \partial \widehat{\boldsymbol{q}} \cdot \mathbf{n} \tag{4.8}$$

Having in mind that $\widehat{\boldsymbol{q}} = \{\hat{h}, \widehat{h\mathbf{u}}\}$, one can simply verify that $\lambda_{\max} = \sqrt{\hat{h}} + |(\widehat{h\mathbf{u}}/\hat{h}) \cdot \mathbf{n}|$ [43]. Now, we define the Lax-Friedrichs stabilization matrix as:

$$\boldsymbol{\tau} = \lambda_{\max} \boldsymbol{I},$$

with $\boldsymbol{I}$ being the $(d+1) \times (d+1)$ identity matrix.

Another option for constructing the stabilization parameter is based on the Roe solver. For this purpose, we form the eigenvalue decomposition of $\boldsymbol{A}$ as: $\boldsymbol{A} = \boldsymbol{R}\,\boldsymbol{\Lambda}\,\boldsymbol{R}^{-1}$, where, $\boldsymbol{R}$ is a matrix whose columns are the right eigenvectors of $\boldsymbol{A}$, and $\boldsymbol{\Lambda}$ contains the eigenvalues of $\boldsymbol{A}$. Thus, we form the stabilization matrix as:

$$\boldsymbol{\tau} = |\boldsymbol{A}| \coloneqq \boldsymbol{R}|\boldsymbol{\Lambda}|\boldsymbol{R}^{-1}, \tag{4.9}$$

with $|\boldsymbol{\Lambda}|$ being a diagonal matrix containing the absolute values of the eigenvalues of $\boldsymbol{A}$.

4.2.2 Boundary conditions

Different types of boundary conditions can be employed in Eq. (4.7). One of the simplest ones is the periodic boundary conditions. Let Γ_1 and Γ_2 denote a pair of periodic boundaries; along these boundaries, we should have $\widehat{\boldsymbol{q}}_h|_{\Gamma_1} = \widehat{\boldsymbol{q}}_h|_{\Gamma_2}$, and the numerical flux passing through them should be conserved. To this end, one can couple the degrees of freedom on Γ_1 and Γ_2 and set the boundary operator in Eq. (4.6) as: $\boldsymbol{B}_h|_{\Gamma_1} = \boldsymbol{F}_h^*|_{\Gamma_1}$, and $\boldsymbol{B}_h|_{\Gamma_2} = \boldsymbol{F}_h^*|_{\Gamma_2}$. This way, the method remains conservative and the required periodic conditions are satisfied.

We can also use Eq. (4.6) to apply inflow/outflow boundary conditions. To this end, we define $\boldsymbol{B}_h$ as:

$$\boldsymbol{B}_h = \boldsymbol{A}^+\boldsymbol{q}_h - |\boldsymbol{A}|\widehat{\boldsymbol{q}}_h - \boldsymbol{A}^-\boldsymbol{q}_\infty,$$

with $\boldsymbol{A}$ and $|\boldsymbol{A}|$ defined in (4.8) and (4.9), and $\boldsymbol{A}^\pm = \boldsymbol{A} \pm |\boldsymbol{A}|$. By applying this boundary condition, when the characteristic directions on a given point

at the boundary are outward, we define the value $\widehat{\boldsymbol{q}}_h$ from $\boldsymbol{q}$. On the other hand, when a characteristic direction is pointing inward, the values of $\widehat{\boldsymbol{q}}_h$ are chosen according to the supported boundary data, i.e. $\boldsymbol{q}_\infty$. As we will see in the second numerical example, by a proper choice of $\boldsymbol{q}_\infty$, we use this boundary condition to apply a wavemaker boundary.

One can also implement a solid wall with slip condition using the boundary operator $\boldsymbol{B}_h$. For this purpose, we need to construct $\boldsymbol{B}_h$, such that $\hat{h}_h = h_h$, and $\hat{\mathbf{u}}_h \cdot \mathbf{n} = 0$ [52].

4.3 Solution procedure

We now consider solving the nonlinear system of equations (4.7) using two approaches. In the first approach, we use an implicit time stepping technique to form a global system of equations and solve it using Newton iterations. In the second approach, we use an explicit time stepping technique to form a set of local nonlinear equations and solve them using local Newton iterations.

4.3.1 Implicit approach

Considering Eq. (4.7), we use Newton-Raphson method to form a linearized equation in terms of the increments of $\boldsymbol{q}_h$ and $\widehat{\boldsymbol{q}}_h$. For the simplicity of the presentation, we consider backward Euler technique as the time integrator, with Δt being the current time step. Hence, denoting by $\boldsymbol{q}_h^{n-1}$ the values of $\boldsymbol{q}_h$ in the previous time level, and $(\bar{\boldsymbol{q}}_h, \bar{\widehat{\boldsymbol{q}}}_h) \in \mathbf{V}_h^p \times \mathbf{M}_h^p$ the corresponding

values in the current iteration, we seek $(\delta \boldsymbol{q}_h, \delta \widehat{\boldsymbol{q}}_h) \in \mathbf{V}_h^p \times \mathbf{M}_h^p$ such that for all $(\boldsymbol{p}, \boldsymbol{\mu}) \in \mathbf{V}_h^p \times \mathbf{M}_h^p$, we have:

$$a_1(\delta \boldsymbol{q}_h, \boldsymbol{p}) + c_1(\delta \widehat{\boldsymbol{q}}_h, \boldsymbol{p}) + f_1(\boldsymbol{p}) = 0, \tag{4.10a}$$

$$\begin{aligned} c_2^T(\delta \boldsymbol{q}_h, \boldsymbol{\mu}) + c_3^T(\delta \boldsymbol{q}_h, \boldsymbol{\mu}) + e_1(\delta \widehat{\boldsymbol{q}}_h, \boldsymbol{\mu}) \\ + e_2(\delta \widehat{\boldsymbol{q}}_h, \boldsymbol{\mu}) + f_2(\boldsymbol{\mu}) + f_3(\boldsymbol{\mu}) = 0. \end{aligned} \tag{4.10b}$$

with the bilinear forms and functionals defined as below:

$$\begin{aligned} a_1(\delta q_j, p_i) &= \frac{1}{\Delta t}(\delta q_j, \delta_{ij} p_i)_{\mathcal{T}_h} - \left(\frac{\partial F_{ik}}{\partial q_j} \delta q_j, p_{i,k} \right)_{\mathcal{T}_h} + \langle \tau_{ij} \delta q_j, p_i \rangle_{\partial \mathcal{T}_h}, \\ c_1(\delta \hat{q}_j, p_i) &= \left\langle \left(\frac{\partial \hat{F}_{ik}}{\partial \hat{q}_j} n_k + \frac{\partial \tau_{ik}}{\partial \hat{q}_j} \bar{q}_k - \frac{\partial \tau_{ik}}{\partial \hat{q}_j} \bar{\hat{q}}_k - \tau_{ij} \right) \delta \hat{q}_j, p_i \right\rangle_{\partial \mathcal{T}_h}, \\ f_1(p_i) &= \frac{1}{\Delta t}(\bar{q}_i - q_i^{n-1}, p_i)_{\mathcal{T}_h} - \langle \hat{F}_{ij} n_j, p_i \rangle_{\partial \mathcal{T}_h} + \langle \tau_{ij} \bar{q}_j, p_i \rangle_{\partial \mathcal{T}_h} \\ &\quad - \langle \tau_{ij} \bar{\hat{q}}_j, p_i \rangle_{\partial \mathcal{T}_h} - (F_{ij}, \partial_j p_i)_{\mathcal{T}_h} - L_i(p_i), \\ c_2^T(\delta q_j, \mu_i) &= \langle \tau_{ij} \delta q_j, \mu_i \rangle_{\partial \mathcal{T} \setminus \partial \Omega}, \quad c_3^T(\delta \boldsymbol{q}_h, \boldsymbol{\mu}) = \left\langle \frac{\partial \boldsymbol{B}_h}{\partial \boldsymbol{q}_h} \delta \boldsymbol{q}_h, \boldsymbol{\mu} \right\rangle_{\partial \Omega}, \\ e_1(\delta \hat{q}_i, \mu_i) &= \left\langle \left(\frac{\partial \hat{F}_{ik}}{\partial \hat{q}_j} n_k + \frac{\partial \tau_{ik}}{\partial \hat{q}_j} \bar{q}_k - \frac{\partial \tau_{ik}}{\partial \hat{q}_j} \bar{\hat{q}}_k - \tau_{ij} \right) \delta \hat{q}_j, \mu_i \right\rangle_{\partial \mathcal{T} \setminus \partial \Omega}, \\ e_2(\delta \widehat{\boldsymbol{q}}_h, \boldsymbol{\mu}) &= \left\langle \frac{\partial \boldsymbol{B}_h}{\partial \widehat{\boldsymbol{q}}_h} \delta \widehat{\boldsymbol{q}}_h, \boldsymbol{\mu} \right\rangle_{\partial \Omega}, \\ f_2(\mu_i) &= \left\langle \hat{F}_{ij} n_j + \tau_{ij} \bar{q}_j - \tau_{ij} \bar{\hat{q}}_j, \mu_i \right\rangle_{\partial \mathcal{T} \setminus \partial \Omega}; \quad f_3(\boldsymbol{\mu}) = \langle \boldsymbol{B}_h, \boldsymbol{\mu} \rangle_{\partial \Omega} \end{aligned} \tag{4.11}$$

In the above definitions, F_{ij}, $\hat{F}_{ij}$, and τ_{ij} denote the element at ith row and jth column of $\boldsymbol{F}(\bar{\boldsymbol{q}}_h)$, $\boldsymbol{F}(\bar{\widehat{\boldsymbol{q}}}_h)$, and $\boldsymbol{\tau}(\bar{\widehat{\boldsymbol{q}}}_h)$, respectively. Meanwhile, δ_{ij} denotes the Kronecker delta.

Thus, the implicit solution approach can be summarized as below:

- Step 1: Having $\bar{\boldsymbol{q}}_h$, and $\bar{\widehat{\boldsymbol{q}}}_h$ from previous time step, solve Eq. (4.10a) in all of the elements to obtain an expression for $\delta\boldsymbol{q}_h$ in terms of $\delta\widehat{\boldsymbol{q}}_h$, $\bar{\boldsymbol{q}}_h$, and $\bar{\widehat{\boldsymbol{q}}}_h$.

- Step 2: Assemble the computed $\delta\boldsymbol{q}_h$ from the previous step into Eq. (4.10b) and form a global system of equations for $\delta\widehat{\boldsymbol{q}}$. Solve the global system and obtain $\delta\widehat{\boldsymbol{q}}_h$.

- Step 3: Use the computed $\delta\bar{\widehat{\boldsymbol{q}}}_h$ from the above global system in Eq. (4.10a) to obtain $\bar{\boldsymbol{q}}_h$, and $\bar{\widehat{\boldsymbol{q}}}_h$ for the next iteration.

- Step 4: Substitute the computed $\bar{\boldsymbol{q}}_h$, and $\bar{\widehat{\boldsymbol{q}}}_h$ from Step 3 into Step 1, and continue iterating through Steps 1-3, for the Newton method to converge.

As mentioned before, the system of equations that we solve in Step 2, is a global system. For later reference, we call this step, the *global step*, while all the other three steps are called *local steps*. From the computational costs point of view, for large problems, solving a sizable global system may not scale well with the computational cores. This is a main motivation to use an explicit method, which does not require solving such a large system of equations.

4.3.2 Explicit approach

Again, we consider Eq. (4.7). As for the explicit time stepping, we first use the computed $\boldsymbol{q}_h^{n-1}$ from the previous time step (or in the first step we take it from the initial condition) in Eq. (4.7b) to derive $\widehat{\boldsymbol{q}}_h^{n-1}$ that results in

the conserved numerical fluxes at time level t_{n-1}. Then, we use $\boldsymbol{q}_h^{n-1}$ and $\widehat{\boldsymbol{q}}_h^{n-1}$ in Eq. (4.7a) along with an explicit time integrator to obtain $\boldsymbol{q}_h^n$ and use it as the initial condition for the next time step. As a result, our solution procedure is comprised of two parts. In the first part, having $\boldsymbol{q}_h^{n-1} \in \mathbf{V}_h^p$ from the last time step and $\bar{\widehat{\boldsymbol{q}}}_h \in \mathbf{V}_h^p$ from the previous iteration, we seek $\delta\widehat{\boldsymbol{q}}_h \in \mathbf{M}_h^p$ such that for all $\boldsymbol{\mu} \in \mathbf{M}_h^p$ we have:

$$e_3(\delta\widehat{\boldsymbol{q}}_h, \boldsymbol{\mu}) + e_2(\delta\widehat{\boldsymbol{q}}_h, \boldsymbol{\mu}) + f_4(\boldsymbol{\mu}) + f_3(\boldsymbol{\mu}) = 0, \tag{4.12}$$

with, e_2 and f_3 defined as in (4.11), while e_3 and f_4 have the following forms:

$$\begin{aligned} e_3(\delta\hat{q}_i, \mu_i) &= \left\langle \left(\frac{\partial \hat{F}_{ik}}{\partial \hat{q}_j} n_k + \frac{\partial \tau_{ik}}{\partial \hat{q}_j} q_k^{n-1} - \frac{\partial \tau_{ik}}{\partial \hat{q}_j} \bar{\hat{q}}_k - \tau_{ij} \right) \delta\hat{q}_j, \mu_i \right\rangle_{\partial\mathcal{T}\setminus\partial\Omega}, \\ f_4(\mu_i) &= \left\langle \hat{F}_{ij} n_j + \tau_{ij} q_j^{n-1} - \tau_{ij} \bar{\hat{q}}_j, \mu_i \right\rangle_{\partial\mathcal{T}\setminus\partial\Omega} \end{aligned} \tag{4.13}$$

A closer look at e_3 and f_4 reveals that we do not perform any iterations on $\boldsymbol{q}_h$. As a result, this equation is local to each face. In other words, instead of solving a large system of equations in implicit method, we solve many small systems of equations, each corresponding to one of the faces in $\mathcal{E}_h$.

After solving Eq. (4.12) through Newton iterations, we have $\boldsymbol{q}_h^{n-1}$ and $\widehat{\boldsymbol{q}}_h^{n-1}$, which satisfy the flux conservation condition. In the second part of the method, we should use some explicit time integration technique to obtain $\boldsymbol{q}_h^n$. For simplicity of formulation, we use forward Euler technique. Hence, we want to find $\boldsymbol{q}_h^n \in \mathbf{V}_h^p$ such that for all $\boldsymbol{p} \in \mathbf{V}_h^p$, we have:

$$\begin{aligned} (\boldsymbol{q}_h^n, \boldsymbol{p}) = (\boldsymbol{q}_h^{n-1}, \boldsymbol{p}) + \Delta t \big(&- \langle \boldsymbol{F}(\widehat{\boldsymbol{q}}_h^{n-1}) \cdot \mathbf{n}, \boldsymbol{p} \rangle - \langle \boldsymbol{\tau} \boldsymbol{q}_h^{n-1}, \boldsymbol{p} \rangle \\ &+ \langle \boldsymbol{\tau} \widehat{\boldsymbol{q}}_h^{n-1}, \boldsymbol{p} \rangle + (\boldsymbol{F}(\boldsymbol{q}_h^{n-1}), \nabla \boldsymbol{p}) + \boldsymbol{L}(\boldsymbol{p}) \big) \end{aligned} \tag{4.14}$$

In other words, the equations in the element interiors are a set of linear equation, which are directly giving the $\boldsymbol{q}_h$ for the next time step. Concerning, this explicit approach, we want to highlight the following points:

- The first step in the above approach, i.e. solving Eq. (4.12) by Newton iterations, is local to each face in $\mathcal{E}_h$. Hence, for large problems, the computational time can be scaled down by increasing the number of working processors. Furthermore, the second part of the approach is also local to each element in $\mathcal{T}_h$. As a result, for large practical problems, this explicit method perfectly fits the high performance computing requirements, i.e. scalability and local operations.

- There are similarities between the above procedure and the Reconstruct-Evolve-Average algorithm, which is used in the finite volume method. Albeit, we prefer to use the terms Extend-Evolve-Project for our approach. This is because, in the first step, we obtain an extension of $\boldsymbol{q}_h^{n-1}$ to a numerical trace $\widehat{\boldsymbol{q}}_h^{n-1}$, which satisfies the flux conservation condition. In the second step, we evolve the solution of the previous step, and finally project it onto $\mathbf{V}_h^p$.

- In order to achieve a higher accuracy for the same time step size, one can use the explicit Runge-Kutta methods instead of the forward Euler method in the above algorithm. In that case, at the end of any Runge-Kutta stage in Eq. (4.14), we have to solve Eq. (4.12) and use those results for the next stage.

- Similar to other explicit techniques, the limitations on the time step size is an important issue in the proposed algorithm. Although by using strong stability-preserving techniques [64] can improve the time step size, we cannot obtain an unconditionally stable technique, similar to what an implicit method can offer.

4.4 Numerical experiments

In this section we present a set of numerical examples to investigate the accuracy, convergence properties and the performance of the described technique. The software that we use to solve these problems is written in C++, and makes use of different numerical libraries such as deal.II, PETSc, MUMPS, and Eigen. This software is an extension of the one developed in [61], and utilizes the shared and distributed memory parallelism.

Example 1: In this example we consider the nondimensionalized version of Eq. (4.1) with a known smooth solution in $\Omega = (-1, 1)^2$. Hence, we want to solve:

$$\partial_t \begin{Bmatrix} h \\ h\mathbf{u} \end{Bmatrix} + \nabla \cdot \begin{Bmatrix} h\mathbf{u} \\ h\mathbf{u} \otimes \mathbf{u} + \frac{1}{2}h^2\mathbf{I} \end{Bmatrix} = \boldsymbol{L}, \quad \text{in } (-1, 1)^2 \subset \mathbb{R}^2. \tag{4.15}$$

Here, all of the variables and derivatives are nondimensionalized, and we define $\boldsymbol{L}$ to balance the above equation for the following manufactured solution:

$$h = 2 + e^{\sin(3x)\sin(3y) - \sin(3t)}, \quad h\mathbf{u} = \left(\cos(4t - x), \sin(4t + y)\right).$$

Fig. 4.1a shows the variation of h at $t = 0$ in the domain. For the implicit approach, we use second order backward difference formula to integrate the

solution in time. For elements of order 0, 1, and 2, we use $\Delta t = 10^{-3}$, and for third order elements we use $\Delta t = 2.5 \times 10^{-4}$ to keep the time integration error below the space discretization error. At time $t = 1$, we compute the $\boldsymbol{L}^2(\Omega)$ error of the $\boldsymbol{q}_h$ and use them to obtain the rates of convergence for a set of uniformly refined meshes. We use Lax-Friedrichs flux in all of the elements, and inflow/outflow boundary condition on all of the boundary faces. As for the space discretization, we utilize modal elements with Legendre polynomials to construct the basis for each element. The finite element mesh will have $2^n \times 2^n$ elements, where $n = 3, 4, 5, 6$. In Fig. 4.1b, we have shown the employed mesh with $n = 6$, which is decomposed into 24 subdomains for a distributed parallel solve. At each Newton-Raphson iteration, the global system of equations is solved using the PETSc wrapper for MUMPS solver. The convergence tolerance for Newton iterations is set to 10^{-12}. Using this tolerance, the maximum number of Newton iterations was six throughout the analysis.

In Fig. 4.2 we have shown the L^2-norm of the error of $\boldsymbol{q}_h$ and the corresponding convergence plots. We observe that in all of the polynomial orders we get the optimal $(p + 1)$ order of convergence. The computational time for solving 1000 time steps for the models with $n = 6$, and different polynomial orders are listed in Table 4.1. For this timing, we have used 24 processors of the Lonestar 5 supercomputer in the Texas Advanced Supercomputing Center (TACC). Since, each node of Lonestar 5 contains 24 processors, this test does not involve any node-to-node communication. However, we have used distributed memory parallelism to run these tests. By no means, we claim

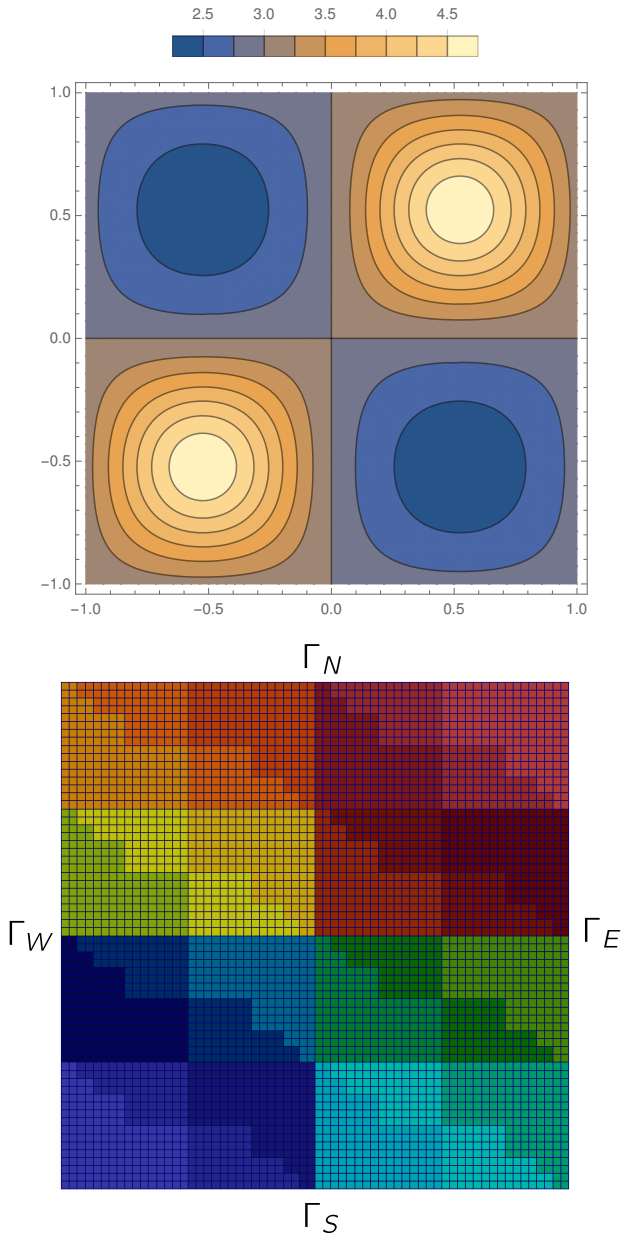


Figure 4.1: (a): The schematic plot of the initial state of h in example 1 at time $t = 0$; (b): The decomposed computational mesh between 24 processors for $n = 6$, i.e. 2^6 elements in each direction.

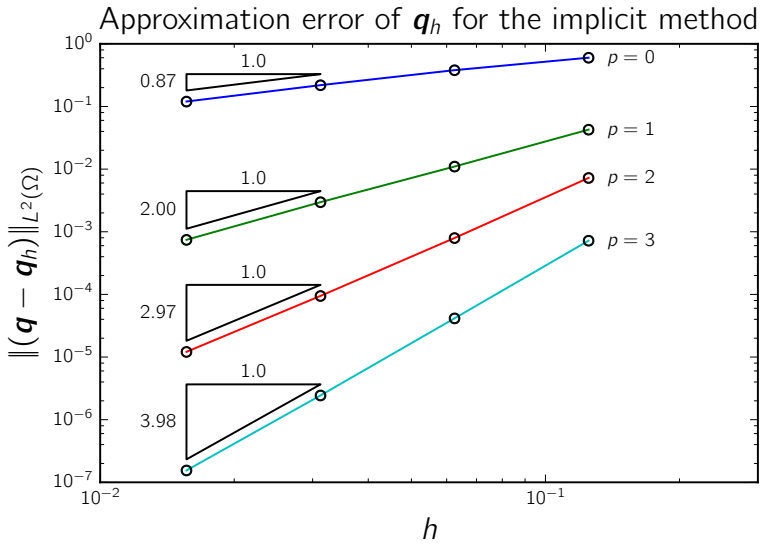


Figure 4.2: Approximation error and convergence rate of the implicit method for solving the first example in $\Omega \equiv [-1,1]^2$, using $2.0/h$ elements in each direction, and polynomial order $p = 0, 1, 2, 3$.

that our code is optimal for conducting a conclusive performance test; nevertheless, both implicit and explicit methods have received the same amount of optimization in our code. Hence, we can use this code for a comparison between the performance of implicit and explicit methods.

Next, we use the introduced explicit method to solve the same problem

Table 4.1: Execution time of local and global steps for solving 1000 time steps of example 1 using the implicit method for the case of $2^6 \times 2^6$ elements, and different polynomial orders p

p	Local DOFs	Global DOFs	Execution time (seconds)			CPU time share (%)	
			Local steps	Global step	Total time	Local Steps	Global Steps
0	12,288	24,960	35.2	48.1	83.3	42%	58%
1	49,152	49,920	92.4	200	292	32%	68%
2	110,592	74,880	224	378	602	37%	63%
3	196,608	99,840	535	636	1171	46%	54%

as explained above. For our time integration algorithm, we use the four stage low-storage explicit Runge-Kutta method. For the case of $p = 0, 1, 2$, we use $\Delta t = 10^{-3}$, and for $p = 3$, we set $\Delta t = 5 \times 10^{-4}$. Similar to the implicit case, we compute the $\boldsymbol{L}^2(\Omega)$-error of $\boldsymbol{q}_h$ at $t = 1$. To solve the decoupled equations on each face, we use MUMPS direct solver. The convergence tolerance for Newton iterations is set to 10^{-12}, which results in a maximum of three iterations per stage for the solution to converge. We use the same meshes as described in the implicit method to solve this problem.

In Fig. 4.3, we have shown the L^2 errors of $\boldsymbol{q}_h$ for meshes with different element sizes and polynomial orders. Similar to the implicit method, we observe optimal convergence for all of the polynomial orders. Moreover, the errors for the same mesh size seems to be smaller than the implicit method errors. This can be due to the higher order time integration technique that we use here, and the strong variation of $\boldsymbol{q}$ with time in this example. We have also listed the computational times for solving 1000 time step of the models with $n = 6$ and different polynomial orders in Table 4.2. One can observe that the explicit solver is around eight times faster than the implicit solver, for the same setup and computational power.

For practical applications one can also increase the time step size of an implicit solver to obtain a comparable performance to the explicit methods. We will consider this in the next example, where the time step size of the implicit solver is around 100 times larger than the explicit method. As such, gaining around eight times performance boost by using an explicit solver,

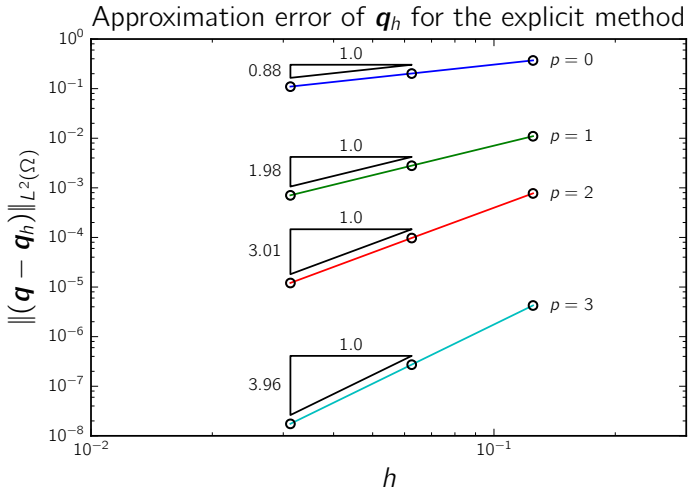


Figure 4.3: Approximation error and convergence rate of the explicit method for solving the first example in $\Omega \equiv [-1, 1]^2$, using $2.0/h$ elements in each direction, and polynomial order $p = 0, 1, 2, 3$.

Table 4.2: Execution time of local and global steps for solving 1000 time steps of example 1 using the explicit method for the case of $2^6 \times 2^6$ elements, and different polynomial orders p

p	Local DOFs	Global DOFs	Execution time (seconds)			CPU time share (%)	
			Local steps	Global step	Total time	Local Steps	Global Steps
0	12,288	24,960	8.9	28.0	37	24%	76%
1	49,152	49,920	17.9	44.2	62.1	29%	71%
2	110,592	74,880	33.2	58.7	91.9	36%	64%
3	196,608	99,840	65.4	80.5	146	44%	55%

may not sound very promising. Nevertheless, one should also note that, in practical applications, the memory usage of an implicit solver can be its main bottleneck. In large problems, this can also result in the higher scalability of explicit solvers, by increasing the number of processing cores.

Example 2: In this example, we consider the propagation of a long wave in a relatively fine mesh. In Eq. (4.15) set $\boldsymbol{L} = 0$, and let us solve it in $\Omega \equiv (-1, 1)^2$. As for the initial condition we set $h|_{t=0} = 3$ and $\mathbf{u}|_{t=0} = 0$ everywhere in the domain. We use solid wall with slip condition on Γ_S, Γ_N, and Γ_E (c.f. Fig. 4.1b). On Γ_W, we apply inflow/outflow boundary condition with $\mathbf{u}_\infty = 0$, and h_∞ defined in the following form:

$$h_\infty = \begin{cases} 2 + \cos(t) & t \leq t_0 \\ 0 & t_0 < t \end{cases}$$

In this example we consider two cases with $t_0 = 1, 3$. Hence, the excitation frequency in both of these cases is the same, and the only difference is the longer loading time in the case with $t_0 = 3$. The case with $t_0 = 1$ imitates the waves induced by a tide from the right side of the domain. We want to solve this problem in the time interval $t \in (0, 20]$.

Our finite element mesh for this problem consists of piecewise quadratic elements with $n = 7$, i.e. $2^7 \times 2^7$ elements. This results in roughly 300,000 global unknowns. We use both implicit and explicit methods to solve this problem. Our initial perception of this process suggests that since the induced waves are fairly long waves, we can choose a large time step size for the implicit method and still obtain a valid solution. Therefore, we choose $\Delta t = 0.1$ for

the implicit method. On the other hand, the time step size for the explicit solver is still restrained by the stability requirements. For our setup, the CFL condition suggests [32]:

$$\Delta t_{\text{Explicit}} \propto \frac{1}{\lambda_{\max}} l_h.$$

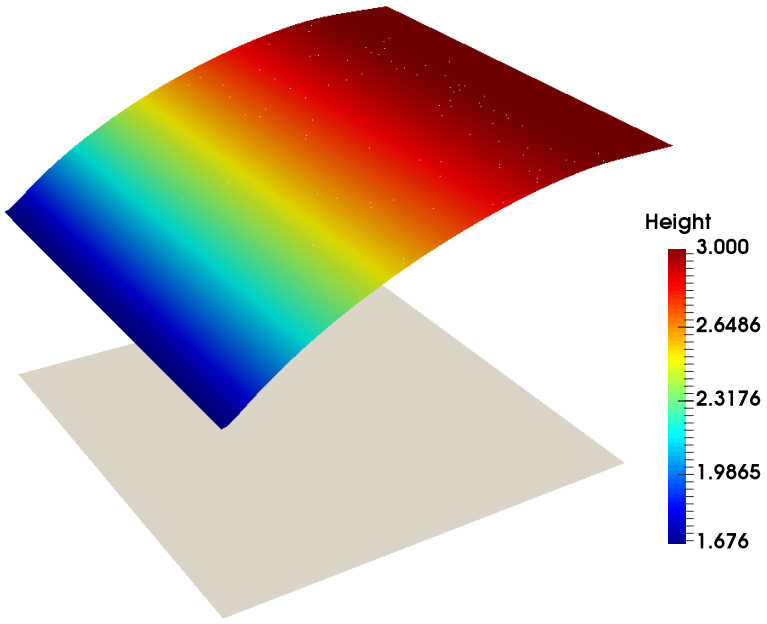
With l_h being the smallest element size, and $\lambda_{\max}$ being the largest eigenvalue of the Jacobian matrix $\boldsymbol{A}$ (as defined in Eq. (4.8)). This results in a time step size around 10^{-3}; although we can choose the time step size adaptively, we use the simple choice of $\Delta t = 10^{-3}$ for the explicit method in all time steps. Thus, if both methods had the same performance for solving the problem at each time step, the implicit method would have been 100 times faster.

The profiles of the water surface at different time steps are shown in Figs. 4.4 and 4.5 for $t_0 = 1$ and $t_0 = 3$, respectively. These figures are obtained using the explicit solver, with $\Delta t = 10^{-3}$. We have also compared the water surface profiles obtained from implicit solver (with $\Delta t = 10^{-1}$) and explicit solver (with $\Delta t = 10^{-3}$) at different time steps In Figs. 4.6 and 4.7. For the case of $t_0 = 1$, i.e. Fig. 4.6, we can see a very good match between the results of $\Delta t = 10^{-1}$ and $\Delta t = 10^{-3}$. On the other hand, in Fig. 4.7, we notice that after $t = 1$, the plots of $\Delta t = 10^{-1}$ and $\Delta t = 10^{-3}$ do not match. Even though both of these figures are obtained for the same wave length, the longer excitation time in the case of $t_0 = 3$ has resulted in the formation of shock waves that require a smaller time step to be resolved in the solution. Hence, when we want to choose the time step size of the implicit solver, the excitation wavelength is not the only deciding factor. Even if the solution seems quite

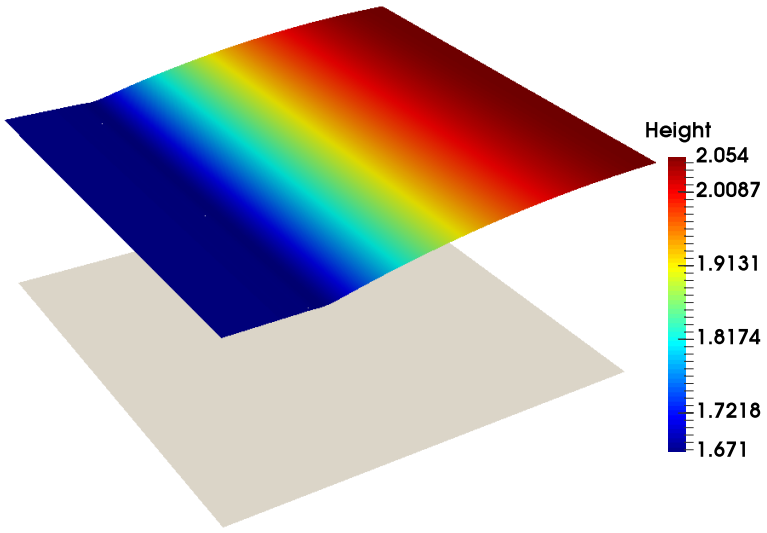
stable and well behaved, we might still loose important phenomena that might be captured if we use a smaller time step.

Despite the accuracy loss for the case of $t_0 = 3$, the implicit method with time step $\Delta t = 0.1$ can produce quite good results for $t_0 = 1$ case. Hence, it is worthwhile to compare the CPU time of the implicit method (with $\Delta t = 10^{-1}$) and explicit method (with $\Delta = 10^{-3}$) for solving the problem using different number of processing cores. In Table 4.3 we have listed the computational time of solving this problem on 12, 24, and 48 cores. Again, we use Lonestar 5 to perform these tests. The first observation in this table is the noticeably lower computational time of the implicit method. As we mentioned earlier, this is due to the much larger time step size of the implicit technique, which requires 200 time steps to solve the problem as opposed to the explicit method, which requires 20000 time steps. Although each time step in the explicit solver is 5 to 8 times faster (depending on the number of processors), it cannot make up for the 100 times fewer time steps required in the implicit method. On the other hand, the explicit solver exhibits a better scalability than implicit method. This can be mainly attributed to the local nature of the computation in the explicit solver, which reduces memory usage and communication between nodes.

Example 3: In our last example, we solve the release of water from a dam into the still water inside a narrowing channel. Consider the trapezoidal domain in Fig. 4.8. This domain is discretized with 192×64 first order elements. We want to solve the nondimensionalized shallow water equa-

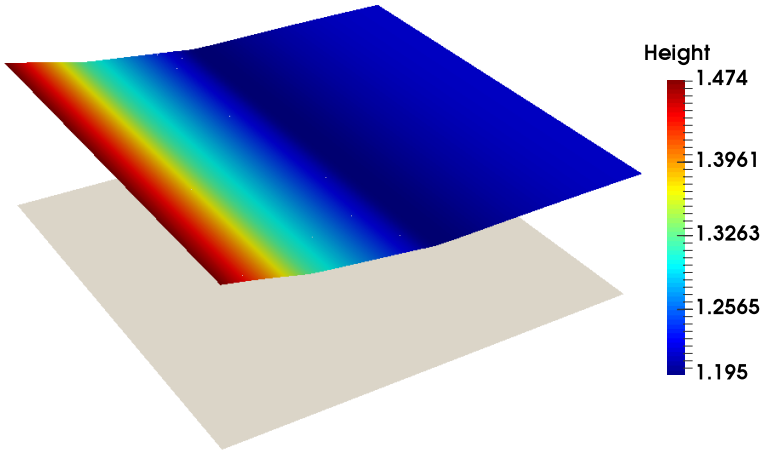


(a) $t = 1.0$

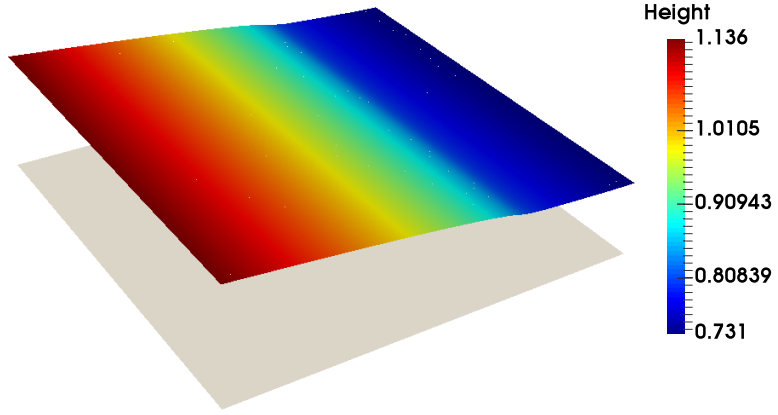


(b) $t = 2.0$

Figure 4.4: The water surface profile of example 2, at different times, with $t_0 = 1$ and $\Delta t = 0.001$.

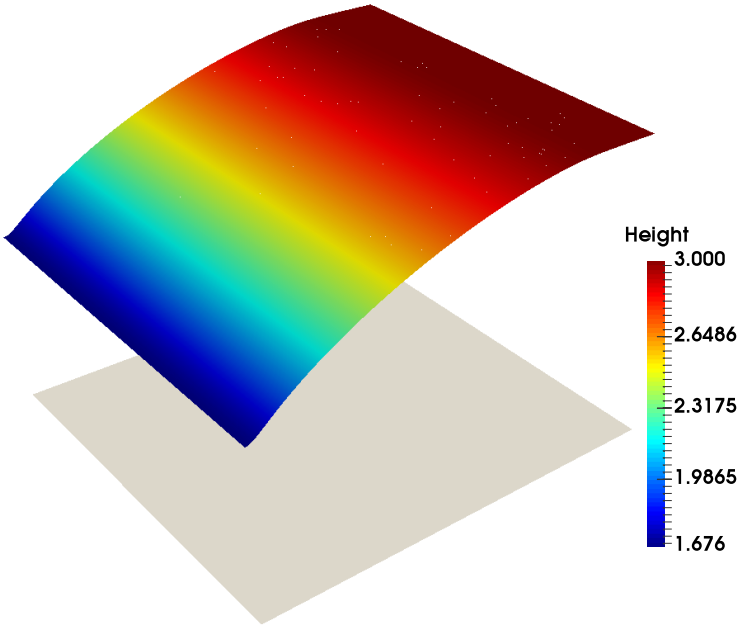


(c) $t = 3.0$

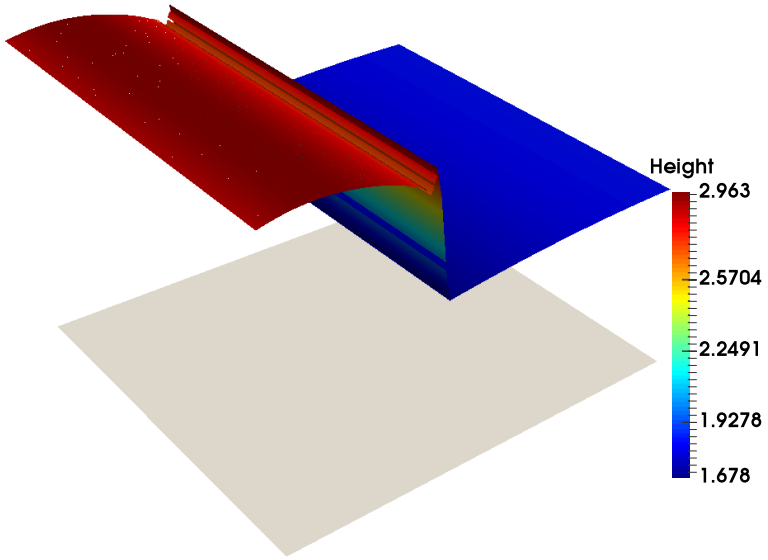


(d) $t = 4.5$

Figure 4.4: (cont'd) The water surface profile of example 2, at different times, with $t_0 = 1$ and $\Delta t = 0.001$.

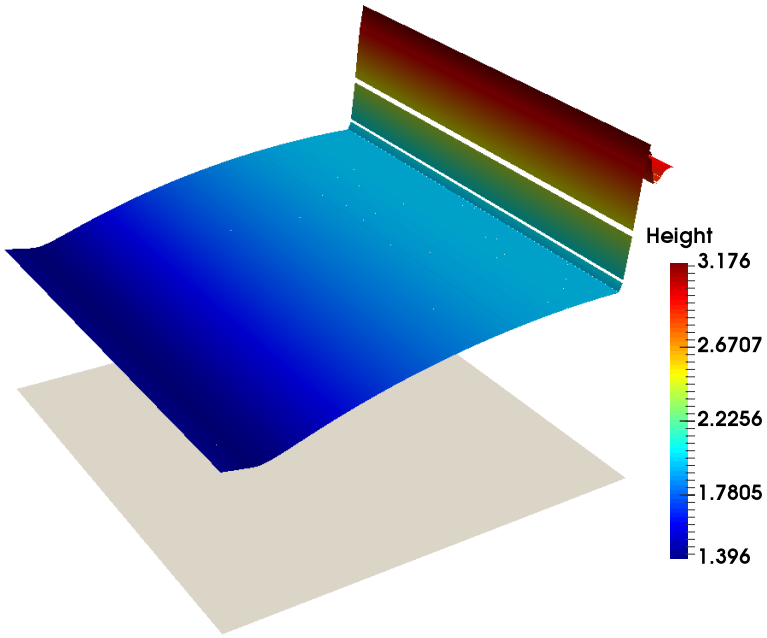


(a) $t = 1.0$

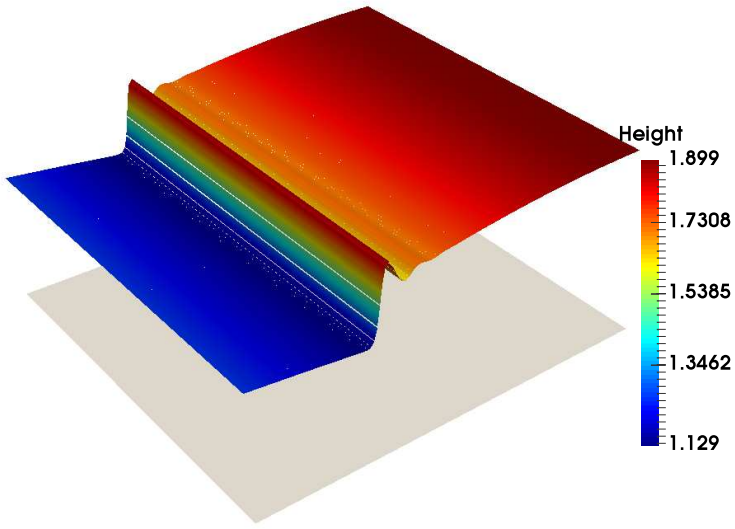


(b) $t = 2.2$

Figure 4.5: The water surface profile of example 2, at different times, with $t_0 = 3$ and $\Delta t = 0.001$.

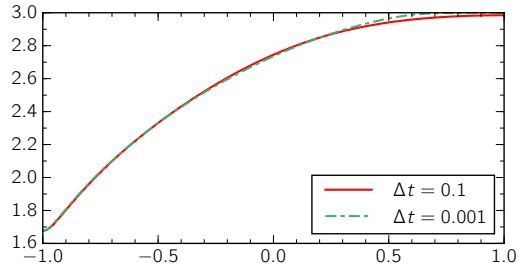


(c) $t = 3.2$

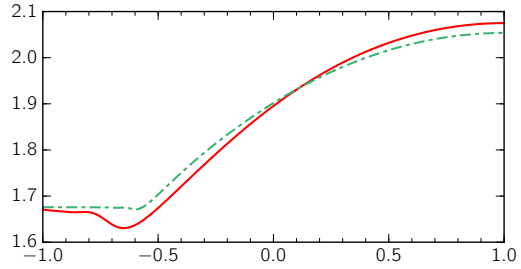


(d) $t = 4.0$

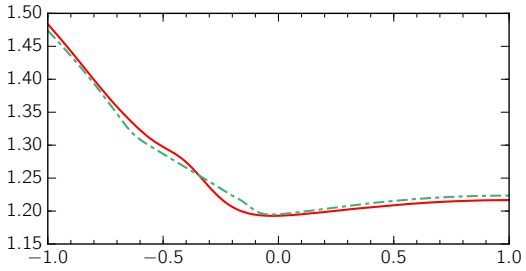
Figure 4.5: (cont'd) The water surface profile of example 2, at different times, with $t_0 = 3$ and $\Delta t = 0.001$.



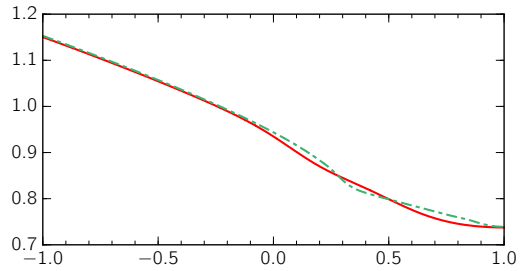
(a) $t = 1.0$



(b) $t = 2.0$

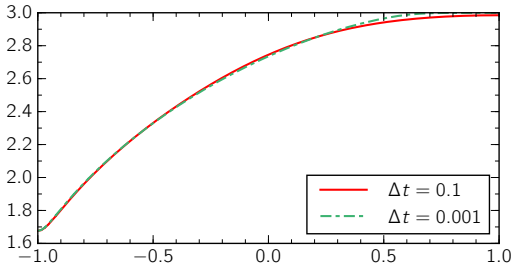


(c) $t = 3.0$

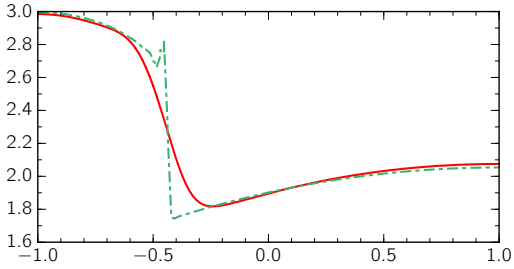


(d) $t = 4.5$

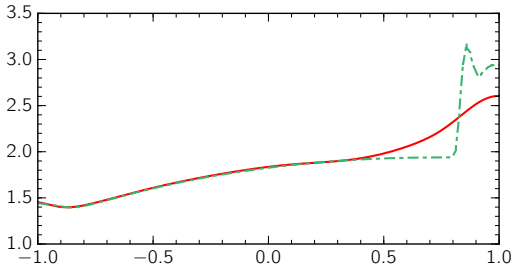
Figure 4.6: Comparison of water surface profile in example 2, between $\Delta t = 0.001$ and $\Delta t = 0.1$, for the case of $t_0 = 1$.



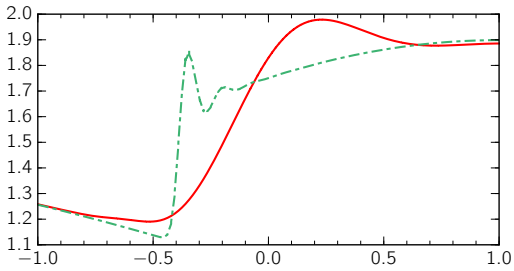
(a) $t = 1.0$



(b) $t = 2.2$



(c) $t = 3.2$



(d) $t = 4.0$

Figure 4.7: Comparison of water surface profile in example 2, between $\Delta t = 0.001$ and $\Delta t = 0.1$, for the case of $t_0 = 3$.

Table 4.3: Execution time and speedup of implicit and explicit methods for solving example 2 with different number of cores. For implicit method we solve 200 steps with $\Delta t = 0.1$, and for explicit approach, we solve 20000 steps with $\Delta t = 0.001$.

Cores	Implicit method CPU time (seconds)				Explicit method CPU time (seconds)			
	Local steps	Global steps	Total	Speedup	Local steps	Global step	Total	Speedup
12	373	357	730	1.00	5522	8733	14255	1.00
24	182	311	493	1.48	2785	5107	7892	1.81
48	94.2	268	362	2.01	1382	2930	4312	3.31

tion (similar to (4.10)) in this domain. As for the initial condition, we set $h|_{t=0} = 1.5$, $\mathbf{u}|_{t=0} = 0$ everywhere in the domain. We apply, inflow/outflow boundary condition on the left boundary (c.f. Fig. 4.8) with $h_\infty = 3$, and $\mathbf{u}_\infty = 0.5\,\mathbf{e}_x$. Meanwhile, on the right boundary, the outflow boundary condition is applied, and on the top and bottom boundaries, we employ solid wall condition. We solve the problem in the time interval $t \in (0, 5]$. In order to integrate the semi-discrete form in time, we use the low-storage fourth order explicit Runge-Kutta method. The time step size is chosen constant equal to 10^{-3} throughout the analysis.

In Fig. 4.9, we show the snapshots of water height at four different time steps. This setup can imitate the water released from a dam into the downstream water at rest. Throughout the channel, the Froude number is less than one, which is an indication of a subcritical flow. Hence, the initial water wave is traveling faster than the flow, until it reaches the narrower sections of the channel. Near the outlet, the water starts to build up, and thus, a new wave start to propagate towards upstream (c.f. Figs. 4.9c,d). This process will eventually results in a steady water profile. Fig. 4.10 presents the velocity in

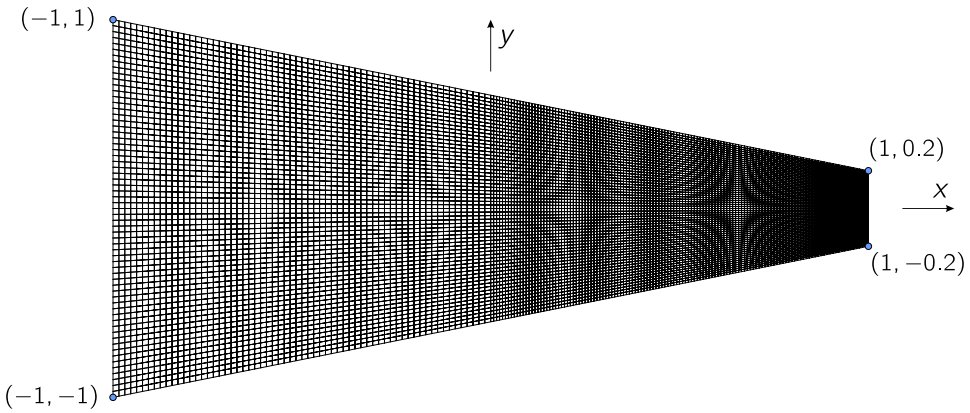
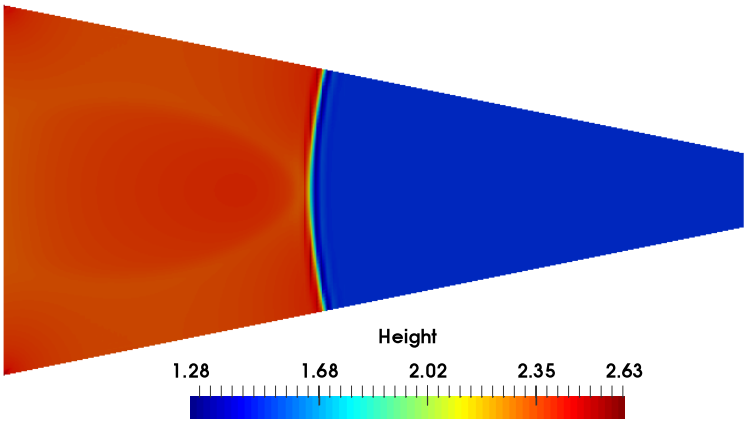


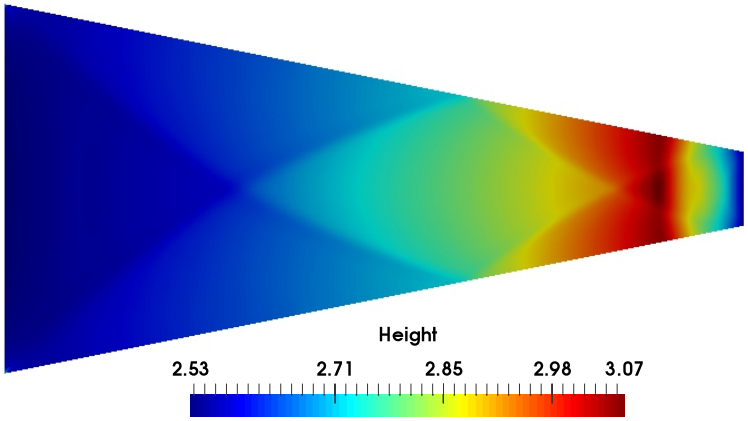
Figure 4.8: The discretized computational domain of example 3. There are 64 divisions in the vertical direction, and in the horizontal direction, we have 64 elements in each of the intervals $x \in (-1, 0)$, $x \in (0, 0.5)$, and $x \in (0.5, 1)$.

y-direction for the same problem at the same time steps.

Overall, based on the presented results, each time step in the explicit approach is faster and scales better by increasing the number of computing cores. On the other hand, the time step size in the implicit approach can be increased significantly, and hence result in a good performance, especially when the evolution of the solution happens slowly. However, by increasing the time step size, there is a chance that we miss important processes in the flow. In practice, both of these methods can have their own applications. While, the lower memory usage of the explicit method and the local nature of its computation results in a better scalability of this approach for large problems, the higher stability of implicit method can be utilized for a more efficient time integration procedure.

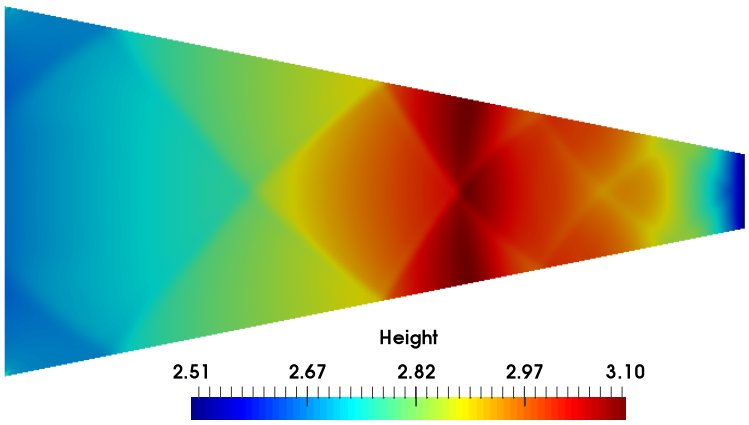


(a) $t = 1.0$

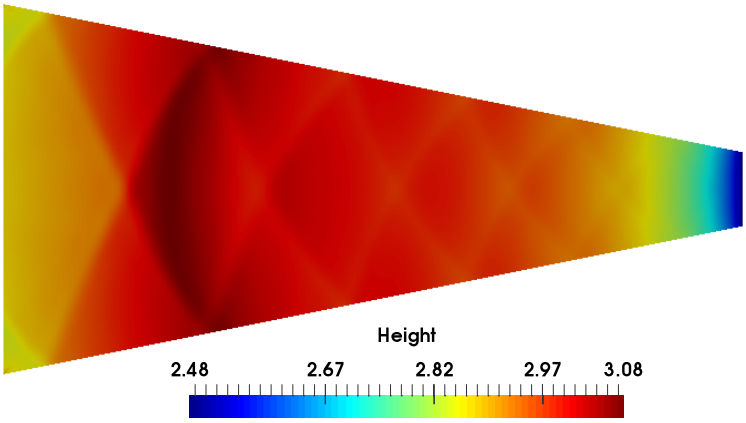


(b) $t = 2.7$

Figure 4.9: Water height in example 3, at different time steps.

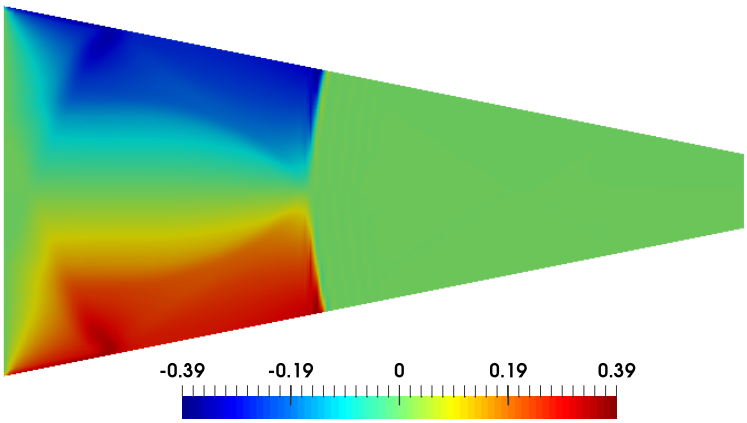


(c) $t = 3.5$

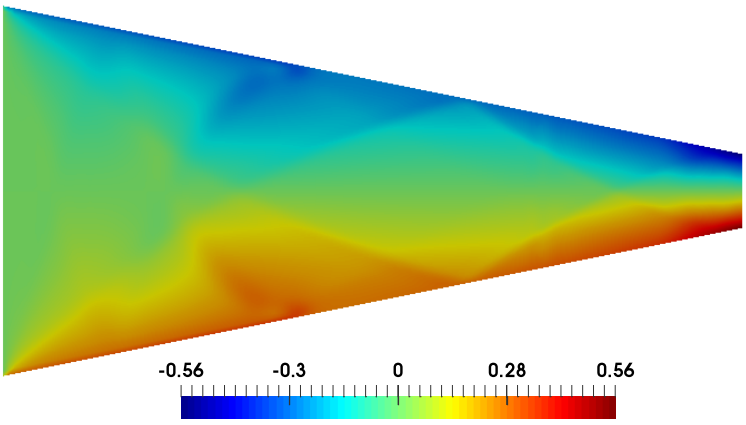


(d) $t = 4.5$

Figure 4.9: (cont'd) Water height in example 3, at different time steps.

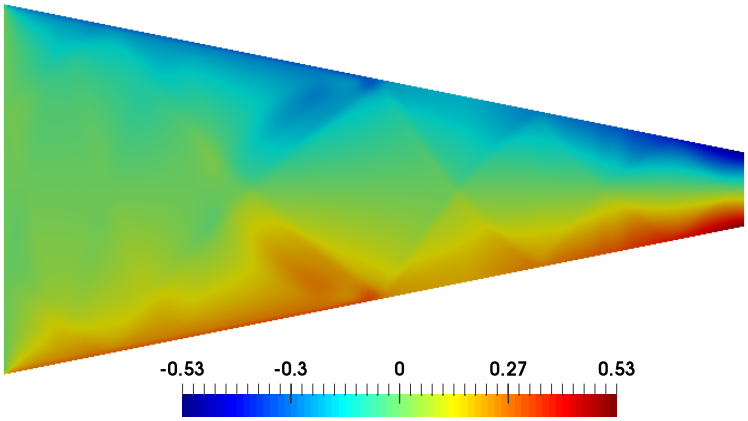


(a) $t = 1.0$

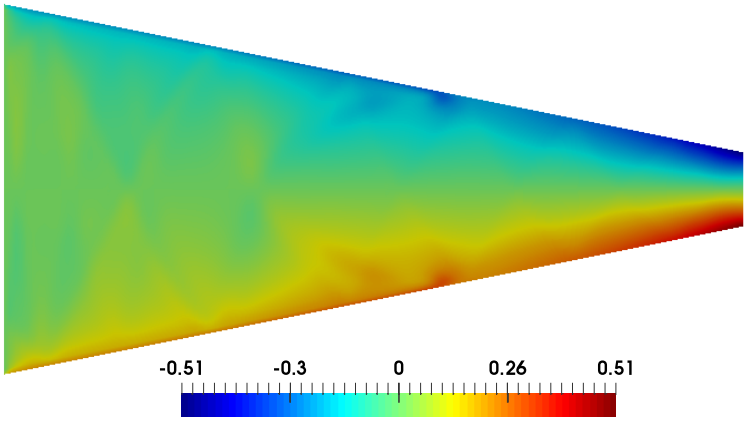


(b) $t = 2.7$

Figure 4.10: Velocity in y-direction in example 3, at different time steps.



(c) $t = 3.5$



(d) $t = 4.5$

Figure 4.10: (cont'd) Velocity in y-direction in example 3, at different time steps.

Chapter 5

Solving Green–Naghdi Equation Using Hybridized Discontinuous Galerkin Method

In this chapter we introduce a hybridized discontinuous Galerkin method for the Green-Naghdi equation. We explained the derivation procedure of this equation in Chapter 2, and here we apply a minor change on the equation to improve its dispersive properties. We then use an operator splitting technique to split the equation to a hyperbolic part and a dispersive part. The hyperbolic part will be solved using the explicit technique that we introduced in Chapter 4. Here, the solution to the dispersive part will be explained in more detail. Let us first recall the G–N equation (2.54) in the nondimensionalized form:

$$\begin{cases} \partial_t \zeta + \nabla \cdot (h\bar{\mathbf{u}}) = 0, \\ (I + \mu T)\left(\partial_t \bar{\mathbf{u}} + \varepsilon(\bar{\mathbf{u}} \cdot \nabla)\bar{\mathbf{u}}\right) + \nabla\zeta + \varepsilon\mu Q_1(\bar{\mathbf{u}}) = 0. \end{cases} \tag{5.1}$$

Here, we have dropped primes, despite all variables and operators being nondimensional. Meanwhile, operators T, Q_1 are defined as:

$$T(\mathbf{w}) = R_1(\nabla \cdot \mathbf{w}) + \beta R_2(\nabla b \cdot \mathbf{w}), \tag{5.2}$$

$$Q_1(\mathbf{w}) = -2R_1\left(\partial_x \mathbf{w} \cdot \partial_y \mathbf{w}^{\perp} + (\nabla \cdot \mathbf{w})^2\right) + \beta R_2\left(\mathbf{w} \cdot (\mathbf{w} \cdot \nabla)\nabla b\right), \tag{5.3}$$

where, $\mathbf{w}^{\perp} = (-w_2, w_1)$, and:

$$R_1(w) = -\frac{1}{3h}\nabla(h^3 w) - \beta\frac{h}{2}w\nabla b, \tag{5.4}$$

$$R_2(w) = \frac{1}{2h}\nabla(h^2 w) + \beta w\nabla b. \tag{5.5}$$

Although Eq. (5.1) is a dispersive equation, there are modified versions of this equation which can offer better dispersive properties [48]. The main idea here is to add some terms of order $O(\mu^2)$, such that the approximation order of the equations are not affected, but the equations will be appropriate for a wider range of μ. To this end, based on the second equation of (5.1), we have:

$$\partial_t \mathbf{u} = -\nabla\zeta - \varepsilon(\mathbf{u}\cdot\nabla)\mathbf{u} + O(\mu)$$

Now, given $\alpha \in \mathbb{R}$, we have:

$$\partial_t \mathbf{u} = \alpha\partial_t\mathbf{u} + (1-\alpha)\partial_t\mathbf{u} = \alpha\partial_t\mathbf{u} - (1-\alpha)\left(\nabla\zeta + \varepsilon(\mathbf{u}\cdot\nabla)\mathbf{u}\right) + O(\mu)$$

Substituting $\partial_t\mathbf{u}$ from above into equation (5.1) and dropping all terms of order μ^2 or higher, we will have:

$$\begin{cases} \partial_t\zeta + \nabla\cdot(h\bar{\mathbf{u}}) = 0, \\ (I+\mu\alpha T)\left(\partial_t\bar{\mathbf{u}} + \varepsilon(\bar{\mathbf{u}}\cdot\nabla)\bar{\mathbf{u}}\right) + (I-\mu(1-\alpha)T)\nabla\zeta + \varepsilon\mu Q_1(\bar{\mathbf{u}}) = 0. \end{cases} \tag{5.6}$$

We rewrite the term $(I-\mu(1-\alpha)T)\nabla\zeta$ as:

$$\begin{aligned} (I-\mu(1-\alpha)T)\nabla\zeta &= (I+\mu\alpha T)\nabla\zeta - \mu T\nabla\zeta \\ &= \frac{\nabla\zeta - \nabla\zeta + \alpha(I+\mu\alpha T)\nabla\zeta - \mu\alpha T\nabla\zeta}{\alpha} \\ &= \frac{1}{\alpha}\nabla\zeta + \frac{\alpha-1}{\alpha}(I+\mu\alpha T)\nabla\zeta. \end{aligned}$$

Next, we will replace the variables $(\zeta, \mathbf{u})$ with the conserved variables $(h, h\mathbf{u})$. In this regard, we use $\partial_t \mathbf{u} = \frac{1}{h}\partial_t(h\mathbf{u}) + \frac{\varepsilon}{h}\nabla \cdot (h\mathbf{u})\mathbf{u}$, and $\nabla \cdot (h\mathbf{u} \otimes \mathbf{u}) = \nabla \cdot (h\mathbf{u})\mathbf{u} + h(\mathbf{u} \cdot \nabla)\mathbf{u}$ to get:

$$\begin{cases} \partial_t h + \varepsilon \nabla \cdot (h\bar{\mathbf{u}}) = 0, \\ \partial_t (h\bar{\mathbf{u}}) + \varepsilon \nabla \cdot (h\bar{\mathbf{u}} \otimes \bar{\mathbf{u}}) + \dfrac{\alpha - 1}{\alpha} h \nabla \zeta \\ \qquad\qquad + \left(I + \mu\alpha h T_{\frac{1}{h}}\right)^{-1} \left[\dfrac{1}{\alpha} h \nabla \zeta + \varepsilon \mu h Q_1(\bar{\mathbf{u}})\right] = 0. \end{cases} \tag{5.7}$$

Finally, we write the equations in the dimensionalized form:

$$\begin{cases} \partial_t h + \nabla \cdot (h\bar{\mathbf{u}}) = 0, \\ \partial_t (h\bar{\mathbf{u}}) + \nabla \cdot (h\bar{\mathbf{u}} \otimes \bar{\mathbf{u}}) + \dfrac{\alpha - 1}{\alpha} g h \nabla \zeta \\ \qquad\qquad + \left(I + \alpha h T_{\frac{1}{h}}\right)^{-1} \left[\dfrac{1}{\alpha} g h \nabla \zeta + h Q_1(\bar{\mathbf{u}})\right] = 0. \end{cases} \tag{5.8}$$

5.1 Dispersive Properties of the Modified G–N Equation

Before we continue to the solution of Eq. (5.8), we need to explain the effect of α on the dispersive properties of this equation. Let us refer to (2.6), and linearize it by setting $\beta = \epsilon = 0$, and $\mathbf{u} = 0$. Hence, h (nondimensionalized water depth) is equal to 1, and $Q_1(\mathbf{u}) = 0$. Moreover, $T(\partial_t \mathbf{u}) = -(1/3)\nabla(\nabla \cdot (\partial_t \mathbf{u}))$. Thus, the linearized version of equation (5.6) reads as:

$$\begin{cases} \partial_t \zeta + \nabla \cdot \mathbf{u} = 0, \\ \partial_t \mathbf{u} - \dfrac{\mu\alpha}{3} \nabla\nabla \cdot \partial_t \mathbf{u} + \nabla\zeta + \mu \dfrac{1 - \alpha}{3} \nabla\nabla \cdot \nabla\zeta = 0. \end{cases} \tag{5.9}$$

The dimensionalized equation, corresponding to (5.9) finds the following form:

$$\begin{cases} \partial_t \zeta + H_0 \nabla \cdot \mathbf{u} = 0, \\ \partial_t \mathbf{u} - \dfrac{\alpha}{3} H_0^2 \nabla\nabla \cdot \partial_t \mathbf{u} + g\nabla\zeta + \dfrac{1-\alpha}{3} g H_0^2 \nabla\nabla \cdot \nabla\zeta = 0. \end{cases} \tag{5.10}$$

By taking Fourier transform of this equation with respect to the horizontal variable and time we obtain the corresponding dispersion relation. A less intricate approach for this purpose is to use $(\zeta, \mathbf{u}) = (\hat{\zeta}, \hat{\mathbf{u}}) e^{i(\xi\cdot\mathbf{x}-\omega t)}$ as the solution and obtain the following relation:

$$\omega(\xi) = |\xi|\sqrt{gH_0}\sqrt{\frac{1+\frac{\alpha-1}{3}(|\xi|H_0)^2}{1+\frac{\alpha}{3}(|\xi|H_0)^2}} \tag{5.11}$$

Now, we want to compute the phase and group velocities based on this dispersion relation. One can easily check that:

$$c_P^{\mathrm{GN}}(|\xi|H_0) = \sqrt{gH_0}\sqrt{\frac{1+\frac{\alpha-1}{3}(|\xi|H_0)^2}{1+\frac{\alpha}{3}(|\xi|H_0)^2}} \tag{5.12}$$

$$c_G^{\mathrm{GN}}(|\xi|H_0) = \frac{9+6(a-1)(kH_0)^2+a(a-1)(kH_0)^4}{(3+(a-1)(kH_0)^2)^{1/2}\,(3+a(kH_0)^2)^{3/2}} \tag{5.13}$$

These relations are an approximation to the phase and group velocities that we derived in Chapter 2 (refer to Eqs. (2.19), (2.21)). In Figs. 5.1 and 5.2 we have plotted the ratios $\frac{c_P^{\mathrm{GN}}(|\xi|H_0)}{c_P(|\xi|H_0)}$ and $\frac{c_G^{\mathrm{GN}}(|\xi|H_0)}{c_G(|\xi|H_0)}$ for $|\xi|H_0 \in [0,4]$. Based on these graphs and the range of wavenumbers that we have to resolve in a problem, we can choose the proper value for α to make our model more dependable in practical applications.

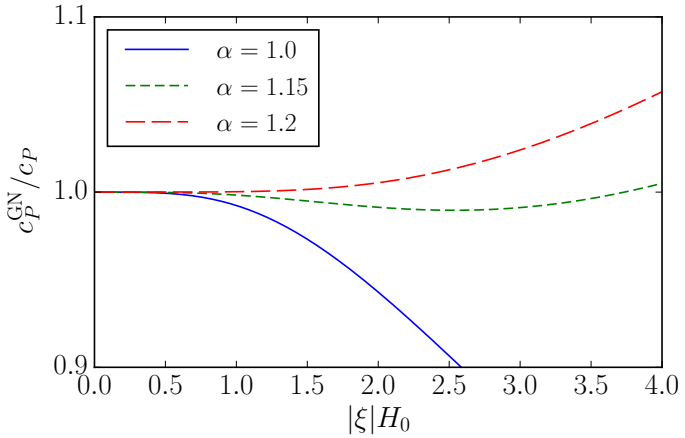


Figure 5.1: The ratio of approximate phase velocity based on the modified G–N equation to the exact linearized wave model, for different values of α.

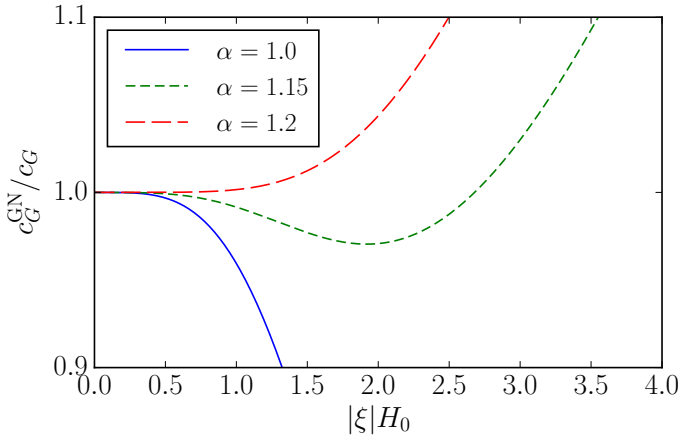


Figure 5.2: The ratio of approximate group velocity based on the modified G–N equation to the exact linearized wave model, for different values of α.

5.2 Solution Approach

Here we introduce an operator splitting approach which will be used to solve Eq. (5.8). The splitting method that we use here is widely referred to as *Strang splitting* [66]. This splitting is known to be second order accurate if each of its components are at least second order accurate.

In order to apply this method, we first consider $\mathcal{S}_1$ as the solution operator associated with hyperbolic part of (5.8):

$$\begin{cases} \partial_t h + \nabla \cdot (h\mathbf{u}) = 0, \\ \partial_t (h\mathbf{u}) + \nabla(\frac{1}{2} g h^2) + \nabla \cdot (h\mathbf{u} \otimes \mathbf{u}) + gh\nabla b = 0. \end{cases} \tag{5.14}$$

This operator takes the solution at the previous time level and computes the evolution of the solution during the current time step. Moreover, $\mathcal{S}_2$ is the solution operator for the dispersive part:

$$\begin{cases} \partial_t h = 0, \\ \partial_t (h\mathbf{u}) - \frac{1}{\alpha} gh\nabla\zeta + (1 + \alpha h T \frac{1}{h})^{-1} \left[\frac{1}{\alpha} gh\nabla\zeta + hQ_1(\mathbf{u}) \right] = 0. \end{cases} \tag{5.15}$$

Then the Strang splitting suggests that the solution operator corresponding to system (5.8) is:

$$\mathcal{S}(\Delta t) = \mathcal{S}_1(\Delta t/2)\mathcal{S}_2(\Delta t)\mathcal{S}_1(\Delta t/2). \tag{5.16}$$

Thus, we start our computation by solving Eq. (5.14), and use the solution obtained from this equation at time $\Delta t/2$, as the initial condition for Eq. (5.15). Then we use the solution of (5.15) at time Δt as the initial condition for (5.14) and solve this equation to obtain the solution at time Δt. We continue this process in the next time steps. A graphical representation of the employed technique is shown in Fig. 5.3.

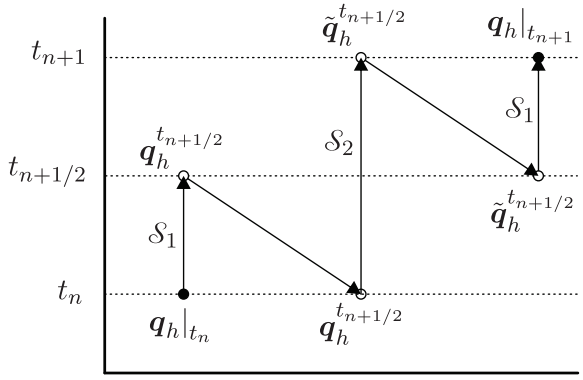


Figure 5.3: The splitting technique used to solve the coupling between the hyperbolic and dispersive sub-problems. We start with $\boldsymbol{q}_h|_{t_n}$, and obtain $\boldsymbol{q}_h|_{t_{n+1}}$ at the end of the time step.

Since the second step in this splitting, takes into account the dispersive terms in the Green–Naghdi equation, in some references [33], this step has been called the *dispersive correction step*. The significance of this naming is that by turning this correction off, we can reduce the computational cost of the dispersive terms in those parts of the domain where these effects can be neglected. For example, as we mentioned in Chapter 2, for ocean modeling applications, near the coast, the value of μ is large enough to consider using $O(\mu^2)$ models. Hence, using the dispersive correction is an apt choice. However, away from the coast, the values of μ are small and an $O(\mu)$ model (such as Saint-Venant wave equation) can give us the required precision.

The solution to system (5.14) was explained in Chapter 4, and here, we only discuss the solution to system (5.15). To this end, we are looking for

$(h, h\mathbf{u})$ that solves the following equation:

$$\begin{cases} \partial_t h = 0, \\ \partial_t (h\mathbf{u}) - \frac{1}{\alpha} gh\nabla\zeta + \mathbf{w}_1 = 0, \end{cases} \tag{5.17}$$

where $\mathbf{w}_1$ is obtained using:

$$(1 + \alpha h T \frac{1}{h})\mathbf{w}_1 = \frac{1}{\alpha} gh\nabla\zeta + hQ_1(\mathbf{u}). \tag{5.18}$$

Using definition (5.2) with $\beta = 1$, the above equation finds the following form:

$$\begin{aligned} \mathbf{w}_1 + \alpha h T\left(\tfrac{1}{h}\mathbf{w}_1\right) = \mathbf{w}_1 - \frac{\alpha}{3}\nabla\left(h^3\nabla\cdot\left(\tfrac{1}{h}\mathbf{w}_1\right)\right) - \frac{\alpha h^2}{2}\nabla\cdot\left(\tfrac{1}{h}\mathbf{w}_1\right)\nabla b \\ + \frac{\alpha}{2}\nabla\left(h\nabla b\cdot\mathbf{w}_1\right) + \alpha\nabla b\cdot\mathbf{w}_1\nabla b. \end{aligned} \tag{5.19}$$

We also expand the operator $Q_1(\mathbf{u})$ in the right hand side of (5.18) as follows:

$$\begin{aligned} hQ_1(\mathbf{u}) =& \frac{2}{3}\nabla\left(h^3\partial_x\mathbf{u}\cdot\partial_y\mathbf{u}^\perp + h^3(\nabla\cdot\mathbf{u})^2\right) + \frac{1}{2}\nabla\left(h^2\mathbf{u}\cdot(\mathbf{u}\cdot\nabla)\nabla b\right) \\ &+ h^2\left(\partial_x\mathbf{u}\cdot\partial_y\mathbf{u}^\perp + (\nabla\cdot\mathbf{u})^2\right)\nabla b + h\left(\mathbf{u}\cdot(\mathbf{u}\cdot\nabla)\nabla b\right)\nabla b. \end{aligned} \tag{5.20}$$

It should be noted that in order to compute $Q_1(\mathbf{u})$, one needs to obtain the second derivatives of $\mathbf{u}$ in the domain of each element. As will be explained later, we use a local discontinuous Galerkin approximation to $\nabla\mathbf{u}$ and $\nabla\nabla\mathbf{u}$ to compute the high order derivatives of $\mathbf{u}$ inside $Q_1(\mathbf{u})$.

Based on the above relations, (5.18) can be written as a system of first order equations:

$$\begin{cases} \nabla\cdot(\frac{1}{h}\mathbf{w}_1) - h^{-3}w_2 = 0, \\ \mathbf{w}_1 - \frac{\alpha}{3}\nabla(w_2) - \frac{\alpha}{2h}w_2\nabla b + \frac{\alpha}{2}\nabla(h\nabla b\cdot\mathbf{w}_1) \\ \qquad\qquad + \alpha\mathbf{w}_1\nabla b\otimes\nabla b = \frac{1}{\alpha}gh\nabla\zeta + hQ_1(\mathbf{u}). \end{cases} \tag{5.21}$$

We are going to use an explicit method to solve (5.17). Hence, in the process of solving (5.21), h and $\mathbf{u}$ are known. However, the assumption that $h \geq h_{\min}$ with $h_{\min}$ being a uniform positive constant should be taken into account.

An important feature of Eq. (5.18) is the regularization effect of this equation on the solution [41]. One can realize this fact, by comparing the effect of the term $gh\nabla\zeta$ in Eqs. (5.17) and (5.18). This term can significantly affect the momentum equation in NSWE and is usually in charge of the development of sharp features in this equation. Since, in the dispersive correction, we apply the operator $\mathcal{S}_2$ after each hyperbolic solve, we actually remove the sharp features which might be developed by $gh\nabla\zeta$ and replace them with $\mathbf{w}_1$, which is the solution to a globally solved equation. This property improves the stability of the numerical method and diffuses some of the features that will develop in the solution due to the nonlinear hyperbolic part, i.e. Eq. (5.14).

5.3 Variational Formulation

In this section we use the hybridized DG method to solve Eq. (5.17). To this end, we refer to the functional settings introduced in Section 4.1.2, and define an additional space $\bar{\mathbf{M}}_h^p$ as follows:

$$\bar{\mathbf{M}}_h^p \coloneqq \{\mu \in \left(L^2(\mathcal{E}_h)\right)^d : \mu|_e \in (\mathcal{Q}^p(e))^d \quad \forall e \in \mathcal{E}_h\}. \tag{5.22}$$

Now, we want to find $(\mathbf{w}_{1h}, w_{2h}) \in \mathbf{V}_h^p$, and $\hat{\mathbf{w}}_{1h} \in \bar{\mathbf{M}}_h^p$ such that:

$$\begin{cases} (h^{-3}\, w_{2h}, p_2) - \left\langle \hat{h}^{-1}\, \hat{\mathbf{w}}_{1h} \cdot \mathbf{n}, p_2 \right\rangle + (h^{-1}\, \mathbf{w}_{1h}, \nabla p_2) = 0. \\ (\mathbf{w}_{1h}, \mathbf{p}_1) - \frac{\alpha}{3} \left\langle \mathbf{w}_{2h}^* \cdot \mathbf{n}, \mathbf{p}_1 \right\rangle + \frac{\alpha}{3} (w_{2h}, \nabla \cdot \mathbf{p}_1) - \frac{\alpha}{2} \left(\frac{1}{h} \nabla b\, w_{2h}, \mathbf{p}_1 \right) \\ \qquad + \frac{\alpha}{2} \left\langle \hat{h} \nabla b \cdot \hat{\mathbf{w}}_{1h}, \mathbf{p}_1 \cdot \mathbf{n} \right\rangle - \frac{\alpha}{2} (h \nabla b \cdot \mathbf{w}_{1h}, \nabla \cdot \mathbf{p}_1) \\ \qquad + \alpha\, (\nabla b \otimes \nabla b\, \mathbf{w}_{1h}, \mathbf{p}_1) = l_{01}(\mathbf{p}_1), \end{cases} \tag{5.23}$$

for all $(\mathbf{p}_1, p_2) \in \mathbf{V}_h^p$. Here, the definition of $l_{01}(\mathbf{p}_1)$ can be inferred by comparing the above system with (5.21); moreover, the numerical flux $\mathbf{w}_{2h}^* \cdot \mathbf{n}$ is defined as:

$$\mathbf{w}_{2h}^* \cdot \mathbf{n} = w_{2h} \mathbf{I} \cdot \mathbf{n} + \boldsymbol{\tau} \left(\mathbf{w}_{1h} - \hat{\mathbf{w}}_{1h} \right), \tag{5.24}$$

where $\mathbf{I}$ is the $d \times d$ identity matrix and $\boldsymbol{\tau}$ is the stabilization parameter matrix. We will use a constant and uniform diagonal matrix for this purpose.

Next, we define the following bilinear forms and functionals:

$$\begin{gathered} a_{02}(w_{2h}, p_2) = (h^{-3} w_{2h}, p_2); \quad b_{01}^T(\mathbf{w}_{1h}, p_2) = (h^{-1} \mathbf{w}_{1h}, \nabla p_2); \\ c_{01}(\hat{\mathbf{w}}_1, p_2) = \langle \hat{h}^{-1} \hat{\mathbf{w}}_{1h} \cdot \mathbf{n}, p_2 \rangle; \quad b_{02}(w_{2h}, \mathbf{p}_1) = (\nabla w_{2h}, \mathbf{p}_1); \\ a_{01}(\mathbf{w}_{1h}, \mathbf{p}_1) = (\mathbf{w}_{1h}, \mathbf{p}_1) + \alpha\, (\nabla b \otimes \nabla b\, \mathbf{w}_{1h}, \mathbf{p}_1)\,; \\ d_{01}(\mathbf{w}_{1h}, \mathbf{p}_1) = \langle \tau \mathbf{w}_{1h}, \mathbf{p}_1 \rangle; \quad b_{03}^T(\mathbf{w}_{1h}, \mathbf{p}_1) = (h \nabla b \cdot \mathbf{w}_{1h}, \nabla \cdot \mathbf{p}_1) \\ a_{03}(w_{2h}, \mathbf{p}_1) = \left(\tfrac{1}{h} \nabla b\, w_{2h}, \mathbf{p}_1 \right); \\ c_{02}(\hat{\mathbf{w}}_{1h}, \mathbf{p}_1) = \langle \tau \hat{\mathbf{w}}_{1h}, \mathbf{p}_1 \rangle + \frac{3}{2} \left\langle \hat{h} \nabla b \cdot \hat{\mathbf{w}}_{1h}, \mathbf{p}_1 \cdot \mathbf{n} \right\rangle. \end{gathered} \tag{5.25}$$

We are now able to write Eq. (5.23) as:

$$\begin{cases} A_{02} w_{2h} + B_{01}^T \mathbf{w}_{1h} - C_{01} \hat{\mathbf{w}}_{1h} = 0 \\ \left(A_{01} - \frac{\alpha}{2} B_{03}^T - \frac{\alpha}{3} D_{01} \right) \mathbf{w}_{1h} - \left(\frac{\alpha}{2} A_{03} + \frac{\alpha}{3} B_{02} \right) w_{2h} + \frac{\alpha}{3} C_{02} \hat{\mathbf{w}}_{1h} = L_{01} \end{cases} \tag{5.26a}$$

Finally, we also require that the numerical flux be conserved across element edges. In other words, we have:

$$\langle \mathbf{w}_{2h}^* \cdot \mathbf{n}, \mu \rangle_{\partial \mathcal{T}_h \setminus \partial\Omega} + \langle \mathcal{B}_h, \mu \rangle_{\partial \mathcal{T}_h \cap \partial\Omega} = 0, \tag{5.26b}$$

for all $\mu \in \bar{\mathbf{M}}_h^p$. Here $\mathcal{B}_h$ is the boundary operator, which can be defined based on the applied boundary conditions.

5.3.1 Boundary Conditions

In this study, the following types of boundary conditions have been applied through the operator $\mathcal{B}_h$:

- Periodic boundary condition: In this case, we couple $\hat{\mathbf{w}}_{1h}$ on the two periodic boundaries and also set the incoming and outgoing fluxes from these boundaries equal to each other. Hence, $\mathcal{B}_h = \mathbf{w}_{2h}^* \cdot \mathbf{n}$, which satisfies the conservation of the numerical flux $\mathbf{w}_{2h}^* \cdot \mathbf{n}$ across the periodic boundaries. This also means that the periodic boundary will be treated as if it is a boundary between two elements inside the domain.

- Solid wall boundary: At the solid wall, we simply set $\hat{\mathbf{w}}_{1h} \cdot \mathbf{n} = 0$. However, this condition does not set $\hat{\mathbf{w}}_{1h}$ in the tangent direction to the wall. In the tangent direction, we set $\hat{\mathbf{w}}_{1h} \cdot \mathbf{t}$ based on the projection of $\mathbf{w}_{1h} \cdot \mathbf{t}$ onto $\bar{\mathbf{M}}_h^p$ at the corresponding face ($\mathbf{t}$ being the tangent vector to the solid wall). This means, we take the tangential component of $\hat{\mathbf{w}}_{1h}$ from the tangential component of $\mathbf{w}_{1h}$. We include the conditions on

normal and tangential directions of $\hat{\mathbf{w}}_{1h}$ using the following definition of $\mathcal{B}_h$ in (5.26b):

$$\mathcal{B}_h = \mathbf{w}_{1h} - (\mathbf{w}_{1h} \cdot \mathbf{n})\mathbf{n} - \hat{\mathbf{w}}_{1h}. \tag{5.27}$$

One can simply check that $\mathcal{B}_h \cdot \mathbf{n} = \hat{\mathbf{w}}_{1h} \cdot \mathbf{n}$, and $\mathcal{B}_h \cdot \mathbf{t} = \hat{\mathbf{w}}_{1h} \cdot \mathbf{t} - \mathbf{w}_{1h} \cdot \mathbf{t}$.

We will use the above boundary conditions in the examples presented below. Here, we also need inflow boundary conditions. To apply these types of boundary condition, we simply use a region with no dispersive correction. Hence, we simply solve NSWE in that region to be able to apply more sophisticated boundary conditions. As a result, there is no need for including those types of boundary conditions directly in the formulation of $\mathcal{B}_h$ in (5.26b).

5.3.2 Computation of Higher Order Derivatives in $Q_1(\mathbf{u})$

Solving equation (5.18) involves computation of the 1st and 2nd order derivatives of the velocity vector. Among all other terms, we need to compute the term $\nabla \left(h^3 \partial_x \mathbf{u} \cdot \partial_y \mathbf{u}^\perp + h^3 (\nabla \cdot \mathbf{u})^2 \right)$ in each element. If this computation is performed in a local manner in each element independent of the others, we lose a significant order of accuracy. It can be easily checked that by computing this term locally, our solution will not converge for elements with first order polynomial approximation. On the other hand, since we use this term in our weak formulation, one might consider using the integration by parts technique to transfer the gradient of the term in parentheses to the test function, and replace their flux with a proper numerical flux. However, finding such a flux formulation for the extremely nonlinear terms like $(\nabla \cdot \mathbf{u})^2$ or $\partial_x \mathbf{u} \cdot \partial_y \mathbf{u}^\perp$ is not

straightforward. Therefore, in this study we use a local discontinuous Galerkin technique to obtain approximations to $\nabla\mathbf{u}$ and $\nabla\nabla\mathbf{u}$. It is worthwhile to note that, $\nabla\mathbf{u}$ is a 2-tensor and $\nabla\nabla\mathbf{u}$ is a 3-tensor; As a result, we switch to index notation for clarity. We use u_i to denote the components of $\mathbf{u}$ and define the tensors r_{ij} (which contains the components of $\nabla\mathbf{u}$), and s_{ijk} (containing the components of $\nabla\nabla\mathbf{u}$) as follows:

$$r_{ij} - \partial_j u_i = 0, \tag{5.28a}$$

$$s_{ijk} - \partial_k r_{ij} = 0. \tag{5.28b}$$

Next, we write the variational formulations corresponding to these equations in an element ($K \in \mathcal{T}_h$):

$$(r_{ij}, \sigma_{ij})_K = \langle \hat{u}_i, \sigma_{ij} n_j \rangle_{\partial K} - (u_i, \partial_j \sigma_{ij})_K \,, \tag{5.29a}$$

$$(s_{ijk}, \eta_{ijk})_K = \langle \hat{r}_{ij}, \eta_{ijk} n_k \rangle_{\partial K} - (r_{ij}, \partial_k \eta_{ijk})_K \,. \tag{5.29b}$$

In these equations, $\hat{u}_i$ and $\hat{r}_{ij}$ are the numerical fluxes, which should be defined based on the values of u_i and r_{ij} in the two neighboring elements. In this study we will use the centered fluxes [1], i.e. $\hat{u}_i = \{\!\{u_i\}\!\}$, $\hat{r}_{ij} = \{\!\{r_{ij}\}\!\}$.

By using this technique, we can compute the derivatives of $\mathbf{u}$, and substitute them in (5.20) to compute $hQ_1(\mathbf{u})$, and solve the system (5.26a) by an explicit time integration method.

5.4 Numerical Examples

In this section, we present five numerical example, for verification and validation of the presented technique. The purpose of the first example is to

show the convergence properties of the numerical approximation with respect to the element size (Δx) and the polynomial order (p). We also measure the amount of time spent on each phase of the operator splitting to give a sense of the cost of each phase of this procedure. In the second example we consider the amplifying effect of the reflection from a solid wall on the amplitude of a solitary wave. These kinds of simulations are useful in the design of levees and dikes. We compare our numerical results with a number of experimental data from the literature. In the third example, we study the shoaling and reflection of a solitary wave. This example tests the model's dispersive and nonlinear properties before the breaking stage. In the fourth example, we compare our numerical results for a wave traveling on a mild slope with the corresponding experimental observations. In the last example, we solve a two dimensional problem. Even though, we do not compare our results for this example with experimental data, we find our numerical results to be consistent with the observations in 1D experiments. In all of the numerical tests presented here, we use the regular Runge-Kutta time integration technique for each part of the operator splitting. Similar to Chapter 4, we use our software which is written in C++, and uses the libraries deal.II, PETSc, and MUMPS for solving the GN equation.

Example 1: In this example we consider the exact solution to the nonlinear Green-Naghdi equation on a flat bathymetry in one dimension. This solution, which is derived by Serre [63], should match our numerical results

with $\alpha = 1$. This analytical solution is given by:

$$h(t,x) = H_0 + a_0 \operatorname{sech}^2\left(\kappa(x - x_0 - c_0 t)\right), \tag{5.30a}$$

$$h\mathbf{u}(t,x) = c_0 h(t,x) - c_0 H_0, \tag{5.30b}$$

$$\kappa = \frac{\sqrt{3a_0}}{2H_0\sqrt{H_0 + a_0}}, \tag{5.30c}$$

$$c_0 = \sqrt{g(H_0 + a_0)} \tag{5.30d}$$

Here, we consider $H_0 = 0.5$, and two values for a_0/H_0, i.e. 0.05, 0.2. We solve the problem in the domain shown in Fig. 5.4. The domain is a stripe with 20 m length and 0.2 m width, and is oriented with an angle of 30° with respect to the x-axis. The reason for choosing a rotated domain is to include as many nonzero terms as possible in Eq. (5.18). Since we have rotated the domain, the x-coordinate in the analytical solution (5.30) should be replaced by x_1 (refer to Fig. 5.4). At all boundaries we consider solid wall conditions. In our numerical scheme, we assign the initial conditions according to the above $h, h\mathbf{u}$ at $t = 0$, $x_0 = -4$, and let the solitary wave propagate in the positive x-direction (refer to Fig. 5.5).

We compute the errors of the numerical results at time $t = 0.375$ s in the L^2-norm, i.e. $\|\boldsymbol{q} - \boldsymbol{q}_h\|_{L^2}$ with $\boldsymbol{q} = (h, h\mathbf{u})$. Next, we compute the corresponding rates of convergence on a set of successively refined meshes for polynomial orders $p = 0, 1, 2, 3$. The time step size in all of the simulations is chosen such that the CFL condition is satisfied. The corresponding plots for $a_0/H_0 = 0.2, 0.4$ are shown in Fig. 5.6. We can observe that for $a_0/H_0 = 0.2$, the convergence rates are very close to the optimal rates, i.e. $p + 1$, for all

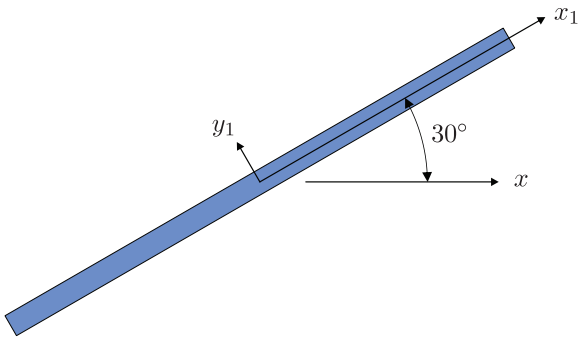


Figure 5.4: Schematic plot of the domain of Example 1. The stripe is 20 m long and 0.2 m wide.

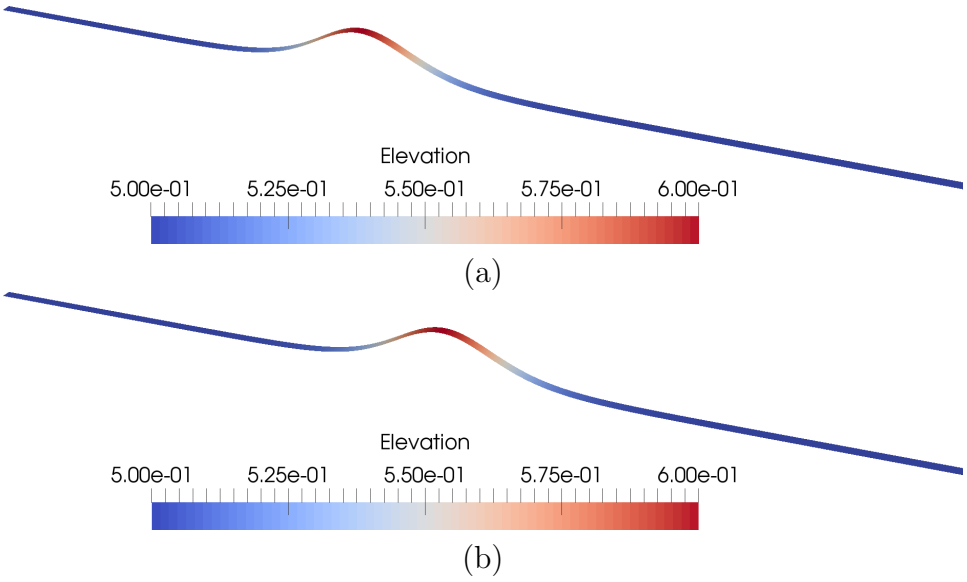
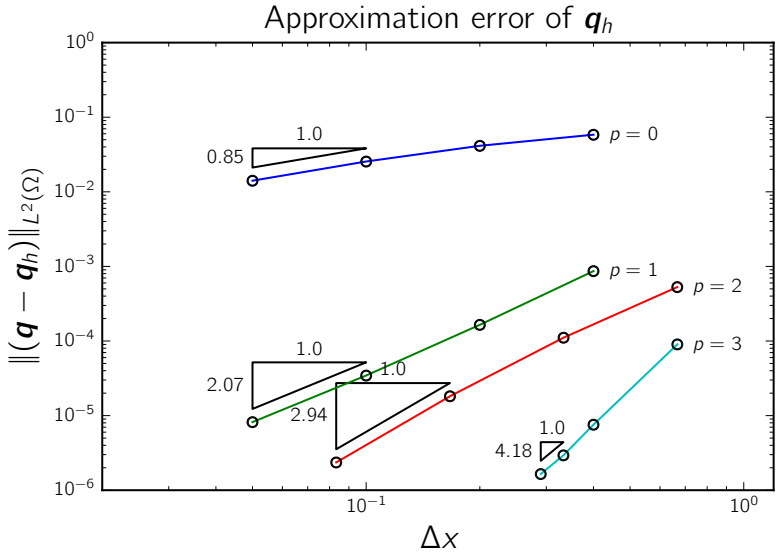


Figure 5.5: Plots of the numerical results of Example 1 with $a_0/H_0 = 0.2$, at times (a): $t = 0$ s and (b): $t = 0.375$ s.

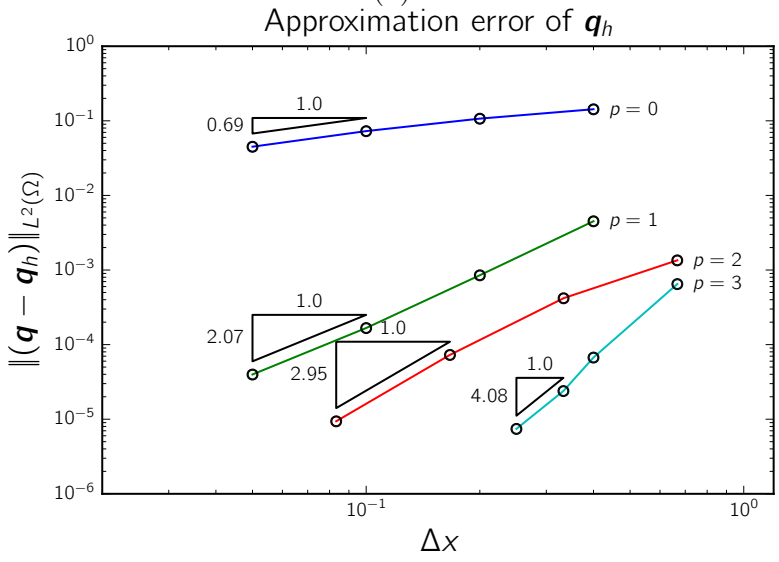
orders of polynomial approximations. An important feature in these plots is the convergence of the results for $p = 0$ with the order 0.85. This is of special interest in practical applications where one needs to use filters to stabilize the numerical results, and the convergence in lowest order approximation can be very helpful [29]. The same observation as above is also true for $a_0/H_0 = 0.4$, except the lower convergence rate for $p = 0$. As a final remark, it should be noted that in this example, the analytical solution of $\mathbf{u}$ is not exactly zero at the two ends of the domain i.e. $x_1 = \pm 10$ m. Hence, the error caused by applying the solid wall becomes the dominant error as we decrease the discretization errors. As a result we cannot achieve errors lower than 10^{-6} in this example.

Finally, from the computational cost point of view, we compare the CPU time of different phases of the computation in our splitting scheme. As mentioned before, our computation at each time step consists of two NSWE solves and one dispersive solve. We have measured computational time of each of these steps on a mesh with 480 elements with $p = 2$, when 4 computational cores on a single machine are utilized. We observe that in each time step, around 52% of the CPU time is taken by the two NSWE solves and 48% is taken by the dispersive solve step. It is worthwhile to mention that these cost shares can vary based on the size of the matrices and the way we employ parallelism.

Example 2: In this example, we compare our numerical results with the experimental data regarding the reflection of a solitary wave from a solid



(a)



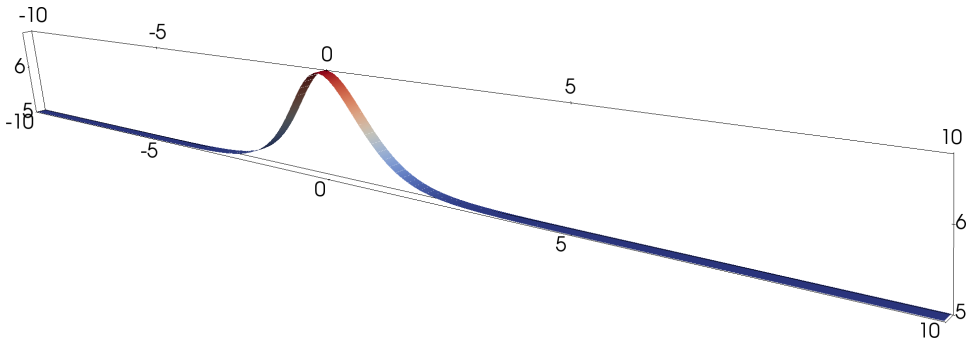
(b)

Figure 5.6: The approximation errors and rates of convergence for different mesh sizes and polynomial orders (a): $a_0/H_0 = 0.2$, (b): $a_0/H_0 = 0.4$.

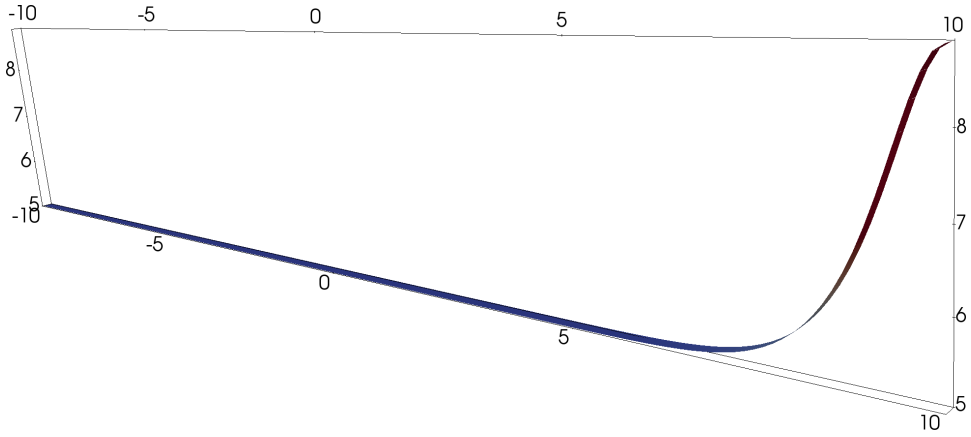
wall [5, 49, 58, 67]. The phase change and the amplification of a nonlinear wave reaching a solid wall is of special interest in the design of levees and dikes.

Here, we consider a similar setup as Example 1, but we let the wave reach the solid wall and report the maximum water elevation that it exhibits at this location. For this test we use a mesh of first order elements with $\Delta x = 0.1$ m. As an example, we show the snapshots of the water surface for a solitary wave with $a_0/H_0 = 0.35$ at two different times in Fig. 5.7. One can observe that the maximum wave height reaches 0.887 m near the wall, which results in $\zeta_{\max}/H_0 = 0.774$. This value is very close to the experimental data for $a_0/H_0 = 0.35$. In Fig. 5.8, the wave amplification factors are computed for multiple values of a_0/H_0 and compared with the experimental data. Moreover, we include the results from linear dispersive wave theory in this graph. One can observe that, for large values of a_0/H_0, the linear wave theory has noticeable errors compared to the Green-Naghdi model. It is also worthwhile to mention that simulating this phenomenon using the nonlinear shallow water equation requires a filtering or limiting process. This process will in turn result in the loss of the peak amplitude, which happens next to the wall.

Example 3: In this example we validate our numerical results against the experimental data regarding the reflection of a solitary wave over a sloping beach [26, 69]. The geometry of this problem is shown in Fig. 5.9. The incident wave does not break prior to touching the wall; however, after the reflection its shape changes dramatically, which requires a fully nonlinear model to capture



(a)



(b)

Figure 5.7: Profile of water height $h = \zeta + H_0$ (measured in 0.1 meter) at two times for the case of $a_0/H_0 = 0.35$; (a): $t = 0.0$, (b): $t = 17.36/\sqrt{g/H_0}$.

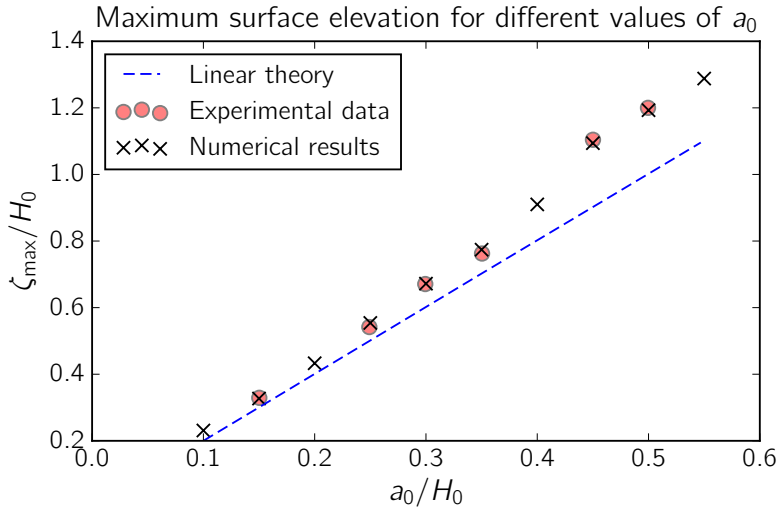


Figure 5.8: Amplification effect of a solid wall on the reflected solitary wave in Example 2.

its behavior.

The numerical model is 40 m long, and the solid wall condition is applied at both ends. The initial water depth is $H_0 = 0.7$ m, and two values are used for the initial wave amplitude, i.e. $a_0 = 7$ cm, and $a_0 = 12$ cm. The wave starts its propagation at $x = 10$ m (refer to Fig. 5.9), and the beach with the slope 1:50 starts at $x = 20$ m. The element size is 8 cm and we use first order elements to discretize the domain. To satisfy the stability criterion, the time step size is taken to be 0.01 s. The time history of water surface elevation at different locations in the domain is available in the literature [69]. Here, we present our results for a reading station located at $x = 37.75$ m.

In Fig. 5.10, we show the snapshots of the water surface rise during the simulation for the initial amplitudes $a_0 = 7$ and 12 cm. In Fig. 5.11 we

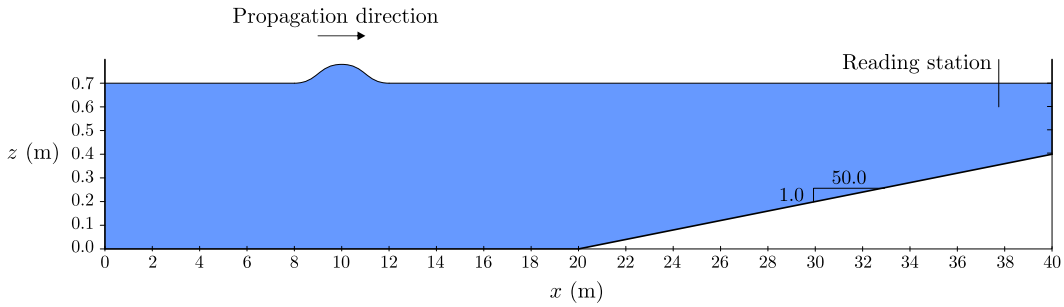
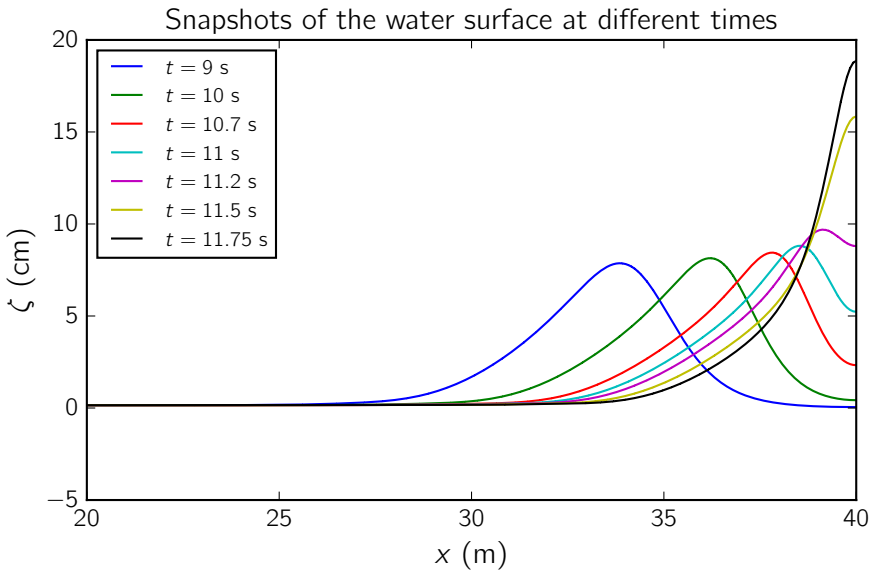


Figure 5.9: The geometry of the numerical model of Example 3.

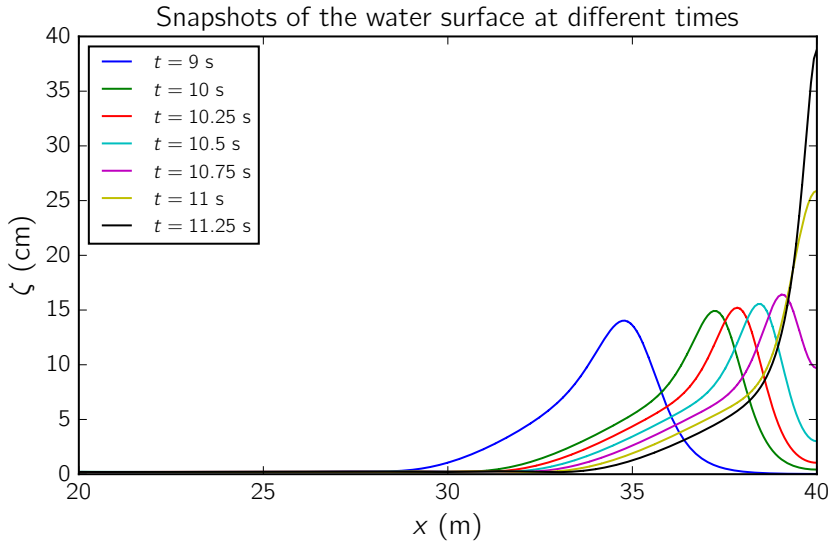
show the time history of the water surface elevation at the reading station with $x = 37.75$ m. The numerical results have been able to capture the peaks in the experimental data quite well; however, as the reflected waves return from the wall, we can observe differences between numerical and experimental results.

Example 4: In this example, we validate our numerical results against experimental data for the nonlinear shoaling of waves propagating over a sloping beach. The sloping beach causes the waves traveling over it to steepen and shoal and finally break. Including the nonlinear shoaling is a required feature for the models used in the coastal ocean simulation. The experimental data for the current test can be found in [4, 27]. The interested reader can also refer to the original reports in French, which were prepared in the Laboratoire des Ècoulements Gèophysiques et Industriels (LEGI) in Grenoble.

As shown in Fig. 5.12, we generate a solitary wave at $x = 0$, and let it propagate towards the sloping beach, which has a mild slope of 1:30. The water height at the flat part of the channel is $H_0 = 25$ cm. We use four different values for the amplitude of the solitary wave, i.e. $a_0/H_0 =$

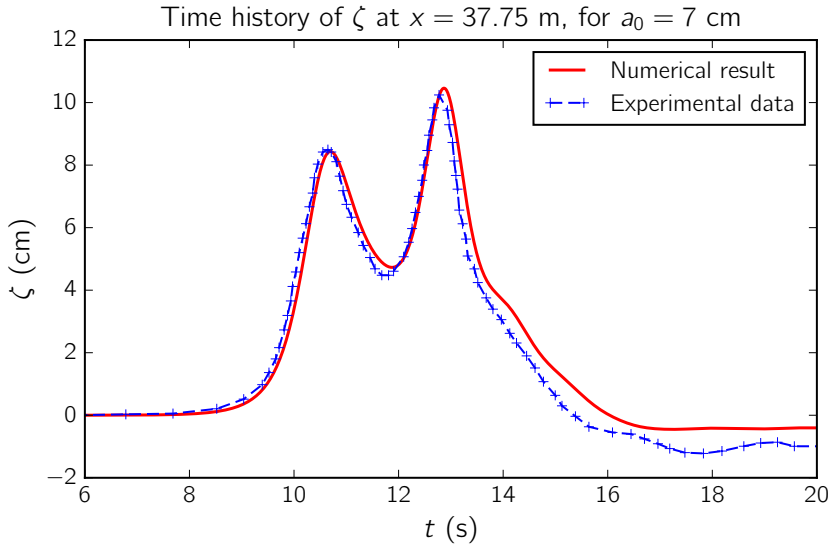


(a) For $a_0 = 7$ cm

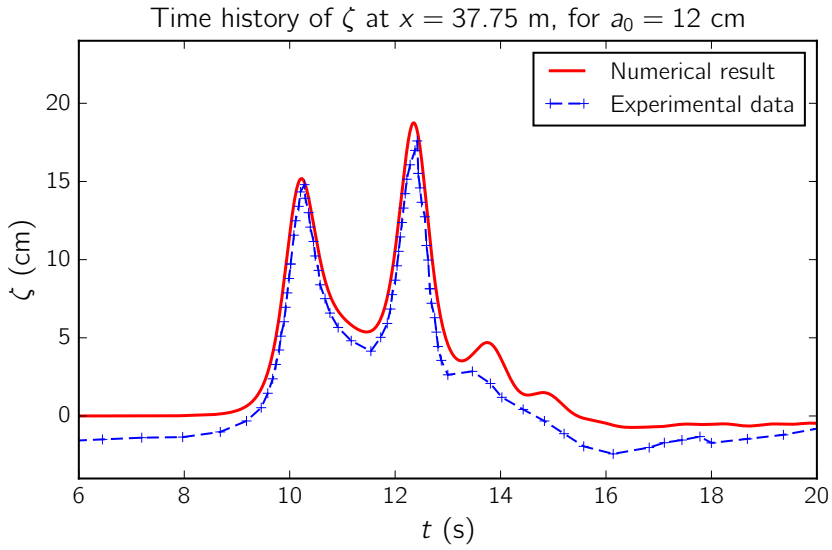


(b) For $a_0 = 12$ cm

Figure 5.10: The snapshots of the water surface (ζ) in Example 3, at different times.



(a) For $a_0 = 7$ cm



(b) For $a_0 = 12$ cm

Figure 5.11: Time history of the water surface at reading station ($x = 37.75$ m) in Example 3.

$0.096, 0.298, 0.456, 0.534$. As the solitary wave travels into the sloping beach, it shoals and gets close to the breaking point. We plot the time-history of the water surface at four stations near the shore and compare it with the data presented in [4].

As for the numerical mesh, we use a 25 m long 2D domain, with 0.2 m width. In the longitudinal direction, we use 420 elements with $\Delta x_{\min} = 0.05$ m and $\Delta x_{\max} = 0.1$ m. The polynomial order of all of the elements is $p = 1$. The time step size is taken $\Delta t = 0.015$ seconds.

The plots of the time history of water surface elevation at five stations near the shore are shown in Fig. 5.13. For moderate nonlinearity parameter, i.e. $a_0/H_0 = 0.298, 0.456$, we can observe a good agreement between the numerical results and the experimental data. However, the unexpected observation in these figures is the noticeable difference in the phase and amplitude of the numerical and experimental results for the case of $a_0/H_0 = 0.096$. Since, both nonlinearity and shallowness parameters are smaller in this case compared to all other cases, we expect to see a good match for $a_0/H_0 = 0.096$. The reason for this discrepancy, which has been also reported in the literature [4] is unknown to us.

Example 5: In this example we solve a two dimensional problem with varying bathymetry. It should be mentioned that, most of the available test cases for the two dimensional experiments require an absorbing boundary condition to be implemented in the numerical solver. Employing such absorbing boundary conditions is a demanding process in a two dimensional problem.

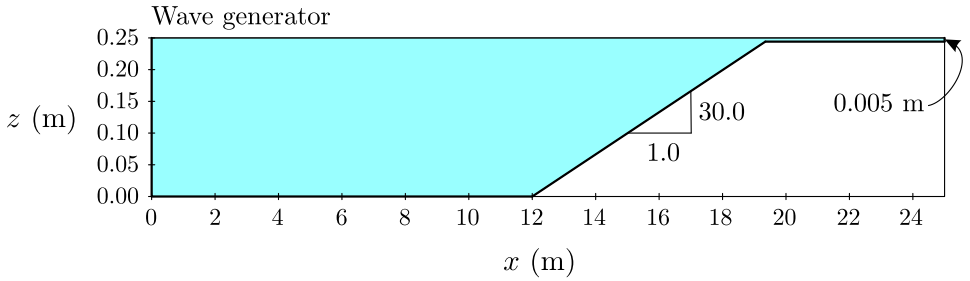


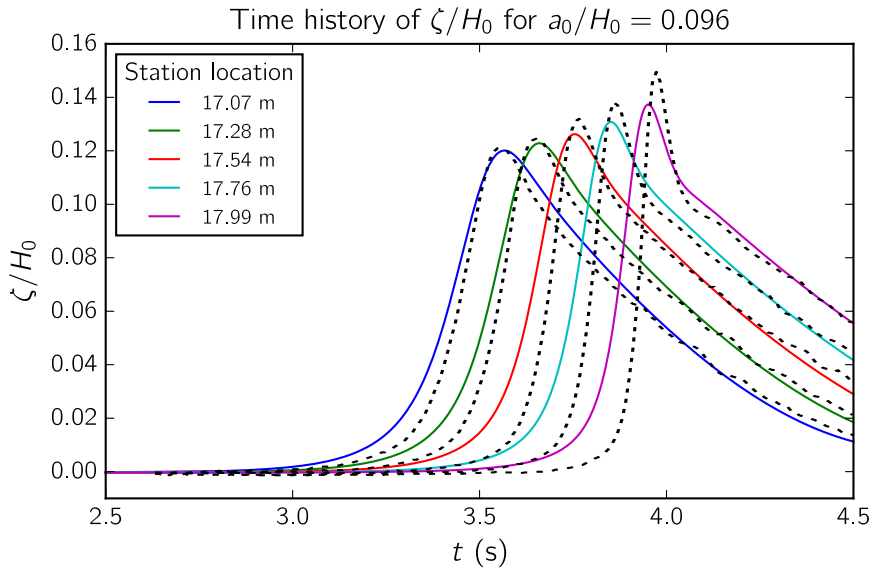
Figure 5.12: The setup of the numerical test of Example 4.

Hence, we only present our numerical results for a sample 2D problem and leave the more sophisticated test problems for later developments.

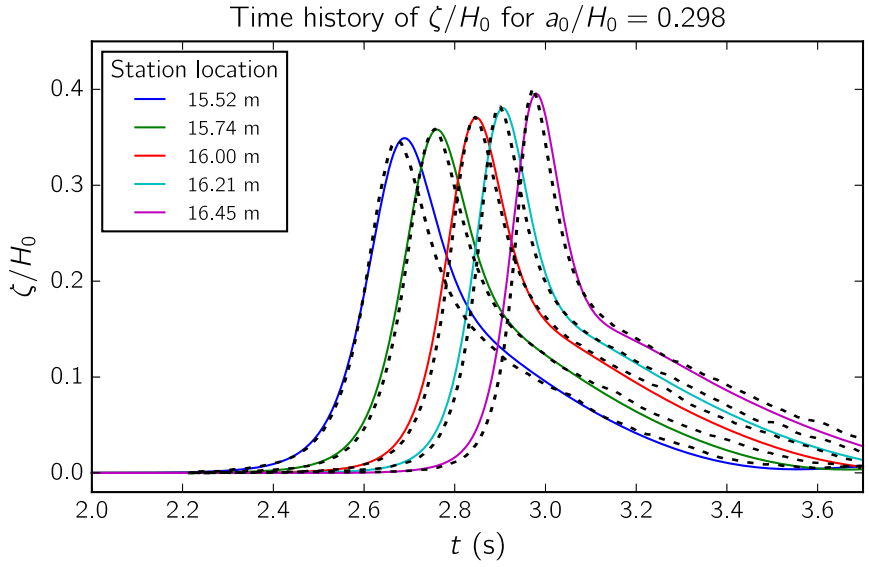
The geometry of the problem, along with the initial water elevation is shown in Fig. 5.14. Here, we have a rectangular domain $\Omega = (-3, 7) \times (-2.5, 2.5)$ m, with solid wall boundary conditions on all of its boundaries. The topography is flat, except a cosine hump which is located inside a circle with radius 2.0 m and centered at $x = 2$ m, $y = 0$ m. The following formula is used for the bathymetry:

$$b = \begin{cases} 0.0 & \sqrt{(x-2)^2 + y^2} > 2.0 \\ \frac{1}{8}\left[1 + \cos\left(\frac{\pi}{2}\sqrt{(x-2)^2 + y^2}\right)\right] & \sqrt{(x-2)^2 + y^2} \leq 2.0 \end{cases}$$

One can observe that by using the above definition for b, the gradient of b in Eq. (5.20) is not zero. Actually, we have to obtain ∇b, $\nabla\nabla b$, and $\nabla\nabla\nabla b$ (which are first, second, and third rank tensors) at the integration points and support them to the solver. We use the solitary wave introduced in Example 1 with $H_0 = 0.3$ m and $a_0 = 0.1$ m, as our initial condition. The domain is divided into 120×50 elements with $\Delta y = 0.1$ m, $\Delta x_{\min} = 0.05$ m, and $\Delta x_{\max} = 0.1$ m. All of the elements have second order polynomial basis and accordingly

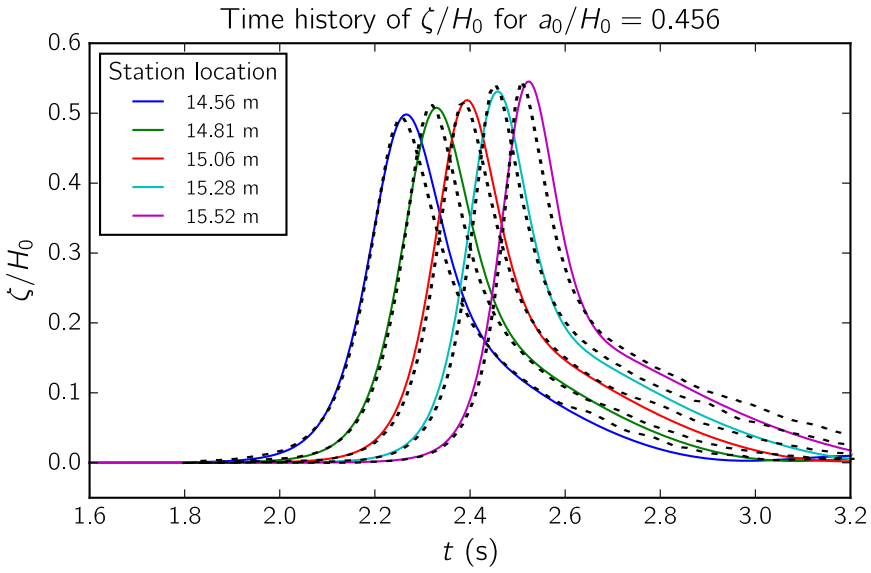


(a)

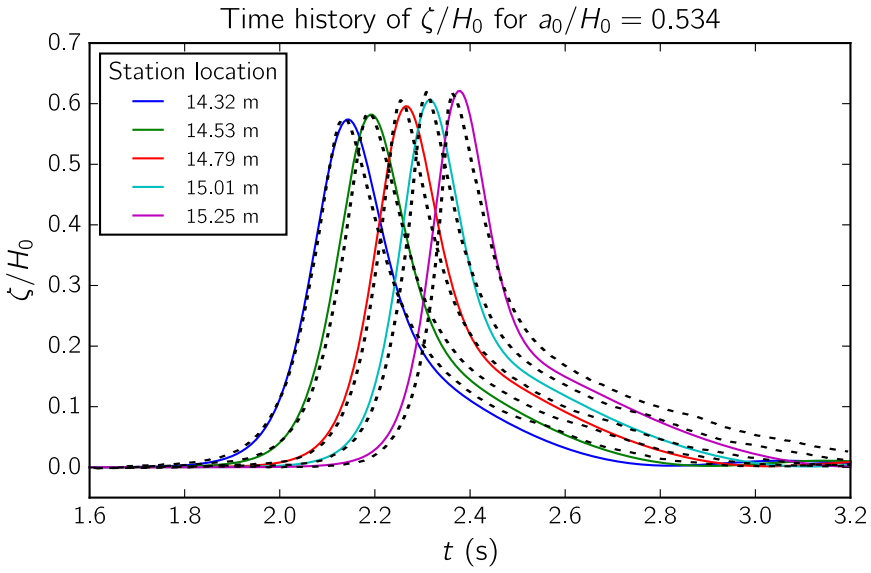


(b)

Figure 5.13: Time history of water surface elevation at different locations near the shore, for Example 4. The numerical results are shown in continuous lines and the corresponding experimental values are shown in dotted lines.



(c)



(d)

Figure 5.13 (cont.): Time history of water surface elevation at different locations near the shore, for Example 4. The numerical results are shown in continuous lines and the corresponding experimental values are shown in dotted lines.

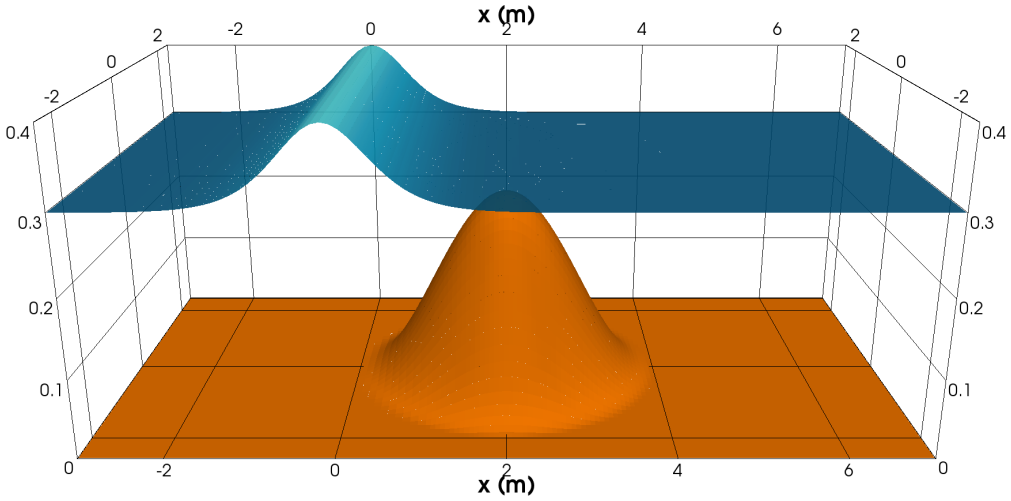
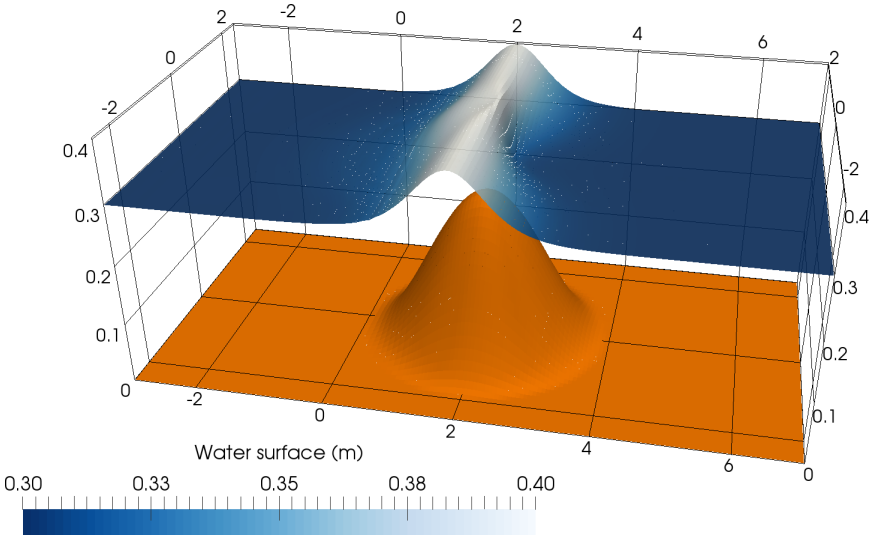
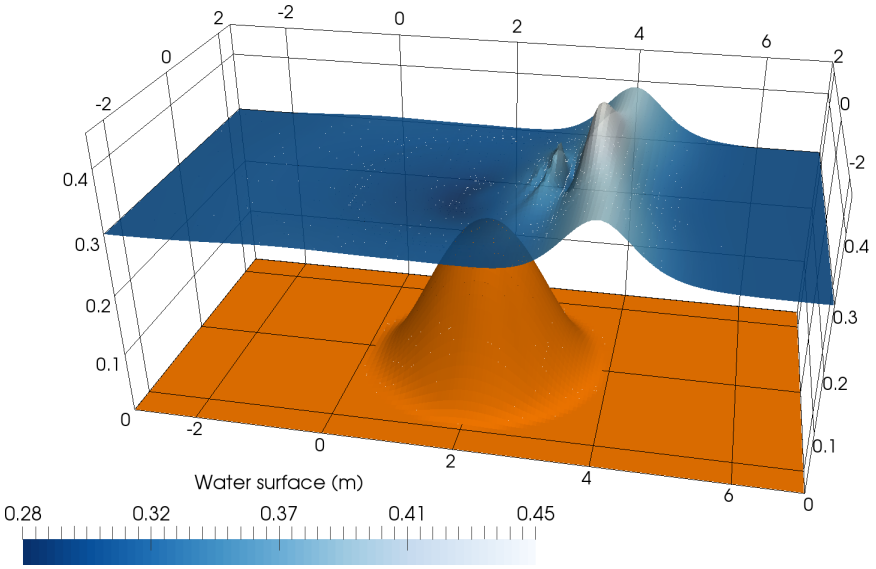


Figure 5.14: The bathymetry and initial state of water surface in Example 5.

the time step size is taken $\Delta t = 0.005$ s. The water surface at different time steps during the simulation is shown in Fig. 5.15. As the solitary wave travels over the hump, its shape starts to change (5.15a) and it is no more the exact solution to the Green-Naghdi equation. Hence, we notice in 5.15b that the long waves have traveled ahead of the shorter waves and after the reflection in 5.15c, the same pattern is followed by the returning waves.

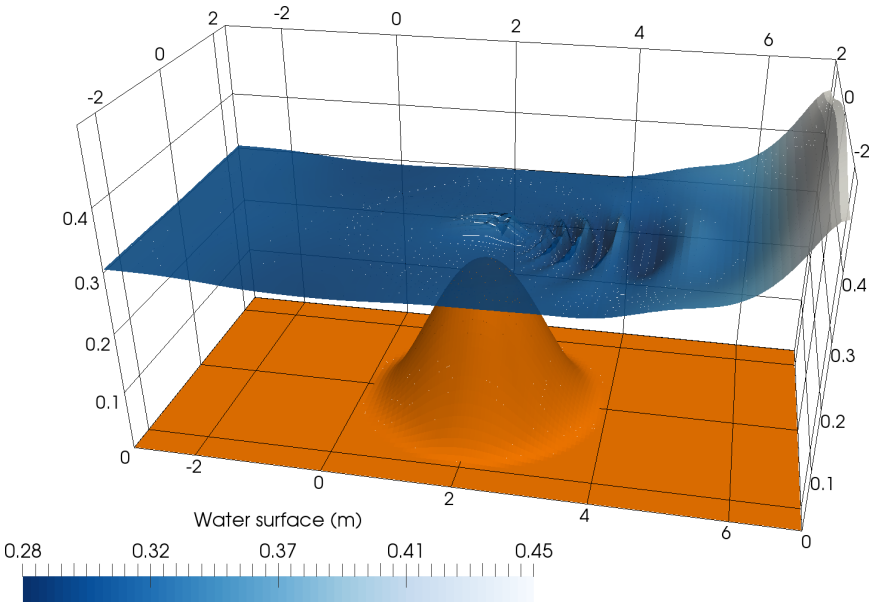


(a) $t = 1.01$ s

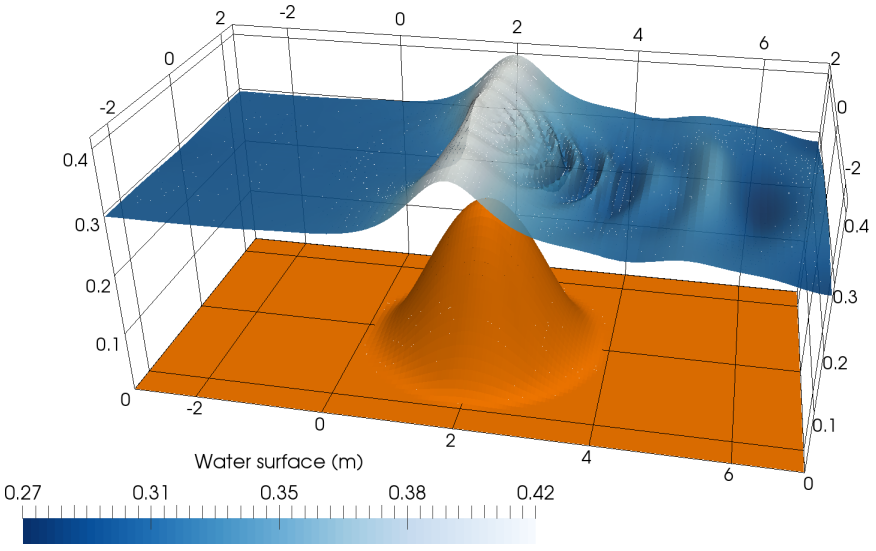


(b) $t = 2.00$ s

Figure 5.15: Water surface elevation of Example 5 at different time steps.

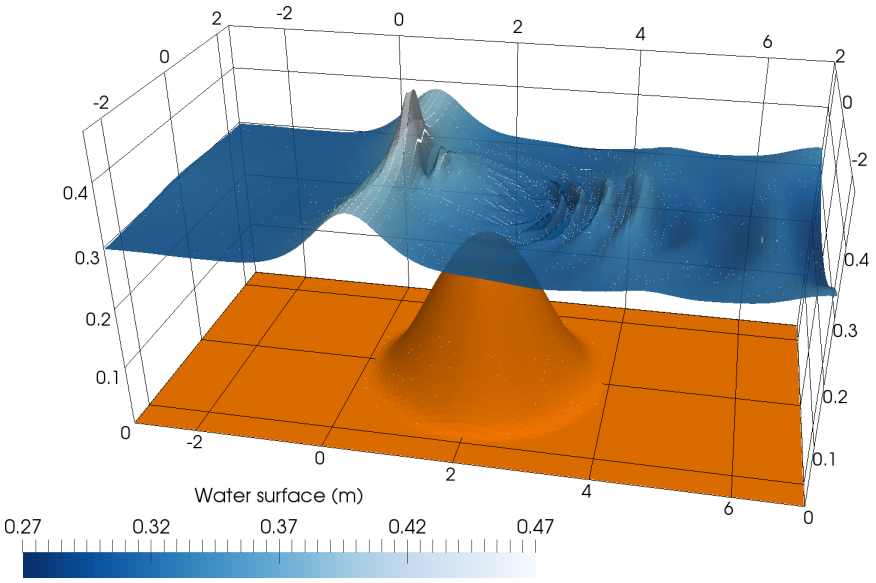


(c) $t = 3.82$ s



(d) $t = 6.28$ s

Figure 5.15 (cont.): Water surface elevation of Example 5 at different time steps.



(e) $t = 6.96$ s

Figure 5.15 (cont.): Water surface elevation of Example 5 at different time steps.

Chapter 6

Conclusion

In this study we solved three equations of the shallow water regime using the hybridized discontinuous Galerkin method. We first discussed the solution of the KdV equation, which belongs to the family of weakly nonlinear dispersive waves. We have shown the proposed method is stable and highly accurate compared to the previous methods used in the literature. When HDG is applied to lower order equations, we express the approximate variables and numerical fluxes in terms of the numerial trace of u. Here, in order to keep the same workflow as the common HDG schemes, we include the numerical traces of $\partial_x u$ and u as our global unknowns. By using this technique, the method can be conveniently extended to higher order equations. Despite adding another global unknown, the number of global equations is still $\mathcal{O}(k^{d-1}/h)$. Although, we solve our global set of equations for two numerical traces, the solution procedure is similar to that of common HDG implementations. Hence, we expect to inherit the corresponding properties of the HDG, especially the optimal convergence of the method. In our method, the numerical fluxes are related to the unknown traces through two stabilization parameters, σ and τ. We derived the sufficient conditions on σ and τ to construct a stable method for the linear problem. Next, for the nonlinear case, we chose the stabilization

parameter (τ) based on a Lax-Fredriech choice of flux. Afterwards, through a set of numerical experiments we have shown that by using elements of order k, the computed solution would converge optimally with order $k+1$ for every approximate solution, i.e. u, p, and q.

Next, we solved the nonlinear shallow water equation using an explicit hybridized DG. Although the NSWE is not a dispersive wave equation, it was necessary to solve this equation to be able to tackle the fully nonlinear Green-Naghdi equation by an operator splitting technique. There are a number of papers for solving nonlinear conservation laws using HDG in the literature; however, to the best of our knowledge, the presented method is one the few explicit implementations of the hybridized DG for this purpose. We demonstrated the optimal convergence of the results of the proposed method and explained the main advantages of an explicit technique compared to the more developed implicit scheme for the NSWE.

In the final part of this work, we explained our technique to solve the Green-Naghdi equation using the hybridized DG method. This equation is a fully nonlinear and dispersive equation, which is used to model nearshore water waves. We also employed a well known modification of this equation to improve its dispersive properties for a wider range of wavenumbers. The developed method is based on an operator splitting of the original equation, which involves solving a hyperbolic and a dispersive part. This operator splitting is second order accurate, if each of its constituents are at least second order accurate. To solve the hyperbolic equation we used the explicit tech-

nique, which we developed for the Saint-Venant equation. The dispersive part of the problem was also solved using the standard HDG flux. This part of the splitting involves an implicit solve step, which is known to have regularizing effects on the solution, and hence improves the numerical stability of the scheme. Finally, we carried out a set of numerical experiments to verify and validate the proposed method. We first discussed the convergence properties of the proposed technique. Next we simulated a set of standard experimental tests, which are often used as benchmarks for dispersive water waves. We showed good agreement of the numerical results of these problems, with the experimental data.

For the continuation of this research, we suggest the following directions:

- In the Green-Naghdi equation that we solved here, a matrix factorization is required at every time step for the elliptic solve stage. To avoid this expensive procedure, one can use the proposed method in this study to solve the new family of Green-Naghdi equation [42], which are known to be more computationally efficient than the one we solved here.

- Adding wetting and drying techniques to the current solver is needed to make it more appropriate for the simulation of run-up.

- As we mentioned earlier, the dispersive correction requires a good spatial discretization to give accurate results. By refining the mesh for this equation, we will need a reduced time step size to maintain the stability of

the NSWE solver. However, if we use an implicit solver for the hyperbolic part of splitting, we can get a more efficient solver. Moreover, it will be possible to use the adaptive mesh refinement which is available in the developed software.

- Applying the developed software to more practically relevant problems.

Bibliography

[1] Francesco Bassi and Stefano Rebay. A high-order accurate discontinuous finite element method for the numerical solution of the compressible navier–stokes equations. *Journal of computational physics*, 131(2):267–279, 1997.

[2] Thomas Brooke Benjamin, Jerry Lloyd Bona, and John Joseph Mahony. Model equations for long waves in nonlinear dispersive systems. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 272(1220):47–78, 1972.

[3] Daniele Boffi, Franco Brezzi, and Michel Fortin. *Mixed Finite Element Methods and Applications*, volume 44. Springer Science & Business Media, 2013.

[4] Philippe Bonneton, Florent Chazel, David Lannes, Fabien Marche, and Marion Tissier. A splitting approach for the fully nonlinear and weakly dispersive Green–Naghdi model. *Journal of Computational Physics*, 230(4):1479–1498, 2011.

[5] Robert K-C Chan and Robert L Street. A computer study of finite-amplitude water waves. *Journal of Computational Physics*, 6(1):68–94, 1970.

[6] G Chavent and G Salzano. A finite-element method for the 1-d water flooding problem with gravity. *Journal of Computational Physics*, 45(3):307–344, 1982.

[7] Guy Chavent and Bernardo Cockburn. The local projection $P^0 - P^1$-discontinuous-Galerkin finite element method for scalar conservation laws. *RAIRO-Modélisation mathématique et analyse numérique*, 23(4):565–592, 1989.

[8] S Chippada, Clint N Dawson, ML Martinez, and Mary F Wheeler. Finite element approximations to the system of shallow water equations i: Continuous-time a priori error estimates. *SIAM journal on numerical analysis*, 35(2):692–711, 1998.

[9] S Chippada, CN Dawson, ML Martinez-Canales, and MF Wheeler. Finite element approximations to the system of shallow water equations, part ii: Discrete-time a priori error estimates. *SIAM journal on numerical analysis*, 36(1):226–250, 1998.

[10] Bernardo Cockburn and Clint Dawson. Some extensions of the local discontinuous Galerkin method for convection-diffusion equations in multidimensions. 1999.

[11] Bernardo Cockburn, Jayadeep Gopalakrishnan, and Raytcho Lazarov. Unified hybridization of discontinuous Galerkin, mixed, and continuous Galerkin methods for second order elliptic problems. *SIAM Journal on Numerical Analysis*, 47(2):1319–1365, 2009.

[12] Bernardo Cockburn, Suchung Hou, and Chi-Wang Shu. The Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws. iv. the multidimensional case. *Mathematics of Computation*, 54(190):545–581, 1990.

[13] Bernardo Cockburn, San-Yih Lin, and Chi-Wang Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws iii: one-dimensional systems. *Journal of Computational Physics*, 84(1):90–113, 1989.

[14] Bernardo Cockburn and Chi-Wang Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws. ii. general framework. *Mathematics of computation*, 52(186):411–435, 1989.

[15] Bernardo Cockburn and Chi-Wang Shu. The local discontinuous Galerkin method for time-dependent convection-diffusion systems. *SIAM Journal on Numerical Analysis*, 35(6):2440–2463, 1998.

[16] Bernardo Cockburn and Chi-Wang Shu. The runge–kutta discontinuous Galerkin method for conservation laws v: multidimensional systems. *Journal of Computational Physics*, 141(2):199–224, 1998.

[17] Adrian Constantin and David Lannes. The hydrodynamical relevance of the Camassa–Holm and Degasperis–Procesi equations. *Archive for Rational Mechanics and Analysis*, 192(1):165–186, 2009.

[18] Walter Craig. An existence theory for water waves and the Boussinesq and Korteweg-deVries scaling limits. *Communications in Partial Differential Equations*, 10(8):787–1003, 1985.

[19] Walter Craig and Catherine Sulem. Numerical simulation of gravity waves. *Journal of Computational Physics*, 108(1):73–83, 1993.

[20] Walter Craig, Catherine Sulem, and Pierre-Louis Sulem. Nonlinear modulation of gravity waves: a rigorous approach. *Nonlinearity*, 5(2):497, 1992.

[21] MJP Cullen. A finite element method for a non-linear initial value problem. *IMA Journal of Applied Mathematics*, 13(2):233–247, 1974.

[22] Clint Dawson. Conservative, shock-capturing transport methods with nonconservative velocity approximations. *Computational Geosciences*, 3(3-4):205–227, 1999.

[23] Clint Dawson, Joannes J Westerink, Jesse C Feyen, and Dharhas Pothina. Continuous, discontinuous and coupled discontinuous–continuous Galerkin finite element methods for the shallow water equations. *International Journal for Numerical Methods in Fluids*, 52(1):63–88, 2006.

[24] A. B. de Saint-Venant. Thorie du mouvement non permanent des eaux, avec application aux crues des rivires et l'introduction des mares dans leur lit. *C. R. Acad. Sc. Paris*, 73, 1871.

[25] Daniele Antonio Di Pietro and Alexandre Ern. *Mathematical aspects of discontinuous Galerkin methods*, volume 69. Springer Science & Business Media, 2011.

[26] Nicholas Dodd. Numerical model of wave run-up, overtopping, and regeneration. *Journal of Waterway, Port, Coastal, and Ocean Engineering*, 124(2):73–81, 1998.

[27] Arnaud Duran and Fabien Marche. Discontinuous Galerkin discretization of a new class of Green-Naghdi equations. *Communications in Computational Physics*, 17(03):721–760, 2015.

[28] Arnaud Duran and Fabien Marche. A discontinuous Galerkin method for a new class of Green-Naghdi equations on simplicial unstructured meshes. *arXiv preprint arXiv:1604.05227*, 2016.

[29] Allan P Engsig-Karup, Jan S Hesthaven, Harry B Bingham, and Per A Madsen. Nodal dg-fem solution of high-order Boussinesq-type equations. *Journal of engineering mathematics*, 56(3):351–370, 2006.

[30] David R Fuhrman and Harry B Bingham. Numerical solutions of fully non-linear and highly dispersive Boussinesq equations in two horizontal dimensions. *International Journal for Numerical Methods in Fluids*, 44(3):231–255, 2004.

[31] Albert E Green and Paul M Naghdi. A derivation of equations for wave

propagation in water of variable depth. *Journal of Fluid Mechanics*, 78(02):237–246, 1976.

[32] Jan S Hesthaven and Tim Warburton. *Nodal discontinuous Galerkin methods: algorithms, analysis, and applications*. Springer Science & Business Media, 2007.

[33] Helge Holden, Kenneth Hvistendahl Karlsen, and Nils Henrik Risebro. Operator splitting methods for generalized Korteweg–de Vries equations. *Journal of Computational Physics*, 153(1):203–222, 1999.

[34] Justin Holmer. The initial-boundary value problem for the kortewegde vries equation. *Communications in Partial Differential Equations*, 31(8):1151–1190, 2006.

[35] Mutsuto Kawahara, Hirokazu Hirano, Khoji Tsubota, and Kazuo Inagaki. Selective lumping finite element method for shallow water flow. *International Journal for Numerical Methods in Fluids*, 2(1):89–112, 1982.

[36] Ingemar Kinnmark. *The shallow water wave equations: formulation, analysis and application, Ph.D. Thesis, Department of Civil Engineering*. Princeton University, 1984.

[37] RL Kolar, JJ Westerink, ME Cantekin, and CA Blain. Aspects of nonlinear simulations using shallow-water models based on the wave continuity equation. *Computers & fluids*, 23(3):523–538, 1994.

[38] Diederik Johannes Korteweg and Gustav De Vries. Xli. on the change of form of long waves advancing in a rectangular canal, and on a new type of long stationary waves. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 39(240):422–443, 1895.

[39] M Kronbichler, S Schoeder, C Müller, and WA Wall. Comparison of implicit and explicit hybridizable discontinuous Galerkin methods for the acoustic wave equation. *International Journal for Numerical Methods in Engineering*, 2015.

[40] David Lannes. *The water waves problem - mathematical analysis and assymptotics*. American Mathematical Society, 2013.

[41] David Lannes and Philippe Bonneton. Derivation of asymptotic two-dimensional time-dependent equations for surface water wave propagation. *Physics of Fluids (1994-present)*, 21(1):016601, 2009.

[42] David Lannes and Fabien Marche. A new class of fully nonlinear and weakly dispersive Green–Naghdi models for efficient 2d simulations. *Journal of Computational Physics*, 282:238–268, 2015.

[43] Randall J LeVeque. *Finite volume methods for hyperbolic problems*, volume 31. Cambridge university press, 2002.

[44] Pengzhi Lin. *Numerical modeling of water waves*. CRC Press, 2008.

[45] RA Luettich Jr, JJ Westerink, and Norman W Scheffner. Adcirc: An advanced three-dimensional circulation model for shelves, coasts, and es-

tuaries. report 1. theory and methodology of adcirc-2ddi and adcirc-3dl. 1992.

[46] Daniel R Lynch and William G Gray. A wave equation model for finite element tidal computations. *Computers & fluids*, 7(3):207–228, 1979.

[47] Daniel R Lynch and William G Gray. Finite element simulation of flow in deforming regions. *Journal of Computational Physics*, 36(2):135–153, 1980.

[48] Per A Madsen, HB Bingham, and Hua Liu. A new Boussinesq method for fully nonlinear waves from shallow to deep water. *Journal of Fluid Mechanics*, 462:1–30, 2002.

[49] T Maxworthy. Experiments on collisions between solitary waves. *Journal of Fluid Mechanics*, 76(01):177–186, 1976.

[50] Chiang C Mei. *The Applied Dynamics of Ocean Surface Waves*. New York: John Wiley, 1983.

[51] Beny Neta and RT Williams. Stability and phase speed for various finite element formulations of the advection equation. *Computers & fluids*, 14(4):393–410, 1986.

[52] Ngoc Cuong Nguyen and Jaume Peraire. Hybridizable discontinuous Galerkin methods for partial differential equations in continuum mechanics. *Journal of Computational Physics*, 231(18):5955–5988, 2012.

[53] Ngoc Cuong Nguyen, Jaume Peraire, and Bernardo Cockburn. An implicit high-order hybridizable discontinuous Galerkin method for linear convection–diffusion equations. *Journal of Computational Physics*, 228(9):3232–3254, 2009.

[54] Ngoc Cuong Nguyen, Jaume Peraire, and Bernardo Cockburn. An implicit high-order hybridizable discontinuous Galerkin method for nonlinear convection–diffusion equations. *Journal of Computational Physics*, 228(23):8841–8855, 2009.

[55] Nishant Panda, Clint Dawson, Yao Zhang, Andrew B Kennedy, Joannes J Westerink, and Aaron S Donahue. Discontinuous Galerkin methods for solving Boussinesq–Green–Naghdi equations in resolving non-linear and dispersive surface water waves. *Journal of Computational Physics*, 273:572–588, 2014.

[56] J Peraire, OC Zienkiewicz, and K Morgan. Shallow water problems: a general explicit formulation. *International Journal for Numerical Methods in Engineering*, 22(3):547–574, 1986.

[57] Jaime Peraire, NC Nguyen, and Bernardo Cockburn. A hybridizable discontinuous Galerkin method for the compressible euler and Navier-Stokes equations. 2010.

[58] Henry Power and Allen T Chwang. On reflection of a planar solitary wave at a vertical wall. *Wave Motion*, 6(2):183–195, 1984.

[59] Sander Rhebergen and Bernardo Cockburn. A space–time hybridizable discontinuous Galerkin method for incompressible flows on deforming domains. *Journal of Computational Physics*, 231(11):4185–4204, 2012.

[60] Ali Samii, Craig Michoski, and Clint Dawson. A comparison of the explicit and implicit hybridizable discontinuous Galerkin methods for nonlinear shallow water equation. *In preparation*, 2016.

[61] Ali Samii, Craig Michoski, and Clint Dawson. A parallel and adaptive hybridized discontinuous Galerkin method for anisotropic nonhomogeneous diffusion. *Computer Methods in Applied Mechanics and Engineering*, 304:118–139, 6 2016.

[62] Ali Samii, Nishant Panda, Craig Michoski, and Clint Dawson. A hybridized discontinuous Galerkin method for the nonlinear Korteweg–de Vries equation. *Journal of Scientific Computing*, 68(1):191–212, 2016.

[63] François Serre. Contribution à l'étude des écoulements permanents et variables dans les canaux. *La Houille Blanche*, (6):830–872, 1953.

[64] Chi-Wang Shu and Stanley Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of Computational Physics*, 77(2):439–471, 1988.

[65] M Stanglmeier, NC Nguyen, J Peraire, and B Cockburn. An explicit hybridizable discontinuous Galerkin method for the acoustic wave equation.

Computer Methods in Applied Mechanics and Engineering, 300:748–769, 2016.

[66] Gilbert Strang. On the construction and comparison of difference schemes. *SIAM Journal on Numerical Analysis*, 5(3):506–517, 1968.

[67] CH Su and Rida M Mirie. On head-on collisions between two solitary waves. *Journal of Fluid Mechanics*, 98(03):509–525, 1980.

[68] MP Ueckermann and PFJ Lermusiaux. Hybridizable discontinuous Galerkin projection methods for navier–stokes and Boussinesq equations. *Journal of Computational Physics*, 306:390–421, 2016.

[69] MA Walkley, Martin Berzins, et al. A finite element method for the one-dimensional extended Boussinesq equations. *International Journal for Numerical Methods in Fluids*, 29(2):143–157, 1999.

[70] Roy A Walters and Graham F Carey. Analysis of spurious oscillation modes for the shallow water and Navier-Stokes equations. *Computers & Fluids*, 11(1):51–68, 1983.

[71] Hsuan-heng Wang, Paul Halpern-ism, Jim Douglas, et al. Numerical solutions of the one-dimensional primitive equations using Galerkin approximations with localized basis functions.

[72] Ge Wei, James T Kirby, et al. A fully nonlinear Boussinesq model for surface waves. part 1. highly nonlinear unsteady waves. 1995.

[73] RT Williams. On the formulation of finite-element prediction models. *Monthly Weather Review*, 109(3):463–466, 1981.

[74] RT Williams and OC Zienkiewicz. Improved finite element forms for the shallow-water wave equations. *International Journal for Numerical Methods in Fluids*, 1(1):81–97, 1981.

[75] Jue Yan and Chi-Wang Shu. A local discontinuous Galerkin method for kdv type equations. *SIAM Journal on Numerical Analysis*, 40(2):769–791, 2002.

[76] Vladimir E Zakharov. Stability of periodic waves of finite amplitude on the surface of a deep fluid. *Journal of Applied Mechanics and Technical Physics*, 9(2):190–194, 1968.

[77] JA Zelt. The run-up of nonbreaking and breaking solitary waves. *Coastal Engineering*, 15(3):205–246, 1991.

[78] Yao Zhang, Andrew B Kennedy, Nishant Panda, Clint Dawson, and Joannes J Westerink. Boussinesq–Green–Naghdi rotational water wave theory. *Coastal Engineering*, 73:13–27, 2013.