The University of Leeds

School of Mathematics

Modelling the Transition of Tides to Bores

Submitted in accordance with the requirements for the degree of
MSc. Atmosphere-Ocean Dynamics

Author:
Kieran Newman

Supervisor:
Prof. Onno Bokhove

August 2015

The candidate confirms that the work submitted is his/her own and that appropriate credit has been given where reference has been made to the work of others.

I am most grateful to Professor Onno Bokhove for his help throughout this project, to Professor Stephen Griffiths for his help with the tidal boundary conditions, and to Jessica Eastwell for cups of tea and exceptional moral support. This dissertation is dedicated to my late Granddad, who had his own up close and personal experience with the Trent Aegir, but may not have had fond memories of it.

Abstract

In this dissertation we will attempt to form a model to represent the motion of water in tidal bores on estuaries and rivers, with as much accuracy as possible. We will then attempt to apply this to the case of the Trent Aegir, the bore on the River Trent, off the Humber Estuary. We first present a short introduction to some of the notation, approximations and concepts used, then introduce the Trent/Ouse/Humber system. The system will then be scaled and non-dimensionalised. We then go on to discuss different approximations to the fluid motion, starting with nonconservative and linearised shallow water equations, and moving on to conservative shallow water, variational and Green-Naghdi models.

The theory of tidal bores is then discussed, as well as some examples from around the world. We also consider the boundary conditions needed for tidal models, as well as the coupling of two models.

A detailed discussion of finite element, Galerkin and Ritz methods for spatial discretisation follows, with additional focus on mesh generation and basis functions in one and two dimensions. Time discretisation is also discussed, and we seek an energy-conserving method, thus formulating a discrete energy equation.

Finally, we discuss the results and approximations used, and present a detailed list of further work and topics that would improve the modelling. The full model code is also presented in the appendices, as well as the detailed energy formulation for the two dimensional model.

Contents

List of Figures

Chapter 1

Introduction

Tidal bores are a phenomenon still undergoing considerable study (in areas such as turbulence and air-sea exchange processes). The basic principles of bore motion and hydraulic jumps are well know, however most bore research in the UK seems to focus on the (admittedly large) Severn or Mersey bores. Here we consider the bore on the Trent, Britain's third longest river. This bore, or Aegir, can be very well developed, and penetrates a significant way upstream. The shifting bottom of the Humber Estuary can also have a large effect. While not so much of a problem in modern times as a hundred years ago, tidal bores can still pose a hazard to small shipping in some cases, and still draw crowds of spectators when a particularly spectacular breaker is forecast, with surfers also drawn to the larger waves.

1.1 Notation

The first partial derivative is denoted by

$$\frac{\partial}{\partial t} = \partial_t, \tag{1.1}$$

and the second by

$$\frac{\partial^2}{\partial t^2} := \partial_{tt}, \tag{1.2}$$

$$\frac{\partial^2}{\partial x \partial y} := \partial_{xy} \tag{1.3}$$

for double and mixed derivatives respectively. Vectors are denoted by

$$\boldsymbol{a} = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix}. \tag{1.4}$$

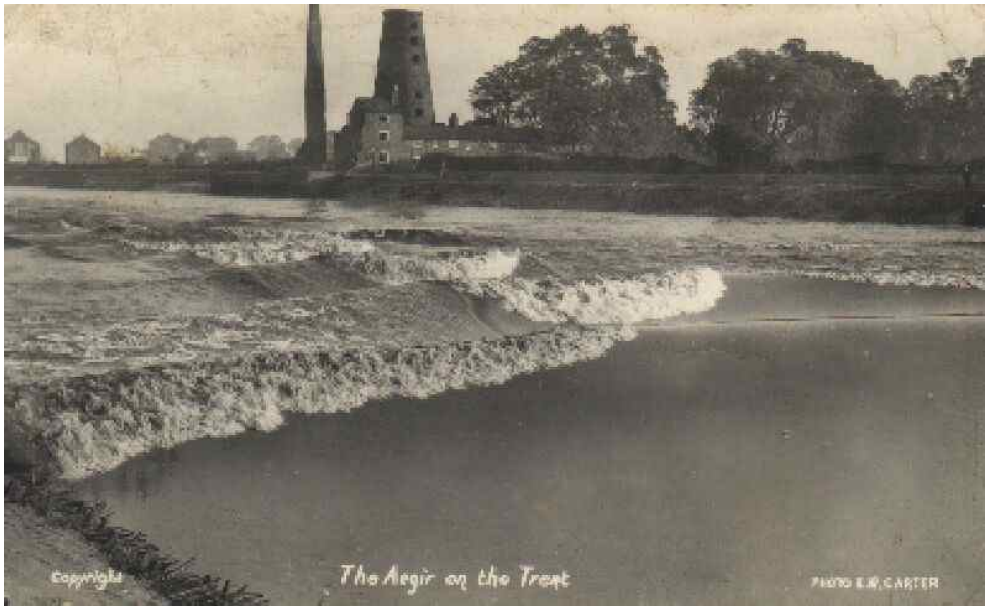

Figure 1.1: The Trent Aegir, showing the "Seven Sisters" or whelps, a series of smaller waves behind the leading wave. From Carter (No Date).

Velocity is notated in three dimensional cartesian coordinates as

$$\boldsymbol{u} = \begin{pmatrix} u \\ v \\ w \end{pmatrix}, \tag{1.5}$$

density as ρ, and pressure as P. (λ, ϕ) is used for latitude and longitude. Also used is the differential operator ∇, defined (for coordinate system $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)^T$) as

$$\nabla = \begin{pmatrix} \partial_{x_1} \\ \partial_{x_2} \\ \vdots \\ \partial_{x_n} \end{pmatrix}. \tag{1.6}$$

1.2 Equations of Motion

For an incompressible fluid, the fundamental equations that describe its motion are the *Navier-Stokes* equations, given in this case by

$$\partial_t \boldsymbol{u} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\frac{1}{\rho}\nabla P + \boldsymbol{F}, \tag{1.7}$$

$$\nabla \cdot \boldsymbol{u} = 0, \tag{1.8}$$

$$\partial_t \rho + \boldsymbol{u} \cdot \nabla \rho = 0, \tag{1.9}$$

where $\boldsymbol{F}$ represents any body force.

1.3 Approximations

As well as the assumption of incompressibility, we often make the following assumptions to simplify the equations involved in predicting fluid motion.

Hydrostatic Equilibrium

A fluid is in *hydrostatic balance* if

$$\partial_z P(z) = -\rho(z)g. \tag{1.10}$$

Dynamically, this means that the pressure gradient force is balanced by the action of gravity on the fluid, implying that pressure increases with depth.

The Boussinesq Approximation

When the fluid we are modelling is water, there is very little deviation in density from the mean value, $\bar{\rho}$. We can use this to simplify our momentum equations, by only selectively considering changes in density when they are important for buoyant effects. We can write

$$\rho(\boldsymbol{x}, t) = \bar{\rho} + (\rho(\boldsymbol{x}, t) - \bar{\rho}), \tag{1.11}$$

under the condition

$$\|\rho(\boldsymbol{x}, t) - \bar{\rho}\| \ll \bar{\rho}. \tag{1.12}$$

Body forces can also be rewritten

$$\boldsymbol{F} = \boldsymbol{g} + \boldsymbol{D}, \tag{1.13}$$

with

$$\boldsymbol{g} = \begin{pmatrix} 0 \\ 0 \\ -g \end{pmatrix}, \tag{1.14}$$

the gravitational force and $\boldsymbol{D}$ any further external forces acting on the fluid. Substituting equations (1.11) and (1.13) into equation (1.7), we get

$$\begin{aligned}[\bar{\rho} + (\rho - \bar{\rho})]\frac{D\boldsymbol{u}}{Dt} = &-\nabla P + [\bar{\rho} + (\rho - \bar{\rho})]\boldsymbol{g} \\ &+ [\bar{\rho} + (\rho - \bar{\rho})]\boldsymbol{D}.\end{aligned} \tag{1.15}$$

As a result of the condition equation (1.12), we can take $[\bar{\rho} + (\rho - \bar{\rho})] \approx \bar{\rho}$. We selectively apply this, as density variations combined with gravity give rise to bouyant effects, so we retain the full $\bar{\rho} + (\rho - \bar{\rho})$ in the gravity term.

However, in the other terms, the small density variations are less important, so we obtain

$$\frac{D\boldsymbol{u}}{Dt} = -\frac{1}{\bar{\rho}}\nabla P - \frac{\rho}{\bar{\rho}}\boldsymbol{g} + \boldsymbol{D}, \tag{1.16}$$

$$\nabla \cdot u = 0, \tag{1.17}$$

$$\frac{D\rho}{Dt} = 0, \tag{1.18}$$

the *Boussinesq approximation*.

Chapter 2

The Humber Estuary

The Humber Estuary stretches 62 km from Spurn Head in the east to the confluence of the Ouse and Trent rivers at Faxfleet/Trent Falls in the west. The largest tributaries to the estuary are the Hull, Ouse and Trent rivers (directly), and the Don, Aire, Wharfe and Derwent rivers (through the Ouse), with a combined total fresh water input of around $250\,\mathrm{m^3\,s^{-1}}$ (van Rijn, 2011). The estuary (and associated rivers) are tidal for a significant portion of their length, with a total tidal length of 300 km, and a mean tidal range (at spring tides) of around 6 m (UK Hydrographic Office, 1993). Further analysis of UK Hydrographic Office (1993), as in van Rijn (2011) gives a rough mouth width of 6000 m and depth of 15 m at Grimsby, shallowing to around 3 m (on average, with a range of 0.61 m–6.60 m) at Keadby, around halfway along the tidal stretch of the river Trent (Environment Agency, 2015). A map of the Humber (including the Ouse and Trent) can be seen in figure 2.1. For incorporation into the model, the estuary map in figure 2.2 (a simpler drawing than figure 2.1) was loaded into imaging software and traced with a path tool. The coordinates of this trace were then exported and used to form a CSG polygon in MATLAB, which was used as the geometry for mesh generation.

2.1 Scales

Using the data provided in van Rijn (2011), we can estimate the values of various nondimensional parameters for the Humber Estuary.

The Rossby number gives an indication of the relative importance of Coriolis effects compared to inertial effects. A high Rossby number indicates that Coriolis effects can be ignored, while a low number suggests they are important to the dynamics of the system. The Rossby number is given by

$$Ro = \frac{U}{fL}, \tag{2.1}$$

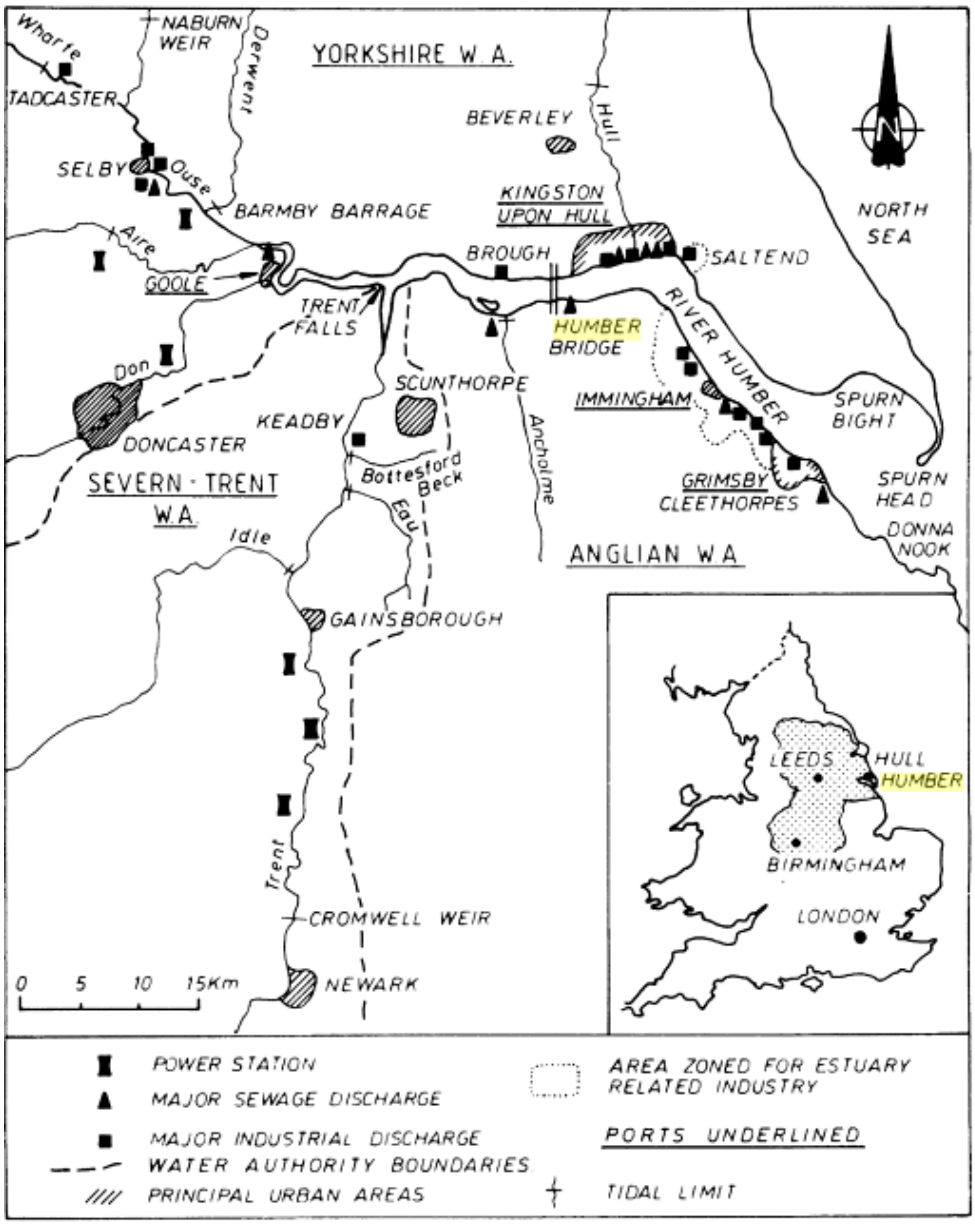

Figure 2.1: Map of the Humber Estuary. From Edwards et al. (1988, p. 688).

where U is a typical horizontal velocity scale, L a typical horizontal length scale, and $f = 2\Omega \sin(\lambda)$ is the Coriolis parameter, for $\Omega = 7.29 \times 10^{-5}\,\mathrm{rad/s}$, the rotation rate of the Earth, at latitude λ.

For the Humber, we take $U = 1.5\,\mathrm{m\,s^{-1}}$, $L = 15\,\mathrm{km}$, and $\lambda = 53.74°$ (Hull). This gives us $Ro = 1.7$, sufficiently high that Coriolis effects can probably be neglected.

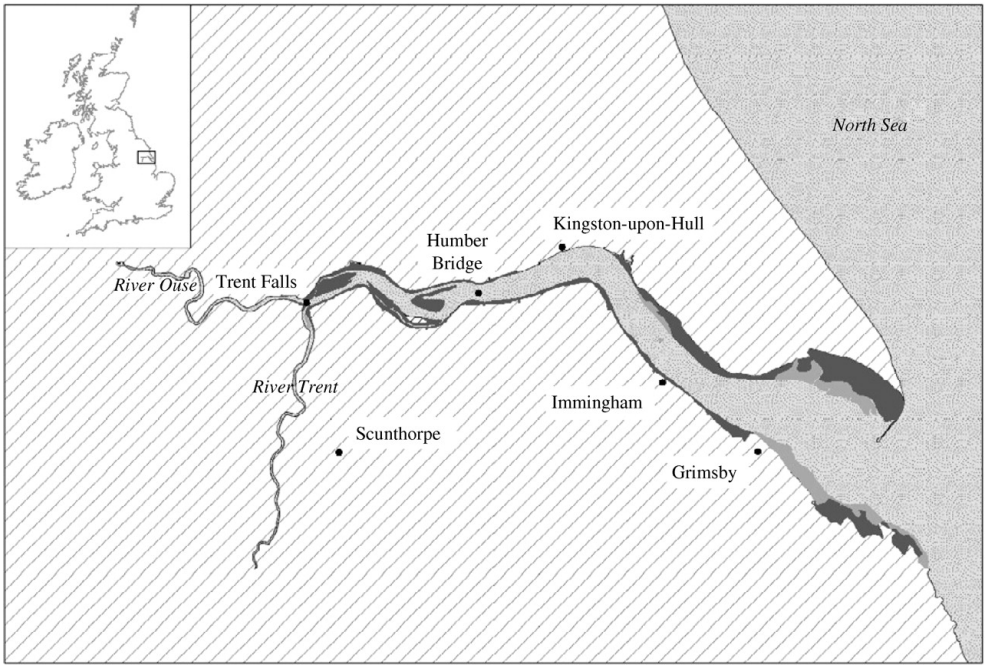

Figure 2.2: Map of the Humber Estuary used for polygon tracing. From Townend et al. (2007).

The choice of length scale L is based on the *tidal excursion* E_t in the Humber (see figure 2.3). This is a measure of the distance a fluid parcel would be moved by the tide, and can be calculated as

$$E_t = \int_{T_{LWS}}^{T_{HWS}} \boldsymbol{u}_t dt, \tag{2.2}$$

where T_{LWS} and T_{HWS} are the times of low water slack tide and high water slack tide respectively (Savenije, 2006, p. 11). The width of the Estuary also limits horizontal scales, however it is assumed that the fluid motion along the estuary is significantly greater than across the estuary. Edwards et al. (1988, p. 687) give an average tidal excursion of 15 km for the Humber, and it is this we use for L.

We assume that scales of horizontal motion and length U and L are much greater than scales of vertical motion and depth W (unknown) and H. This enables us to make the shallow water approximation in section 3.1, and the approximation of fluid moving in vertical columns in chapter 5. This can be seen for length scales, at least, by comparing the depth of domain $H \approx 15$ m at maximum with the size of the domain in the horizontal.

We introduce also non-dimensional coordinates and variables, which are used throughout the rest of this report. Following Bokhove and Griffiths

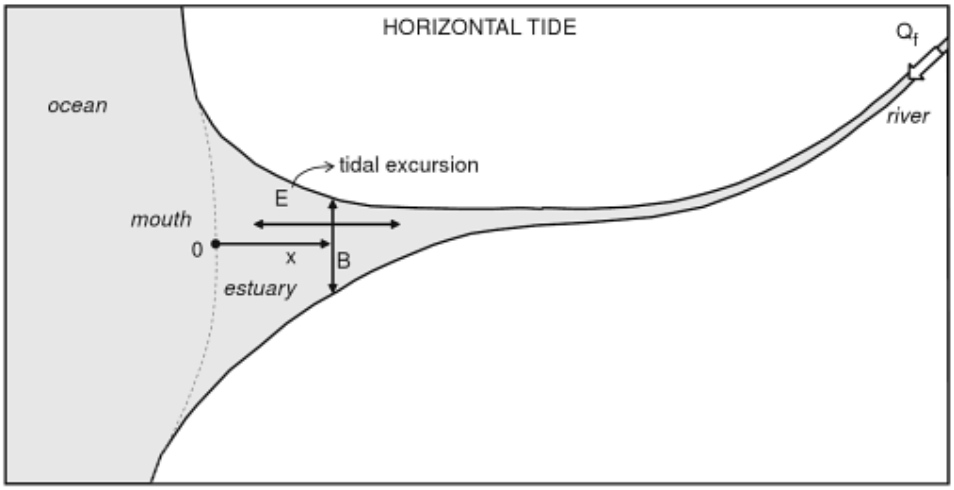

Figure 2.3: Schematic demonstrating the idea of width B compared with *tidal excursion* E, from Savenije (2006, p. 12).

(2015), we define

$$(\hat{x}, \hat{y}) = \frac{(x, y)}{L} \qquad \hat{t} = \frac{tL}{U}, \tag{2.3}$$

$$\hat{u} = \frac{u}{U}, \qquad \hat{v} = \frac{v}{U}, \tag{2.4}$$

for length and velocity scales U, L. These will be used, dropping the "hat" notation, from now on in this report, unless explicitly stated.

2.2 Tides at the Ocean Boundary

The oceanic boundary at the east end of the Humber Estuary is with the North Sea. The principal tidal constituent for the North Sea is the M_2, principal semi-diurnal lunar tide, as determined by Proudman and Doodson (1924). We model this at the boundary with a sinusoidal variation in pertubation surface height (as discussed in section 7.1), with amplitude 6 m and frequency 4π rad/day, which correspond to 4×10^{-4} and 1.454×10^{-8} when converted into our nondimensional coordinates.

2.3 River Trent

The River Trent runs 298 km (making it the third longest river in England) from its source in Staffordshire to its join with the Humber and Ouse at Trent Falls, passing through a significant stretch of the Midlands on the way (Fort, 2008). It is notable both for its well-developed bore, and also for its

significant history of floods, due to the large cachment area (incorporating much of the Peak District). The Trent is tidal up to the weir at Cromwell, and is significantly shallower than the Humber into which it flows (Tricker, 1964, p. 61). There are also a number of sandbanks in the estuary near Trent Falls, which may have an effect on the bore formation.

Trent Aegir

Known locally as the *Aegir* or *Eagre*, the bore on the Trent is one of the most well-developed in the UK, after the Severn and Mersey. The currents and bore near Trent Falls have been responsible for a number of accidents with barges (Tricker, 1964, p. 61), although the name is short for "outfall", meaning the confluence of the two rivers, rather than a reference to walls of water. The formation of the bore here, as previously mentioned, is though to be due to the shallowing of the channel, in conjunction with the sandbanks in the area and the narrowing of the estuary, which amplifies the tidal range. To incorporate this reduced depth, while still keeping a constant H in each

Figure 2.4: The Trent Aegir at Gainsborough, Lincolnshire, on 20th September 2005 (*The Trent Aegir at Gainsborough, Lincolnshire*, 2005).

model, we take H_{GN} (for the Green-Naghdi model) as ${}^{2H_{SW}}/_{15}$, where H_{SW} is the shallow water depth.

Chapter 3

The Shallow Water/Saint Venant Equations

3.1 Approximations

When working with homogeneous, incompressible fluids, we generally use the Boussinesq approximation described in section 1.3. If the hydrostatic approximation is also used (section 1.3), then we can rewrite the equations of motion as

$$\frac{D\boldsymbol{u}}{Dt} - f\boldsymbol{e}_z \times \boldsymbol{u} = -\frac{1}{\rho}\nabla P + \boldsymbol{F}, \tag{3.1}$$

$$\partial_z P = \rho g, \tag{3.2}$$

$$\nabla \cdot \boldsymbol{u} = 0, \tag{3.3}$$

where the vectors and operations in (3.1) are in two dimensions (Bokhove and Griffiths, 2015). $\boldsymbol{F}$ represents any external forces acting horizontally on the fluid. This approximation holds on either the f–plane ($f = f_0$) or the β–plane ($f = f_0 + \beta y$), when the system is assumed *shallow*, i.e. when the typical horizontal length (L) and velocity (U) scales are much larger than their vertical equivalents (H and W, respectively). If we consider a system as in figure 3.1, then we can integrate the hydrostatic relation ((3.2)) over depth to find P, giving

$$P(x, y, z, t) = P_{atm} + \rho g h(x, y, t) - \rho g z. \tag{3.4}$$

This means that the horizontal pressure gradient in (3.1) can be written as a function of h, the free surface height

$$\frac{1}{\rho}\nabla P = g\nabla h. \tag{3.5}$$

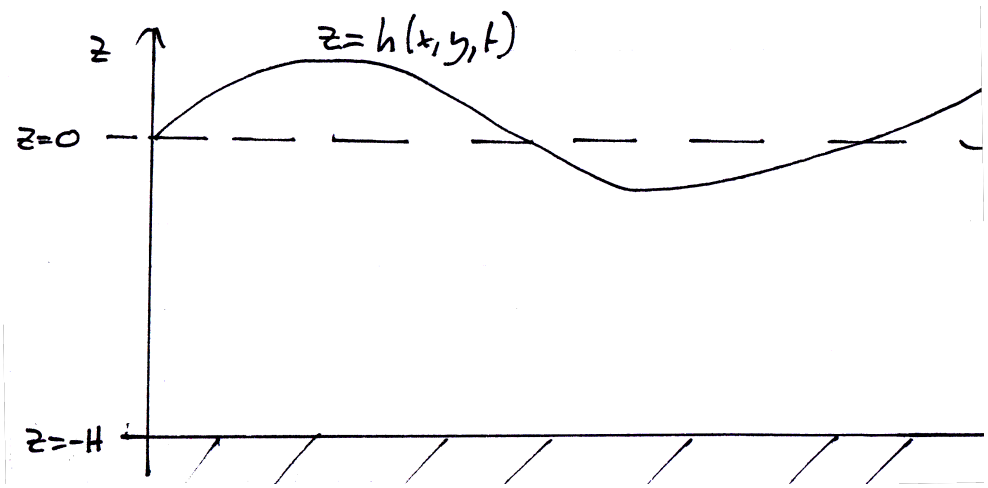

Figure 3.1: Setup for shallow water equations (equations (3.1) to (3.3))

3.2 Standard Form

Using the approximations discussed above, we consider solutions of the form

$$\boldsymbol{u}(x,y,t) = \begin{pmatrix} u(x,y,t) \\ v(x,y,t) \end{pmatrix}, \tag{3.6}$$

i.e. motion is horizontal and independent of z (provided $\boldsymbol{F}$ is also independent of z). With this form of solution, and the formation of the pressure gradient given in (3.5), we have

$$\partial_t \boldsymbol{u} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} - f \boldsymbol{e}_z \times \boldsymbol{u} = -g \nabla h + \boldsymbol{F}, \tag{3.7}$$

where again, the ∇ operator is in two dimensions.

This is two equations in three unknowns u, v and h, so we require a third equation linking these three variables. Expanding (3.3), we have

$$-\left(\partial_x u + \partial_y v\right) = \partial_z w. \tag{3.8}$$

We can use the following assumptions to then integrate (3.8) with respect to z:

A1: No normal flow at the sea bottom $-H(x,y)$ (i.e. $\boldsymbol{u} \cdot \boldsymbol{n} = 0$).

A2: Free surface boundary at $h(x,y,t)$ (i.e. $\boldsymbol{u} \cdot \boldsymbol{n} = \boldsymbol{u}_{surface} \cdot \boldsymbol{n}$).

We then have

$$\int_{-H}^{h} \partial_z w dz = \int_{-H}^{h} -(\partial_x u + \partial_y v) dz, \tag{3.9}$$

with, from (A1),

$$u\partial_x H + v\partial_y H + w = 0 \tag{3.10}$$
$$\Rightarrow w(z = -H) = -(u\partial_x H + \partial_y H) \tag{3.11}$$

and from (A2)

$$\boldsymbol{u} \cdot \begin{pmatrix} -\partial_x h \\ -\partial_y h \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \partial_t h \end{pmatrix} \cdot \begin{pmatrix} -\partial_x h \\ -\partial_y h \\ 1 \end{pmatrix} \tag{3.12}$$
$$\Rightarrow w(z = h) = \partial_t h + u\partial_x h + v\partial_y h. \tag{3.13}$$

(3.9) gives

$$w(z = h) - w(z = -H) = -(\partial_x u + \partial_y v)(h + H), \tag{3.14}$$

and, substituting (3.11) and (3.13), we have

$$\partial_t h + u\partial_x h + v\partial_y h + u\partial_x H + v\partial_y H + (\partial_x u + \partial_y v)(h + H) = 0 \tag{3.15}$$
$$\Rightarrow \partial_t h + u\partial_x(h + H) + v\partial_y(h + H) + (H + h)(\partial_x u + \partial_y v) = 0. \tag{3.16}$$

(3.16) can now be rearranged using the product rule to give

$$\partial_t h + \partial_x(u(h + H)) + \partial_y(v(h + H)) = 0. \tag{3.17}$$

This equation now completes the system, (3.7) and (3.17) are known as the *shallow water equations*, or if we ignore the y–dimension, the *Saint-Venant Equations*.

3.3 Linearisation

If we consider small disturbances to a rest state of the fluid, then we can greatly simplify these equations, and still obtain a good approximation of the fluid motion, provided the motion is approximately linear. To do this, we set $\boldsymbol{u} = 0 + \tilde{\boldsymbol{u}}$, $h = 0 + \tilde{h}$, where the tildes denote the small amplitude disturbances. Feeding this back into equations (3.7) and (3.17), with a constant depth and without body forces, we obtain

$$\partial_t \tilde{\boldsymbol{u}} + \tilde{\boldsymbol{u}} \cdot \nabla \tilde{\boldsymbol{u}} - f\boldsymbol{e}_z \times \tilde{\boldsymbol{u}} = -g\nabla\tilde{h}, \tag{3.18}$$
$$\partial_t \tilde{h} + \tilde{u}\partial_x \tilde{h} + \tilde{h}\partial_x \tilde{u} + \tilde{v}\partial_y \tilde{h} + \tilde{h}\partial_y \tilde{v} + H(\partial_x \tilde{u} + \partial_y \tilde{v} = 0. \tag{3.19}$$

As we are considering small disturbances, we neglect any terms that involve the product of disturbance terms. After this, now removing the tildes, we obtain

$$\partial_t \boldsymbol{u} + g\nabla h - f\boldsymbol{e}_z \times \tilde{\boldsymbol{u}} = 0, \tag{3.20}$$
$$\partial_t h + H(\partial_x u + \partial_y v) = 0, \tag{3.21}$$

the *linearised shallow water equations*.

The equations described here are useful for the analysis of linear waves, but nonlinearities cannot be represented in this system. This means that phenomena such as shocks and hydraulic jumps have to be modified, or cannot be shown. To counter this, we seek an alternative form of equation to represent nonlinear behaviour, which will be discussed in depth in chapter 5.

Chapter 4

Conservation Laws

Before we consider more complex models of fluid motion, it is useful to have an idea of quantities that must be conserved in the fluid. Conservation laws in fluids link the flux $\boldsymbol{J}$ of a quantity across a boundary, and the change in the density of this quantity with time (Bokhove and Griffiths, 2015). They have the general form

$$\partial_t q + \nabla \cdot \boldsymbol{J} = D, \tag{4.1}$$

where q is the density of the quantity and D represents any sources or sinks. We can integrate this equation on a volume V (with closed surface S) to obtain, using Gauss's theorem ($\int_V (\nabla \cdot \boldsymbol{A}) dV = \int_S (\boldsymbol{A} \cdot \boldsymbol{n}) dS$)

$$\frac{d}{dt} \int_V q dV = - \int_S \boldsymbol{J} \cdot \boldsymbol{n} dS + \int_V D dV. \tag{4.2}$$

(4.2) is known as the *integral form* of the conservation law.

The two conservation laws we will use to derive the shallow water equations are

Conservation of Mass Here $q = \rho$, $\boldsymbol{J} = q\boldsymbol{u}$ (transport with the fluid), $D = 0$, so we have

$$\partial_t \rho + \nabla \cdot (\rho \boldsymbol{u}) = 0. \tag{4.3}$$

Conservation of Linear Momentum Here $q = \rho \boldsymbol{u}$, $D = \rho \boldsymbol{b} + \nabla \cdot \boldsymbol{T}$, and $\boldsymbol{J}$ is again transport with the fluid, so we have

$$\partial_t (\rho \boldsymbol{u}) + \nabla \cdot (\rho \boldsymbol{u}^2) - \rho \boldsymbol{b} - \nabla \cdot \boldsymbol{T} = 0, \tag{4.4}$$

where $\boldsymbol{b}$ is the body force density per unit mass, and $\boldsymbol{T} = \boldsymbol{\sigma} \cdot \boldsymbol{n}$ is the Cauchy stress tensor

(Dawson and Mirabito, 2008).

4.1 Conservative Form of the Shallow Water Equations

We can use these laws to derive an alternative form of the shallow water equations presented in chapter 3. We assume that the fluid is incompressible, and that density is constant. We also again assume the fluid is in hydrostatic equilibrium, and thus we have

$$\rho \nabla P = g \nabla h, \tag{4.5}$$

with h the height of the free surface.

Following Dawson and Mirabito (2008), we integrate the incompressibility condition $\nabla \cdot \boldsymbol{u} = 0$ between the domain bottom at $z = -H$ and the fluid free surface at $z = h$, and consider a depth averaged velocity

$$\bar{\boldsymbol{u}} = \frac{1}{H + h} \int_{-H}^{h} \boldsymbol{u} dz. \tag{4.6}$$

We then have

$$\int_{-H}^{h} \nabla \cdot \boldsymbol{u} dz = 0 \tag{4.7}$$

$$\begin{aligned} &\partial_x \int_{-H}^{h} u dz - (u(z = h)\partial_x h + u(z = -H)\partial_x H) \\ &+\partial_y \int_{-H}^{h} v dz - (v(z = h)\partial_y h + v(z = -H)\partial_y H) \\ &+w(z = h) - w(z = -H) = 0, \end{aligned} \tag{4.8}$$

where the Leibniz integral rule

$$\partial_z \left(\int_{a(z)}^{b(z)} f(x, z) dx \right) = \int_{a(z)}^{b(z)} \frac{d}{dz} f(x, z) dx + f(z = b)\partial_z b - f(z = a)\partial_z a \tag{4.9}$$

is used between (4.7) and (4.8). We use boundary conditions at $z = -H$:

$$\boldsymbol{u} \cdot \boldsymbol{n} = 0, \tag{4.10}$$

(i.e. no normal flow) and at $z = h$:

$$\partial_t h + \boldsymbol{u} \cdot \nabla h = 0, \tag{4.11}$$

(i.e. no normal flow *relative to the current state of the free surface*). Using these conditions, and substituting in the depth averaged velocities, we now have

$$\partial_t h + \nabla \cdot (H + h)\bar{\boldsymbol{u}} = 0, \tag{4.12}$$

our new continuity equation (averaged for depth). We now integrate (8.1) over the depth of the fluid, substitute in depth-averaged velocities, and obtain

$$\partial_t((H+h)\bar{u}) + \partial_x((H+h)\bar{u}^2) + \partial_y((H+h)\bar{u}\bar{v}) = -g(H+h)\partial_x h \quad (4.13)$$

$$\partial_t((H+h)\bar{v}) + \partial_x((H+h)\bar{u}\bar{v}) + \partial_y((H+h)\bar{v}^2) = -g(H+h)\partial_y h, \quad (4.14)$$

where frictional terms, and advective terms arising from the averaging of a function product are neglected (Dawson and Mirabito, 2008). These are the shallow water equations in conservative form.

Chapter 5

Green-Naghdi Equations

6 For representation of the river once the channel has sufficiently narrowed, we use a one dimensional (assuming the channel is now narrow and straight enough that the flow is essentially uniform in the y direction) nonlinear model. A potential flow Green-Naghdi model is chosen, to ensure conservation of energy functionals, and accurately represent nonlinear dynamics within the fluid flow. We assume the bottom profile is flat, and located at $z = -H$. The notation used here differs slightly from that used for the shallow water model in chapter 3, here we denote $h = H + \eta$ the total water height above the seabed, where η is the pertubation from the resting depth. We also have a velocity potential ϕ, which is given by $\partial_x \phi \equiv u$, for u the x–directional fluid velocity. This model uses an assumption of fluid motion in vertical columns (Green and Naghdi, 1976), implying a restriction on depth (D) and length (L) scales thus:

$$\epsilon \equiv \left(\frac{D}{L}\right)^2 \ll 1 \tag{5.1}$$

(Miles and Salmon, 1985). This restriction further implies that any dispersion in the model is weak (Miles and Salmon, 1985). Following Miles and Salmon (1985), we work with an approximation to the Lagrangian density, for flow with no potential vorticity. Taking the variation of the energy functional equal to zero (for purposes of minimisation), we obtain

$$\delta \int_0^T \int_0^{L_x} h \left[\partial_t \phi + \frac{1}{2}(\partial_x \phi)^2 - \frac{1}{6}(h \partial_{xx} \phi) + \frac{1}{2} g(h - 2H)\right] dx dt = 0 \tag{5.2}$$

(Miles and Salmon, 1985). We denote the interior of this integral by $L(h, \phi)$, and use the definition of the variation of a functional

$$\delta F(a, b) \equiv \lim_{\epsilon \to 0} \frac{1}{\epsilon}(F(a + \epsilon \delta a, b + \epsilon \delta b) - F(a, b)), \tag{5.3}$$

for $\epsilon \ll 1$ (Owen, 2015). Using this, we have

$$\delta \int_0^T \int_0^{L_x} L(h,\phi)dxdt \equiv \lim_{\epsilon\to 0} \frac{1}{\epsilon} \int_0^T \int_0^{L_x} L(h+\epsilon\delta h, \phi+\epsilon\delta\phi) - L(h,\phi)dxdt \tag{5.4}$$

$$\begin{aligned} &= \lim_{\epsilon\to 0} \int_0^T \frac{1}{\epsilon} \int_0^{L_x} (h+\epsilon\delta h) \\ &\left[\partial_t(\phi+\epsilon\delta\phi) + \frac{1}{2}\left(\partial_x(\phi+\epsilon\delta\phi)\right)^2\right. \\ &\left. -\frac{1}{6}\left((h+\epsilon\delta h)\partial_{xx}(\phi+\epsilon\delta\phi)\right)^2 + \frac{1}{2}g((h+\epsilon\delta h)-2H)\right] \\ &- h\left[\partial_t\phi + \frac{1}{2}(\partial_x\phi)^2 - \frac{1}{6}(h\partial_{xx}\phi)^2 + \frac{1}{2}g(h-2H)\right] dxdt. \end{aligned} \tag{5.5}$$

We now can expand some of the terms within the first square bracket to obtain

$$\left((h+\epsilon\delta h)\partial_{xx}(\phi+\epsilon\delta\phi)\right)^2 = (h+\epsilon\delta h)^2\left((\partial_{xx}\phi)^2 + 2\partial_{xx}\phi\partial_{xx}(\epsilon\delta\phi) + (\partial_{xx}(\epsilon\delta\phi))^2\right) \tag{5.6}$$

$$= (h^2 + 2h\epsilon\delta h + (\epsilon\delta h)^2)\left((\partial_{xx}\phi)^2 + 2\partial_{xx}\phi\partial_{xx}(\epsilon\delta\phi) + (\partial_{xx}(\epsilon\delta\phi))^2\right) \tag{5.7}$$

$$\begin{aligned} &= (h\partial_{xx}\phi)^2 + h^2\left(2\partial_{xx}\phi\partial_{xx}(\epsilon\delta\phi) + (\partial_{xx}(\epsilon\delta\phi))^2\right) \\ &+ (2h\epsilon\delta h + (\epsilon\delta h)^2)\left((\partial_{xx}\phi)^2 + 2\partial_{xx}\phi\partial_{xx}(\epsilon\delta\phi) + (\partial_{xx}(\epsilon\delta\phi))^2\right), \end{aligned} \tag{5.8}$$

as well as

$$\left(\partial_x(\phi+\epsilon\delta\phi)\right)^2 = (\partial_x\phi)^2 + 2\partial_x\phi\partial_x(\epsilon\delta\phi) + (\partial_x(\epsilon\delta\phi))^2. \tag{5.9}$$

Substituting these back into (5.5), and cancelling those terms shared with the second square bracket, as well as any immediately quadratic in ϵ (which would disappear on taking the limit), we now have

$$\begin{aligned} 0 = \lim_{\epsilon\to 0} \int_0^T \frac{1}{\epsilon} \int_0^{L_x} & h\partial_t\epsilon\delta\phi + h\partial_x\phi\partial_x\epsilon\delta\phi - \frac{1}{3}h^3\left(2\partial_{xx}\phi\partial_{xx}\epsilon\delta\phi\right) \\ & - \frac{1}{3}h^2\epsilon\delta h(\partial_{xx}\phi)^2 + \epsilon\delta h\partial_t\phi + \frac{1}{2}\epsilon\delta h(\partial_x\phi)^2 - \frac{1}{6}h^2\epsilon\delta h(\partial_{xx}\phi)^2 \\ & + \frac{1}{2}g\epsilon\delta h(h-2H)dxdt. \end{aligned} \tag{5.10}$$

$$\tag{5.11}$$

At this point, as all terms in contain only a single power of ϵ, we can perform the division by ϵ and remove the limit from the equation. We now split the terms in the integral, aiming to perform a series of integration by parts and simplify the equation. We now have

$$0 = \int_0^T \mathcal{I}_1 + \mathcal{I}_2 - \frac{1}{3}\mathcal{I}_3 - \frac{1}{2}\mathcal{I}_4 + \mathcal{I}_5 dt, \tag{5.12}$$

where

$$\mathcal{I}_1 = \int_0^{L_x} \delta h \partial_t \phi + h \partial_t (\delta\phi) + \frac{1}{2}\delta h (\partial_x \phi)^2 dx, \tag{5.13}$$

$$\mathcal{I}_2 = \int_0^{L_x} h \partial_x \phi \partial_x (\delta\phi) dx \tag{5.14}$$

$$\mathcal{I}_3 = \int_0^{L_x} h^3 \partial_{xx} \phi \partial_{xx} (\delta\phi) dx \tag{5.15}$$

$$\mathcal{I}_4 = \int_0^{L_x} h^2 \delta h (\partial_{xx} \phi)^2 dx \tag{5.16}$$

$$\mathcal{I}_5 = \int_0^{L_x} g h \delta h - g H \delta h dx \tag{5.17}$$

We now use integration by parts in three variables,

$$\int_{x_1}^{x_2} \alpha\beta \frac{d\gamma}{dx} dx = \alpha\beta\gamma|_{x_1}^{x_2} - \int_{x_1}^{x_2} \alpha\gamma \frac{d\beta}{dx} dx - \int_{x_1}^{x_2} \gamma\beta \frac{d\alpha}{dx} dx, \tag{5.18}$$

to obtain, noting that $\delta h, \delta\phi = 0$ at the boundaries,

$$\mathcal{I}_2 = \delta\phi h\partial_x\phi|_0^{L_x} - \int_{x_1}^{x_2} h\delta\phi\partial_{xx}\phi + \delta\phi\partial_x\phi\partial_x h dx \tag{5.19}$$

$$= -\int_{x_1}^{x_2} \delta\phi \left(h\partial_{xx}\phi + \partial_x\phi\partial_x h\right) dx \tag{5.20}$$

$$= -\int_{x_1}^{x_2} \delta\phi\partial_x \left(h\partial_x\phi\right) dx, \tag{5.21}$$

$$\mathcal{I}_3 = h^3\partial_{xx}\phi\partial_x\delta\phi|_0^{L_x} - \int_{x_1}^{x_2} h^3\partial_x\delta\phi\partial_{xxx}\phi + 3h^2\partial_x\delta\phi\partial_{xx}\phi dx \tag{5.22}$$

$$= -\mathcal{I}_{3,2} - \mathcal{I}_{3,2} \tag{5.23}$$

$$\mathcal{I}_{3,1} = \int_{x_1}^{x_2} h^3\partial_x\delta\phi\partial_{xxx}\phi dx \tag{5.24}$$

$$= h^3\partial_{xxx}\phi\delta\phi|_0^{L_x} - \int_{x_1}^{x_2} h^3\delta\phi\partial_{xxxx}\phi + 3h^2\delta\phi\partial_{xxx}\phi dx \tag{5.25}$$

$$= -\int_{x_1}^{x_2} \delta\phi \left(h^3\partial_{xxxx}\phi + 3h^2\partial_{xxx}\phi\right) dx \tag{5.26}$$

$$-\int_{x_1}^{x_2} \delta\phi\partial_x \left(h^3\partial_{xxx}\phi\right) dx \tag{5.27}$$

$$\mathcal{I}_{3,2} = 3h^2\partial_x\delta\phi\partial_{xx}\phi dx \tag{5.28}$$

$$= 3h^2\partial_{xx}\phi\delta\phi|_0^{L_x} - \int_{x_1}^{x_2} 3h^2\delta\phi\partial_{xxx}\phi + 6h\delta\phi\partial_{xx}\phi dx \tag{5.29}$$

$$= -\int_{x_1}^{x_2} \delta\phi\partial_x \left(3h^2\partial_{xx}\phi\right) dx \tag{5.30}$$

$$\Rightarrow \mathcal{I}_3 = \int_0^{L_x} \delta\phi\partial_x \left(h^3\partial_{xxx}\phi + 3h^2\partial_{xx}\phi\right) dx \tag{5.31}$$

$$= \int_0^{L_x} \delta\phi\partial_{xx} \left(h^3\partial_{xx}\phi\right) dx, \tag{5.32}$$

Combining these back into the main integral, we obtain

$$\int_0^T \int_0^{L_x} \delta h\partial_t\phi - \delta\phi\partial_t h 0\frac{1}{2}\delta h \left(\partial_x\phi\right)^2 - \delta\phi\partial_x \left(h\partial_x\phi\right)$$
$$- \frac{1}{3}\delta\phi\partial_{xx} \left(h^2\partial_{xx}\phi\right) - \frac{1}{2}h^2\delta h \left(\partial_{xx}\phi\right)^2 + g\delta h(h - H)dxdt. \tag{5.33}$$

Taking variations of this final equation (5.33) with respect to ϕ and h, we obtain

$$\delta\phi: \qquad \partial_t h - \partial_x \left(h\partial_x\phi\right) + \frac{1}{3}\partial_{xx} \left(h^3\partial_{xx}\phi\right) = 0 \tag{5.34}$$

$$\delta h: \qquad \partial_t\phi + \frac{1}{2} \left(\partial_x\phi\right)^2 + g(h - H) = 0, \tag{5.35}$$

which agree agree with the equations derived in Miles and Salmon (1985) (taking a one dimensional case with $P = 0$).

5.1 Adaptation for Numerical Implementation

The presence of the fourth order derivative of ϕ in the Green-Naghdi model introduces complications when it comes to numerical modelling with a finite element method, as many of the basis functions (such as the Lagrange Polynomials used in 9) are only C^0 continuous, i.e. higher derivatives introduce discontinuities at the edges of elements. There are two possible approaches to the solution of this issue, we could consider a basis that is C^1 continuous, such as cubic splines or Hermite polynomials, however these can be complex to calculate and implement. With this in mind, and as we are working in a single space dimension, we instead consider a method used by Kalogirou and Bokhove (2015) and introduce an *auxilliary variable*. We define this new variable

$$q = \partial_{xx}\phi, \tag{5.36}$$

and substitute into the variational principle (5.4). To preserve the equations under variation and ensure the system is still well-posed, we must also add some factor of q to the principle, so we introduce a further variable λ, be determined, and thus we have

$$0 = \delta \int_0^T \int_0^{L_x} h\partial_t\phi + \frac{1}{2}h\left(\partial_x\phi\right)^2 - \frac{1}{6}h^3q^2 + \frac{1}{2}g\eta^2 + \lambda q + \partial_x\lambda\partial_x\phi dxdt, \tag{5.37}$$

where we have also used $h^2 - 2hH = (h-H)^2 - H^2$, which, as the H^2 term is constant and thus disappears under variation, is equivalent to η^2. Now, taking variations with respect to our new variables ϕ, η, λ and q, we have

$$\delta\lambda : \qquad q = \partial_{xx}\phi \tag{5.38}$$

$$\delta h : \qquad \partial_t\phi + \frac{1}{2}\left(\partial_x\phi\right)^2 - \frac{1}{2}h^2q^2 + g\eta = 0 \tag{5.39}$$

$$\delta\phi : \qquad -\partial_t h - \partial_x\left(h\partial_x\phi\right) - \partial_{xx}\lambda = 0 \tag{5.40}$$

$$\delta q : \qquad \lambda - \frac{1}{3}h^3q = 0. \tag{5.41}$$

We now have an equation for λ, and have lowered the highest order derivative.

5.2 Simplification of Higher Order Nonlinear Terms

To further ease computation, we aim to reduce the higher order non-linear terms in the variational principle (9.19). To this end, as the quadratic and

cubic terms in h are also multiplied by derivatives of ϕ, and are likely to have relatively small variation from the rest depth H, we approximate (9.19) by

$$0 = \delta \int_0^T \int_0^{L_x} \eta \partial_t \phi + \frac{1}{2} H \left(\partial_x \phi\right)^2 - \frac{1}{6} H^3 q^2 + \frac{1}{2} g \eta^2 + \lambda q + \partial_x \lambda \partial_x \phi dx dt, \quad (5.42)$$

giving us a new principle in η, ϕ, q and λ, that should be significantly easier to compute while still preserving the leading order characteristics of the fluid motion. Again performing variations with respect to η, ϕ, q and λ, we now have

$$\delta\lambda : \qquad q = \partial_{xx}\phi \quad (5.43)$$

$$\delta\eta : \qquad \partial_t \phi + \frac{1}{2}\left(\partial_x \phi\right)^2 + g\eta = 0 \quad (5.44)$$

$$\delta\phi : \qquad -\partial_t h - Hq - \partial_{xx}\lambda = 0 \quad (5.45)$$

$$\delta q : \qquad \lambda - \frac{1}{3} h^3 q = 0. \quad (5.46)$$

5.3 Discretisation: The Galerkin-Ritz Method

We now seek to discretise equation (5.42) in space. To obtain a space discretisation of a variational principle, we use the so-called *Galerkin-Ritz Method* (Kalogirou and Bokhove, 2015), where we use the approximations

$$\phi(x,t) \approx \phi_h(x,t) \ = \phi_l(t) w_l(x) \quad (5.47)$$

$$\eta(x,t) \approx \eta_h(x,t) \ = \eta_k(t) w_k(x) \quad (5.48)$$

$$q(x,t) \approx q_h(x,t) \ \ = q_l(t) w_l(x) \quad (5.49)$$

$$\lambda(x,t) \approx \lambda_h(x,t) = \lambda_k(t) w_k(x), \quad (5.50)$$

$$(5.51)$$

for basis functions $w_i \in \mathcal{H}_0^1$.

Substituting these into (5.42), we have

$$\delta \int_0^T M_{kl}\eta_k \frac{d}{dt}\phi_l + H A_{kl}\phi_k\phi_l - \frac{1}{3}H^3 M_{kl} q_k q_l + g M_{kl}\eta_k\eta_l + M_{kl}\lambda_k q_l + A_{kl}\lambda_k\phi_l dt, \quad (5.52)$$

where

$$M_k l = \int_0^{L_x} w_k w_l dx \quad (5.53)$$

$$A_k l = \int_0^{L_x} \partial_x w_k \partial_x w_l dx. \quad (5.54)$$

Now taking variations with respect to ϕ_l, η_k, q_l and λ_k, we get

$$\delta\eta_k : \qquad M_{kl}\frac{d}{dt}\phi_l + gM_{kl}\eta_l = 0 \tag{5.55}$$

$$\delta\phi_l : \qquad -M_{kl}\frac{d}{dt}\eta_k + HA_{kl}\phi_k + A_{kl}\lambda_k = 0 \tag{5.56}$$

$$\delta\lambda_k : \qquad M_{kl}q_l + A_{kl}\phi_l = 0 \tag{5.57}$$

$$\delta q_l : \qquad -\frac{1}{3}H^3 M_{kl}q_k + M_{kl}\lambda_k = 0, \tag{5.58}$$

where the factor of $1/2$ disappears from some terms due to the symmetry of the matrices $\boldsymbol{M}$ and $\boldsymbol{A}$.

From this, we can obtain

$$\begin{aligned} M_{kl}\frac{d}{dt}\phi_l + gM_{kl}\eta_l - M_{kl}\dot{\eta}_k + HA_{kl}\phi_k + A_{kl}\lambda_k + M_{kl}q_l \\ + A_{kl}\phi_l - \frac{1}{3}H^3 M_{kl}q_k + M_{kl}\lambda_k = 0, \end{aligned} \tag{5.59}$$

a single matrix equation in all variables.

Time Discretisation

Similarly to in 8.5, we use the Crank Nicolson method to discretise in time, so we now have

$$\begin{aligned} M_{kl}\phi_l^n - M_{kl}\eta_k^n - \frac{\tau}{2}\Big(&gM_{kl}\eta_l^n + HA_{kl}\phi_k^n + A_{kl}\lambda_k^n \\ &+ M_{kl}q_l^n + A_{kl}\phi_l^n - \frac{1}{3}H^3 M_{kl}q_k^n + M_{kl}\lambda_k^n\Big) \\ = M_{kl}\phi_l^{n-1} - M_{kl}\eta_k^{n-1} + \frac{\tau}{2}\Big(&gM_{kl}\eta_l^{n-1} + HA_{kl}\phi_k^{n-1} + A_{kl}\lambda_k^{n-1} \\ &+ M_{kl}q_l^{n-1} + A_{kl}\phi_l^{n-1} - \frac{1}{3}H^3 M_{kl}q_k^{n-1} + M_{kl}\lambda_k^{n-1}\Big). \end{aligned} \tag{5.60}$$

5.4 Implementation

To implement (5.60), we define a solution vector $\boldsymbol{\Psi}$ and block matrices $\underline{\boldsymbol{M}}, \underline{\boldsymbol{A}}$ as

$$\boldsymbol{\Psi} = \begin{pmatrix} \boldsymbol{\phi} \\ \boldsymbol{\eta} \\ \boldsymbol{q} \\ \boldsymbol{\lambda} \end{pmatrix} \tag{5.61}$$

$$\underline{\boldsymbol{M}} = \begin{pmatrix} \boldsymbol{M} & 0 & 0 & 0 \\ 0 & \boldsymbol{M} & 0 & 0 \\ 0 & 0 & \boldsymbol{M} & 0 \\ 0 & 0 & 0 & \boldsymbol{M} \end{pmatrix} \tag{5.62}$$

$$\underline{\boldsymbol{A}} = \begin{pmatrix} \boldsymbol{A} & 0 & 0 & 0 \\ 0 & \boldsymbol{A} & 0 & 0 \\ 0 & 0 & \boldsymbol{A} & 0 \\ 0 & 0 & 0 & \boldsymbol{A} \end{pmatrix}. \tag{5.63}$$

We also define 'selection' matrices

$$\boldsymbol{S}_\phi = \begin{pmatrix} \boldsymbol{I} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{5.64}$$

$$\boldsymbol{S}_\eta = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & \boldsymbol{I} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{5.65}$$

$$\boldsymbol{S}_q = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \boldsymbol{I} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{5.66}$$

$$\boldsymbol{S}_\lambda = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \boldsymbol{I} \end{pmatrix}, \tag{5.67}$$

where, if N_{nodes} represents the number of nodes we have discretised over, $\boldsymbol{I}$ is the $N_{nodes} \times N_{nodes}$ identity matrix, and each zero in (5.64) represents a N_{nodes} square matrix of zeros. With this, the system we obtain is

$$\left(\underline{\boldsymbol{M}}\boldsymbol{S}_\phi - \underline{\boldsymbol{M}}\boldsymbol{S}_\eta - \frac{\tau}{2}\boldsymbol{F}\right)\boldsymbol{\Psi}^n = \left(\underline{\boldsymbol{M}}\boldsymbol{S}_\phi - \underline{\boldsymbol{M}}\boldsymbol{S}_\eta + \frac{\tau}{2}\boldsymbol{F}\right)\boldsymbol{\Psi}^{n-1}, \tag{5.68}$$

where

$$\boldsymbol{F} = g\underline{\boldsymbol{M}}\boldsymbol{S}_\eta + H\underline{\boldsymbol{A}}\boldsymbol{S}_\phi + \underline{\boldsymbol{A}}\boldsymbol{S}_\lambda - \frac{1}{3}H^3\underline{\boldsymbol{M}}\boldsymbol{S}_q + \underline{\boldsymbol{M}}\boldsymbol{S}_\lambda + \underline{\boldsymbol{M}}\boldsymbol{S}_q + \underline{\boldsymbol{A}}\boldsymbol{S}_\phi. \tag{5.69}$$

We use second order Lagrange polynomials (as discussed in chapter 9) as our basis functions, and use MATLAB to implement the finite element model on a uniform, one dimensional grid.

Chapter 6

Tidal Bores

6.1 Bores Around the World

Bores occur in many rivers and estuaries around the world, some notable examples are mapped in figure 6.1. Some of the most famous include the Severn bore, which can travel upstream as far as Gloucester, and is regularly surfed (see figure 6.2), the Mascaret on the Seine, which reached heights of up to 7.3 m, and almost disappeared after dredging in the 1960s, and the Hangzhou or "Silver Dragon", a powerful bore on the Qiantang River in China (Chanson, 2012, pp. 37-70). Also notable is the Pororoca in the Amazon, Brazil, which unusually develops initially offshore due to the bottom topography of the region, and can reach 5 m high and 25 km wide (Chanson, 2012, pp. 77-78).

The bore on the Trent, known as the Trent Aegir, which is the main focus of this project, is described in more detail in section 2.3

6.2 Theoretical Derivation

Fundamentally, a bore is a travelling hydraulic jump, and thus we explore here the theory behind hydraulic jumps. The classic stationary example is that of steady flow over a localised obstacle, which can be seen clearly in any shallow river where the water surface rises or dips due to an obstacle below the surface. Considering shallow flow in one dimension, and following Bokhove and Griffiths (2015), we seek exact nonlinear solutions to

$$\partial_t u + u\partial_x u + g\partial_x \eta = 0 \tag{6.1}$$

$$\partial_t(\eta + H) + \partial_x(\eta + H) = 0, \tag{6.2}$$

where $H(x)$ is the bottom profile, $\eta(x,t)$ is the free surface pertubation from the rest state, and $u(x,t)$ is the horizontal velocity. These are a form of the shallow water equations discussed in chapter 3.

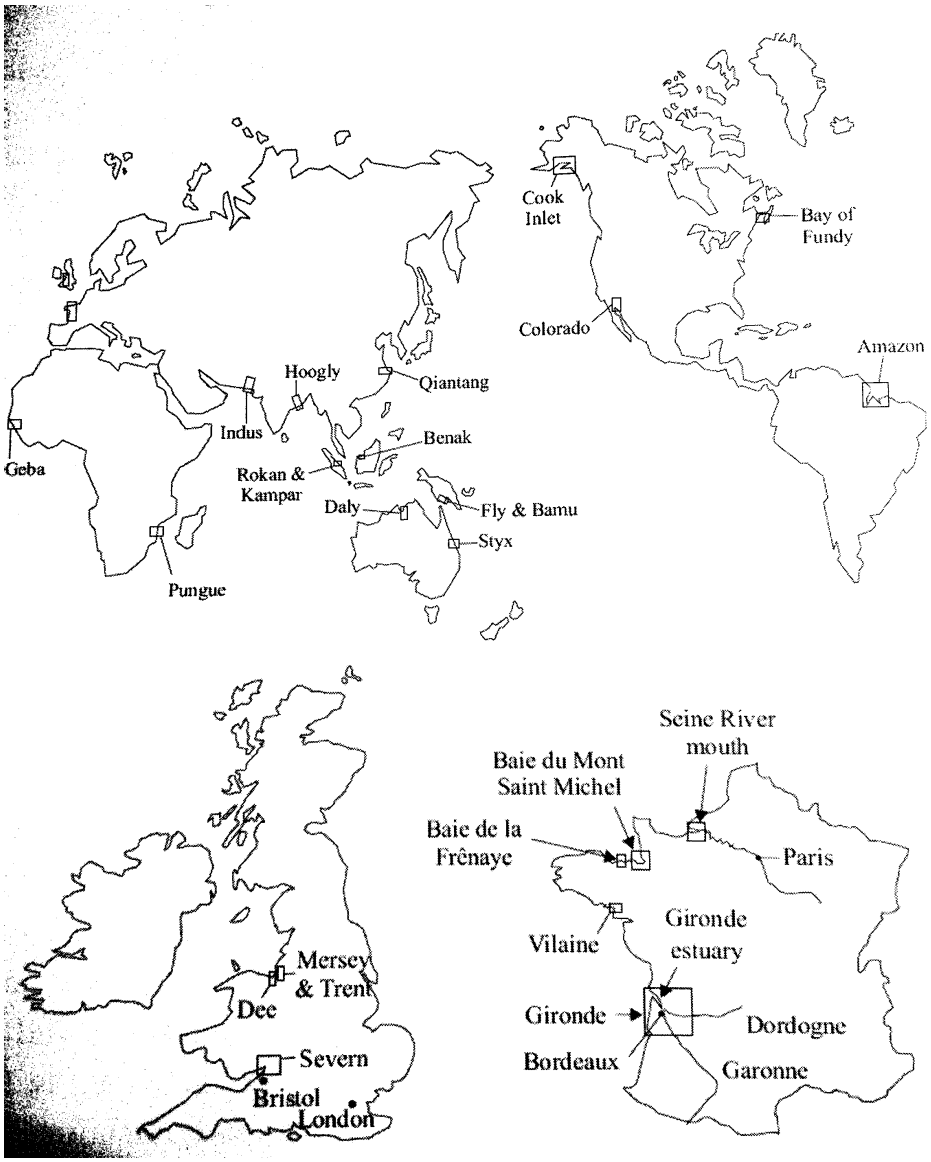

Figure 6.1: Map of Estuaries in which Bores are Observed Worldwide, from Chanson (2012, p. 38)

If we consider flow after such time as the flow has become steady, we can neglect any time derivatives in the equations (6.1)– (6.2). With this

Figure 6.2: Surfers on the Severn Bore, from Smith (2015).

approximation, we now have a system of ordinary differential equations in x,

$$\frac{d}{dx}((H+\eta)u) = 0; \tag{6.3}$$

$$\frac{d}{dx}\left(\frac{1}{2}u^2 + g\eta\right) = 0 \tag{6.4}$$

, so, after integration, and defining $Q = (H+\eta)U$, the volume flux, we have

$$(H+\eta)u = Q, \tag{6.5}$$

$$\frac{1}{2}\frac{Q^2}{(H+\eta)^2} + g\eta = C_1, \tag{6.6}$$

for C_1 some as yet unknown constant of integration.

However, we can now consider that the flow is assumed steady, so is constant both upstream ($H = H_0, u = U_0, \eta = 0$) and downstream of the obstacle. The volume flux is also constant, so $Q_0 = H_0U_0$ and we now have $C_1 = {}^{Q_0^2}/_{2H_0^2}$. Using the dimensionless coordinates we introduced in section 2.1, we can now introduce a new dimensionless parameter, the *Froude Number*. This number gives a ratio of the speed of the flow compared to the speed of propagation of a gravity wave in the domain, and can give important information about the flow profile. Defining $F_0 = {}^{U_0}/_{\sqrt{gH_0}}$, we can write (6.1)–(6.2) as (again

dropping the "hats" from the coordinates)

$$\partial_t u + u\partial_x u = -\frac{1}{F_0^2}\partial_x \eta \tag{6.7}$$

$$\partial_t (H+\eta) + \partial_x ((H+\eta)u) = 0. \tag{6.8}$$

Super– and Sub–Critical Flow

The Froude number is the analogue for incompressible flow of the more well-known *Mach number*, which applies to compressible fluids. Both parameters describe the transition of flow from one state to another, for the Mach number, $M_a > 1$ means the flow is *supersonic*, and $M_a < 1$ means *subsonic* flow (Bokhove and Griffiths, 2015). For the Froude number, $F_0 > 1$ indicates *supercritical* flow, and $F_0 < 1$ *subcritical* flow.

To examine the different effects of these conditions, we return to our nondimensional equations incorporating the Froude number, (6.7)–(6.8), and again consider steady flow, noticing that as we have non-dimensionalised, the values upstream are now equal to 1. We now have

$$(H+\eta)u = Q = 1 \tag{6.9}$$

$$\frac{1}{(H+\eta)^2} + \frac{2\eta}{F_0^2} = 1, \tag{6.10}$$

which, in conjunction with the differential equations given above, can be rearranged to give

$$\frac{d\eta}{dx} = \frac{F^2}{1-F^2}\frac{dH}{dx}, \tag{6.11}$$

where $F^2 = F_0^2 u^2/H+\eta$ is the *Froude variable*. If we assume the flow is again steady, we now have $F^2 = {}^{F_0^2}/_{D^3}$. We can see from (6.11) that if $F > 1$, i.e. supercritical flow, a negative gradient in H (i.e. a rise in bottom topography) would cause a positive gradient in η (i.e. a rise the surface), whereas for $F < 1$ (subcritical flow), the gradients would be the same sign, corresponding to a drop in the surface over an obstacle (Paterson, 1983, p.366). The final case to consider is when $F = 0$, i.e. *critical* flow. For this case, either ${}^{dH}/_{dx} = 0$, or we have a discontinuity or unsteadiness in the flow. We will now examine such situations.

The Critical Case

The conservative shallow water equations, (4.12), (4.13) and (4.14), can be written in one dimension, using terms including the Froude number after some

manipulation, giving

$$\partial_t((H+h)u) + \partial_x\left((H+h)u^2 + \frac{1}{2}\frac{(H+h)^2}{F_0^2}\right) + \frac{H+h}{F_0^2}\partial_x H = 0 \tag{6.12}$$

$$\partial_t h + \partial_x((H+h)u) = 0 \tag{6.13}$$

$$\tag{6.14}$$

(Bokhove and Griffiths, 2015). As we expect a discontinuity, normal derivatives will not be valid, so we use the fundamental theorem of calculus to take the limit of the integrals of (6.13) and (6.12) across the discontinuity. If we say the (moving) discontinuity occurs at $x = \chi(t)$, we have

$$\lim_{\epsilon\to 0}\int_{\chi(t)-\epsilon}^{\chi(t)+\epsilon} \partial_t h + \partial_x((H+h)u) = 0. \tag{6.15}$$

Now, the Leibniz rule can be used again to take the time derivative outside the integral of the first term, and the second term is a direct integral, so we have

$$\lim_{\epsilon\to 0}\frac{d}{dt}\int_{\chi(t)-\epsilon}^{\chi(t)+\epsilon} h dx - \lim_{\epsilon\to 0} h(x=\chi(t)-\epsilon)\frac{d\chi}{dt} - \lim_{\epsilon\to 0} h(x=\chi(t)+\epsilon)\frac{d\chi}{dt} + \lim_{\epsilon\to 0}((H+h)u)|_{\chi-\epsilon}^{\chi+\epsilon} = 0. \tag{6.16}$$

We can perform a similar series of operation on (6.13), and so we obtain

$$\lim_{\epsilon\to 0}\frac{d}{dt}\int_{\chi(t)-\epsilon}^{\chi(t)+\epsilon}(H+h)u dx - \lim_{\epsilon\to 0}\frac{d\chi}{dt}\left((H+h)u\right)|_{\chi-\epsilon}^{\chi+\epsilon} - + \lim_{\epsilon\to 0}|_{\chi-\epsilon}^{\chi+\epsilon} = 0. \tag{6.17}$$

If we notate the time derivative of χ (the speed of translation of the discontinuity) as ϑ, and use the "jump" notation $[f] = \lim_{\epsilon\to 0}(f(\chi+\epsilon) - f(\chi-\epsilon))$ as in Bokhove and Griffiths (2015), we can now write (6.16) and (6.17) as

$$-\vartheta[h] + [(H+h)u] = 0 \tag{6.18}$$

$$-\vartheta[(H+h)u] + \left[(H+h)u^2 + \frac{1}{2}\frac{(H+h)^2}{F_0^2}\right] = 0. \tag{6.19}$$

(6.18) and (6.19) are known as the Rankine-Hugoniot relations for the flow.

For a tidal bore on domain with flat bottom, we have constant H, but differing values of u either side of the boundary, giving rise to the bore situation.

In the Green-Naghdi model, Gagarina et al. (2013), working from a variational principle, give the Rankine-Hugoniot relations as

$$-\vartheta[h] + [u(h+H)] = 0 \tag{6.20}$$

$$-\vartheta[uh] + \left[(h+H)u^2 + \frac{g(h+H)^2}{2} - \frac{(h+H)^3}{3}\left(\partial_{xt}u - (\partial_x u)^2 + u\partial_{xx}u\right)\right] = 0. \tag{6.21}$$

6.3 Undular Bores

In practice, the discontinuity can take different forms. In bores wave breaking and sharp discontinuities are a characteristic of a higher Froude number, with shallower upstream water, and is sometimes confined to shallower regions at the edges of bores. The central region of the bore is then still noticeable, but does not break (this may also stretch across the whole river). This is known as an undular bore, and occurs with deeper water upstream. A photo of an undular section of bore is given in figure 6.3.

Figure 6.3: Undular Bore on the Dordogne River in France, with bore travelling from left to right. Photo by Pierre-Yves Lagrée, in Chanson and Koch (2006).

Chapter 7

Boundary Conditions

Also of importance is the choice of boundary conditions. We use differing boundary conditions for the different parts of the model, including conditions to couple the models.

7.1 2D Basin

For the first part of the model, the basin in two dimensions, we have used the linearised shallow water equations. To complete this system, we need to impose boundary conditions.

Closed Boundaries

For the side walls of the domain (the banks of the estuary), we impose solid, stationary wall boundary conditions. As we are not considering frictional terms, the imposed condition is that of no normal velocity - a *Dirichlet condition*. Mathematically, this is represented by

$$\boldsymbol{u}|_{\partial\Omega} \cdot \boldsymbol{n} = 0, \tag{7.1}$$

where $\boldsymbol{n}$ is a normal vector to the boundary. We also use this condition for initial testing of the model with a standing wave, by setting an initial oscillation in a completely enclosed domain.

Open Boundaries

At the tidal boundary, we have options for how we approach the tidal forcing. Ideally, we would seek a formulation that would allow waves to enter and leave the domain without reflection, however this is computationally expensive and complex to implement.

Clamped Tidal Conditions

The simplest form of tidal boundary condition is to "clamp" the value of either u or η on the boundary to a sinusoidal variation in time (or to a real world tide dataset).

Flather Conditions

Flather Conditions are a more nuanced option, first presented by Flather (1976), they extend the idea of a radiative boundary condition, using a gravity wave phase speed $\sqrt{gh}$ for the wavespeed. Using this along side a continuity equation, we obtain

$$\boldsymbol{u}_b \cdot \boldsymbol{n} = \boldsymbol{u}_f \cdot \boldsymbol{n} \pm \sqrt{\frac{g}{H}} \left(\eta_{b\pm1} - \eta_f\right), \tag{7.2}$$

where subscript b denotes a model value on the boundary and f a fixed tidal term (Carter and Merrifield, 2007). This condition aims to correct errors in volume flux, however it can be difficult to implement for finite element methods, so we stick to clamped conditions for the tidal forcing.

Internal Boundary

At the internal boundary, ideally we would have a method to transfer mass and momentum continuously across the boundary, as in Ambati et al. (2009) or, Kristina et al. (2014), however in the interests of ease of computation and the limited scope of this project, we instead make the assumption that the flow is primarily from the two dimensional basin into the one dimensional river (at least when we are considering bores). To this end, we use a Neumann condition for ϕ at both ends of the one–dimensional domain, with a river velocity in and out, i.e. $\partial_x \phi = U_{river}$. At the right, coupled end, we set $\eta_{GN} = \eta_{SWE}$, so the free surface is continuous across the boundary. Ideally, we would also have at term in ϕ at this right boundary. We also note that, as the river coupling point is small (a single node), while there will be reflected waves with our method, there would likely still be significant reflection in the real world, or a more accurate model.

Chapter 8

Finite Element Methods

The initial tidal basin of the estuary is modelled by the linearised shallow water equations, and implemented using a *finite element method*. In this section, the basics of this method will be demonstrated, and then the application to the model will be shown.

The finite element method offers considerable advantages over finite difference methods when it comes to the solution of partial differential equations on non-rectangular or more complicated domains, where the construction of a grid would be difficult (Cangiani and Georgoulis, 2012).

8.1 Weak Formulation

Integral to the concept of the finite element method is the idea of the *weak derivative*. To define this concept, we must revisit some linear algebra, and thus define the following terms:

Definition 8.1.1. *If we have a set $S \subset \mathbb{R}^d$ the* closure *of S is the smallest closed set C such that $S \in C$.*

Definition 8.1.2. *If we have a function $f : \Omega \to \mathbb{R}^d$, where the domain $\Omega \subset \mathbb{R}^d$ is an open set, then the* support *of the function f is given by the closure of the set of points in the domain where f is non-zero, i.e. $\{\omega \in \Omega | f(\omega) \neq 0\}$.*

Definition 8.1.3. *The support of a function is* compact *if it is a closed set, bounded everywhere.*

We notate the set of functions $f : \Omega \to \mathbb{R}^d$ on an open set $\Omega \subset \mathbb{R}^d$ with compact support that are also infinitely differentiable on Ω as $C_0^\infty(\Omega)$ (Cangiani and Georgoulis, 2012).

We can now define the weak derivative of some function $f : \Omega \to \mathbb{R}$ on the open domain Ω as follows:

Definition 8.1.4. *Let $g : \Omega \to \mathbb{R}^d$ be another function on the domain Ω. We can call g a* weak partial derivative *with respect to x_i of f if we have*

$$\int_\Omega f \partial_{x_i} \phi d\Omega = -\int_\Omega g\phi d\Omega, \tag{8.1}$$

for all $\phi \in C_0^\infty(\Omega)$ (Evans, 1998, p. 242).

The functions ϕ in equation (8.1) are known as *test functions*. We can use equation (8.1) to rewrite partial derivatives in *weak form*, also known as *variational form*, the first step in the finite element method.

We also define the following vector spaces, which will be used when formulating the problem later.

Definition 8.1.5. *The* Lebesgue space *of square integrable functions ϑ on the domain Ω is given by*

$$L^2(\Omega) = \{\vartheta | \int_\Omega |\vartheta|^2 d\Omega < \infty\} \tag{8.2}$$

(Adams and Fournier, 2003, p. 23).

Definition 8.1.6. *The* Hilbert space *of functions ϑ satisfying an arbitrary Dirichlet boundary condition $\vartheta|_{\partial\Omega_D} = G$, for some function G, is given by*

$$\mathcal{H}^1_G(\Omega) = \{\vartheta \in L^2(\Omega) | \partial_x \vartheta, \partial_y \vartheta \in L^2(\Omega) | \vartheta|_{\partial\Omega_D} = G\} \tag{8.3}$$

(Bokhove, 2014).

8.2 Mesh Generation

Compared to a finite difference method, where the solution to an equation is approximated on a (usually evenly spaced) grid of nodes, in the finite element method discretises the problem onto a series of *elements*, whose form depends on the choice of mesh and basis functions (which will be discussed in section 8.3).

For a two-dimensional problem, generally triangular or quadrilateral elements are used, so we must have a method of dividing the domain. For triangular elements, as used in this model, we can use the MATLAB function `initmesh`, which uses *Delauney Triangulation* to subdivide the domain. While the function is used as-is in the code for the model, we will now explore the theory behind this method.

Triangulation

We first must mathematically define what we mean by a triangulation, i.e. the subdivision of the domain into simplex elements.

Definition 8.2.1. *An* n–simplex *is the generalisation of a triangle into* n *dimensions. It can be thought of recursively as the* n *dimensional shape enclosed by* $n+1$ $(n-1)$*–simplices, noting that a 0–simplex is a single point (Coxeter, 1947, p. 120). We also note (for later use) that a* (-1)*–simplex is the empty set* $\emptyset$*.*

Example 8.1. *A straight line (1–simplex) is a one dimensional shape enclosed by two 0–simplices (points).*

Example 8.2. *A triangle (2–simplex) is a two-dimensional shape enclosed by three 1–simplices (straight lines).*

Example 8.3. *A tetrahedron (3–simplex) is a three-dimensional shape enclosed by four 2–simplices (triangles).*

If we have a domain $\Omega \subset \mathbb{R}^d$, and a set S of points in Ω, then we now define the set $\mathcal{T} = \{K_i, i = 1 : N_{elem}\}$ of triangular (for $d = 2$), tetrahedral (for $d = 3$) or higher order simplex (for $d > 3$) elements K_i.

Definition 8.2.2. *We can call* $\mathcal{T}$ *a* triangulation *of* Ω *if the following four conditions hold (George and Borouchaki, 1998, p. 14):*

- $V_e = S$*, where* $V_e = \{\cup K_{i,j}, i = 1 : N_{elem}, j = 1 : 3\}$ *is the union of the sets of the* j *vertices of each* K_i *element.*
- *The domain* Ω *is the union of all elements* K_i *in* $\mathcal{T}$*, i.e.* $\Omega = \{\cup K_i, i = 1 : N_{elem}\}$*.*
- $K_i \neq \emptyset, \forall K_i$*, i.e. no element* K_i *is empty.*
- $K_i \cap K_j$ *is an* n*–simplex, where* $n < d, i \neq j$*. This means that the intersection of two elements is either the empty set, or the sets meet at a vertex, edge, or higher order face.*

Delauney Triangulation

In 1850, Dirichlet showed the possibility of partitioning a set of points into convex cells, including a method for the restricting the proximity of such cells. This proximity criterion was then applied by Voronoi (1908) to develop the *Voronoï Diagram*, a visual representation of a subdivision on the domain, where edges/faces represent a division into regions of proximity to each point. Each line on the diagram is a line of equal distance between the nearest two points, and each cell is the locus of coordinates closer to the point within the cell than to all other points (George and Borouchaki, 1998, p. 36). Mathematically, we can define the Voronï diagram, following George and Borouchaki (1998, pp. 35-36), as follows:

Definition 8.2.3. *If we have a set of points*

$$S = \{\boldsymbol{\alpha}_i = \begin{pmatrix} \alpha_{i,x_1} \\ \alpha_{i,x_2} \\ \vdots \\ \alpha_{i,x_d} \end{pmatrix}, i = 1:1:n\}, \tag{8.4}$$

in d dimensions, then the associated Voronoï diagram is the set of cells given by

$$\boldsymbol{V}_i = \{\boldsymbol{q} \quad | \quad \|\boldsymbol{q} - \boldsymbol{\alpha}_i\|_2 \leq \|\boldsymbol{q} - \boldsymbol{\alpha}_j\|_2, \forall j \neq i, \boldsymbol{\alpha}_i, \boldsymbol{\alpha}_j \in S\}, \tag{8.5}$$

where $\|\boldsymbol{\alpha}\|_p = \left(\sum_{i=1}^{n} |\boldsymbol{\alpha}|^p\right)^{1/p}$ *is the* $\mathcal{L}_p$ *norm.*

Equation (8.5) can take different forms if a different distance metric (and thus a different value of p in the $\mathcal{L}_p$ norm) is used (here we use the Euclidean distance, i.e. $p = 2$). Delauney (1934) then proved the existence and uniqueness of a triangulation from the Voronoï diagram, related to the *empty sphere criterion*. An example of a Voronoï diagram and the associated Delaunay Triangulation is shown in figure 8.1.

Definition 8.2.4. *The* empty sphere criterion *in two dimensions states that, defining* C_i *to be the closed circle circumscribing the element* K_i *in triangulation* $\mathcal{T}$*, then if* C_i *contains only the vertices of* K_i *and no other vertices, for all* K_i*, then* $\mathcal{T}$ *is a* Delauney Triangulation *(George and Borouchaki, 1998, pp. 18-19).*

This can be extended to n dimensions by replacing the 2 dimensional circle (a 1–sphere) with higher order $(n-1)$–spheres (similarly to the simplices discussed earlier).

8.3 Shape/Basis Functions

Now we have an idea of the weak formulation, and have obtained a triangulation of the domain, we must look for a choice of test functions for the finite element method. These basis or shape functions are used to interpolate the solution between the nodal points, and are selected such that the ith function ϕ_i is equal to 1 at the node $\boldsymbol{p}_i$, and zero at all other nodes. As we are using triangular elements in two dimensions, we choose to use linear triangles as our basis functions. For ease of later integration, it is useful to think of a canonical triangular element $\hat{K}$, and then derive a method for transforming each element K_i to this canonical element. The three vertices of each K_i are numbered anticlockwise by convention, and we have (from the output of MATLAB's `initmesh`) an array giving us the co-ordinates (in $(x-y)$–space) of the nodal points, and a further "index" array giving which points correspond to each vertex of each element K_i.

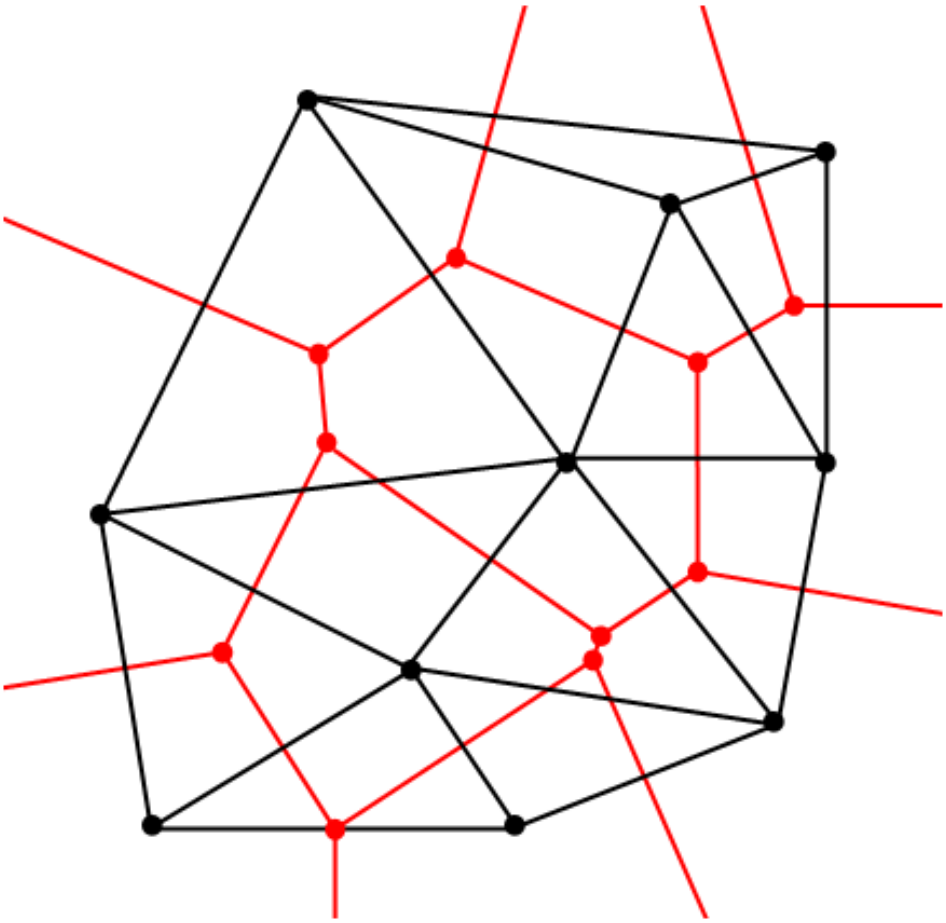

Figure 8.1: Voronoï diagram (red lines) of a set of points (black circles), and the associated Delaunay triangulation (black lines). From (University of Western Australia, School of Computer Science and Software Engineering, 2010)

Natural Coordinates

To transform K_i into the canonical element $\hat{K}$, we perform a co-ordinate transfer from the cartesian (x, y) into the *natural co-ordinates* (ζ_1, ζ_2) (Bokhove, 2014). Within these coordinates, the triangular element $\hat{K}$ is defined by the nodes $(\zeta_1, \zeta_2) = (0, 0), (1, 0), (0, 1)$, (i.e. a right-angled triangle) which gives shape functions

$$\chi_0(\zeta_1, \zeta_2) = 1 - \zeta_1 - \zeta_2, \tag{8.6}$$

$$\chi_1 = \zeta_1, \tag{8.7}$$

$$\chi_2 = \zeta_2. \tag{8.8}$$

These shape functions are used to provide the mapping between $(x, y) \to (\zeta_1, \zeta_2)$ for each element K_i, which is given by

$$(x, y) = \sum_{j=0}^{2} (x, y)_{i,j} \chi_j(\zeta_1, \zeta_2), \tag{8.9}$$

where $(x, y)_{i,j}$ are the cartesian coordinates of the jth vertex of the ith element (Bokhove, 2014). To perform the transformation between cartesian and natural coordinates for the derivatives and integrals, we must compute the *Jacobian* of the transform.

Definition 8.3.1. *The* Jacobian *of a transformation between two systems of two-dimensional coordinates* $(x, y) \to (\zeta_1, \zeta_2)$ *is given by*

$$\boldsymbol{J} = \begin{pmatrix} \partial_{\zeta_1} x & \partial_{\zeta_2} x \\ \partial_{\zeta_1} y & \partial_{\zeta_2} y \end{pmatrix}. \tag{8.10}$$

For natural triangular coordinates, we have

$$\begin{aligned} \partial_{\zeta_1} x &= x_1 - x_0, \qquad \partial_{\zeta_2} x = x_2 - x_0, \\ \partial_{\zeta_1} y &= y_1 - y_0, \qquad \partial_{\zeta_2} y = y_2 - y_0, \end{aligned} \tag{8.11}$$

so our Jacobian is

$$\boldsymbol{J} = \begin{pmatrix} x_1 - x_0 & x_2 - x_0 \\ y_1 - y_0 & y_2 - y_0 \end{pmatrix}. \tag{8.12}$$

To convert the derivatives, we use the chain rule for partial derivatives, to obtain, for some function ϑ:

$$\partial_{\zeta_1} \vartheta = \partial_x \vartheta \partial_{\zeta_1} x + \partial_y \vartheta \partial_{\zeta_1} y, \tag{8.13}$$

$$\partial_{\zeta_2} \vartheta = \partial_x \vartheta \partial_{\zeta_2} x + \partial_y \vartheta \partial_{\zeta_2} y, \tag{8.14}$$

$$\tag{8.15}$$

or, in matrix form:

$$\begin{pmatrix} \partial_{\zeta_1} \\ \partial_{\zeta_2} \end{pmatrix} \vartheta = \begin{pmatrix} \partial_{\zeta_1} x & \partial_{\zeta_1} y \\ \partial_{\zeta_2} x & \partial_{\zeta_2} y \end{pmatrix} \begin{pmatrix} \partial_x \\ \partial_y \end{pmatrix} \vartheta = \boldsymbol{J}^T \begin{pmatrix} \partial_x \\ \partial_y \end{pmatrix} \vartheta. \tag{8.16}$$

Therefore, we have

$$\begin{pmatrix} \partial_x \\ \partial_y \end{pmatrix} \vartheta = (\boldsymbol{J}^T)^{-1} \begin{pmatrix} \partial_{\zeta_1} \\ \partial_{\zeta_2} \end{pmatrix} \vartheta. \tag{8.17}$$

8.4 Formulation of the Two–Dimensional Shallow Water Problem

Recall from section 3.3 that we are using here the linearised shallow water equations

$$\partial_t \boldsymbol{u} + g\nabla h - f\boldsymbol{e}_z \times \tilde{\boldsymbol{u}} = 0, \tag{8.18}$$

$$\partial_t h + H(\partial_x u + \partial_y v) = 0. \tag{8.19}$$

We must now formulate this problem in weak form, for the purposes of applying the finite element method. We first multiply by a test function $\phi \in \mathcal{H}_0^1$, and then integrate over Ω, thus obtaining

$$\int_\Omega (\partial_t u - fv)\phi d\Omega + g\int_\Omega (\partial_x h)\phi d\Omega = 0 \tag{8.20}$$

$$\int_\Omega (\partial_t v + fu)\phi d\Omega + g\int_\Omega (\partial_y h)\phi d\Omega = 0 \tag{8.21}$$

$$\int_\Omega (\partial_t h)\phi d\Omega - H\int_\Omega \nabla\phi \cdot \boldsymbol{u} d\Omega = -\int_{\partial\Omega} \boldsymbol{u}\cdot\boldsymbol{n}\phi d\boldsymbol{S}, \tag{8.22}$$

where, as in Nurijanyan (2013), we perform integration by parts on the h equation only (8.22), for the purposes of energy conservation. When we have the Dirichlet boundary condition that $\boldsymbol{u}\cdot\boldsymbol{n} = 0$, the right hand side of the h equation then disappears.

For later ease of computation, we now seek to combine equations (8.18) and (8.19) into a single vector equation. To this end, we define a vector function

$$\boldsymbol{\Psi}(x,y,t) = \begin{pmatrix} u(x,y,t) \\ v(x,y,t) \\ h(x,y,t) \end{pmatrix}. \tag{8.23}$$

Using this, we now have the weak formulation for our equation: Find $PSI \in \mathcal{H}_G^1(\Omega)$ such that $\forall\phi \in \mathcal{H}_0^1$, $\boldsymbol{\Psi}$ satisfies

$$\int_\Omega \phi\partial_t\boldsymbol{\Psi} d\Omega + \int_\Omega (\boldsymbol{A}\boldsymbol{\Psi})\phi d\Omega + \int_\Omega (\boldsymbol{f}\boldsymbol{\Psi})\phi d\Omega = 0, \tag{8.24}$$

where the matrices $\boldsymbol{A}$ and $\boldsymbol{f}$ are given by

$$\boldsymbol{A} = \begin{pmatrix} 0 & 0 & g\partial_x \\ 0 & 0 & g\partial_y \\ H\partial_x & H\partial_y & 0 \end{pmatrix} \tag{8.25}$$

$$\boldsymbol{f} = \begin{pmatrix} 0 & -f & 0 \\ f & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}. \tag{8.26}$$

8.5 Discretisation

To solve this using a computer, we now must discretise equation (8.24) both in time, and in space onto our mesh.

Space Discretisation

To discretise in space, we approximate our functions u, v, h (and thus $\mathbf{\Psi}$) by the sum of a series of nodal values u_j, v_j, h_j (and $\mathbf{\Psi}_j$) multiplied by a basis function w_j at each node:

$$u(x,y,t) \approx u_d(x,y,t) = u_j(t)w_j(x,y), \tag{8.27}$$
$$v(x,y,t) \approx v_d(x,y,t) = v_j(t)w_j(x,y), \tag{8.28}$$
$$h(x,y,t) \approx h_d(x,y,t) = h_j(t)w_j(x,y), \tag{8.29}$$
$$\Rightarrow \mathbf{\Psi}(x,y,t) \approx \mathbf{\Psi}_d(x,y,t) = \mathbf{\Psi}_j(t)w_j(x,y), \tag{8.30}$$

where Einstein notation is used for summation. (Bokhove, 2014). If we take w_i to be a basis with compact support, we can take $\phi = w_i$ for $i = 1:1:N_{nodes}$. Using this within equations (8.20) to (8.22), we now have a system of equations where u, v, h (dropping the subscript d) can be taken out of the integral

$$\int_{\mathcal{T}} w_i w_j d\Omega \partial_t u_j - f\int_{\mathcal{T}} w_i w_j d\Omega v_j + g\int_{\mathcal{T}} w_i \partial_x w_j d\Omega h_j = 0 \tag{8.31}$$
$$\int_{\mathcal{T}} w_i w_j d\Omega \partial_t v_j + f\int_{\mathcal{T}} w_i w_j d\Omega u_j + g\int_{\mathcal{T}} w_i \partial_y w_j d\Omega h_j = 0 \tag{8.32}$$
$$-\int_{\mathcal{T}} w_i w_j d\Omega \partial_t h_j + H\left(\int_{\mathcal{T}} w_j \partial_x w_i d\Omega u_j + \int_{\mathcal{T}} w_j \partial_y w_i d\Omega v_j\right) = 0, \tag{8.33}$$

working in a similar fashion to Nurijanyan (2013). We can define the matrices $\boldsymbol{M}, \boldsymbol{S}^x, \boldsymbol{S}^y$, with elements $M_{ij}, S^x_{ij}, S^y_{ij}$ respectively given by

$$M_{ij} = \int_{\mathcal{T}} w_i w_j d\Omega, \tag{8.34}$$
$$S^x_{ij} = \int_{\mathcal{T}} w_i \partial_x w_j d\Omega, \tag{8.35}$$
$$S^y_{ij} = \int_{\mathcal{T}} w_i \partial_y w_j d\Omega. \tag{8.36}$$

Combining equations (8.31) to (8.33), we can obtain a matrix equation in terms of $\mathbf{\Psi}$:

$$\underline{\boldsymbol{M}}\partial_t \mathbf{\Psi} + \underline{\boldsymbol{f}}\mathbf{\Psi} + \underline{\boldsymbol{S}}\mathbf{\Psi} = 0, \tag{8.37}$$

where $\underline{\boldsymbol{M}}, \underline{\boldsymbol{f}}$ and $\underline{\boldsymbol{S}}$ are block matrices, whose form depends on our formulation of the discretised $\boldsymbol{\Psi}$. As $\boldsymbol{\Psi} = (u, v, h)^T$, we now take

$$\boldsymbol{\Psi} = \begin{pmatrix} u_1 \\ \vdots \\ u_{N_{nodes}} \\ v_1 \\ \vdots \\ v_{N_{nodes}} \\ h_1 \\ \vdots \\ h_{N_{nodes}} \end{pmatrix}, \tag{8.38}$$

where the numbered subscripts represent the node at which that quantity is approximated. With this form for $\boldsymbol{\Psi}$, we now define

$$\underline{\boldsymbol{M}} = \begin{pmatrix} \boldsymbol{M} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{M} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & -\boldsymbol{M} \end{pmatrix}, \tag{8.39}$$

$$\underline{\boldsymbol{S}} = \begin{pmatrix} \boldsymbol{0} & \boldsymbol{0} & g\boldsymbol{S}^x \\ \boldsymbol{0} & \boldsymbol{0} & g\boldsymbol{S}^y \\ H(\boldsymbol{S}^x)^T & H(\boldsymbol{S}^y)^T & \boldsymbol{0} \end{pmatrix}, \tag{8.40}$$

$$\underline{\boldsymbol{f}} = \begin{pmatrix} \boldsymbol{0} & -f\boldsymbol{M} & \boldsymbol{0} \\ f\boldsymbol{M} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \end{pmatrix}, \tag{8.41}$$

$$\tag{8.42}$$

where $\boldsymbol{0}$ is the $N_{nodes} \times N_{nodes}$ matrix of zeros, and $\boldsymbol{M}, \boldsymbol{S}^x, \boldsymbol{S}^y$ are as given above. The transposition of the $\boldsymbol{S}$ matrices in the formation of $\underline{\boldsymbol{S}}$ is to take into account the reversed indices in the form of (8.33).

Energy of the System

To analyse the effectiveness of model, and the time schemes used in the next section, we must examine the energy in the model. A good model will conserve energy (provided there are no sources or sinks, and the boundary is closed). Returning to the discretised u, v, h equations (8.31),(8.32),(8.33), and following Nurijanyan (2013), we consider the case without the coriolis effect (i.e. $f = 0$). Multiplying (8.31) by u_i, (8.32) by v_i and (8.33) by h_i, and performing a sum over all nodes i, j, we obtain (after some manipulation, given in appendix A)

$$\frac{d}{dt}\left(HM_{ij}(u_i u_j + v_i v_j) + gM_{ij}h_i h_j\right) = 0, \tag{8.43}$$

which signifies that total energy should remain constant in time. The first part of (8.43) denotes the kinetic energy, and the second part the gravitational potential.

Time-Stepping Schemes

We must now discretise in time, to remove the ∂_t derivative in equation (8.37).

The Backwards Euler Method

One of the simplest methods for this discretisation is the *backwards/implicit Euler* scheme. In this method, we approximate the time derivative with a combination of the solution at time t (notated as $\mathbf{\Psi}^n$) and the solution at time $t-\tau$ (notated as $\mathbf{\Psi}^{n-1}$), where τ is a time step, as follows

$$\partial_t \mathbf{\Psi} \approx \frac{\mathbf{\Psi}^n - \mathbf{\Psi}^{n-1}}{\tau}. \tag{8.44}$$

With this method used, equation (A.3) can now be written at each timestep as

$$\underline{\boldsymbol{M}}\mathbf{\Psi}^n + \tau(\underline{\boldsymbol{f}}\mathbf{\Psi}^n + \underline{\boldsymbol{S}}\mathbf{\Psi}^n) = \underline{\boldsymbol{M}}\mathbf{\Psi}^{n-1}. \tag{8.45}$$

The backwards Euler method is unconditionally stable, however is only first-order accurate with respect to τ (Cangiani and Georgoulis, 2012). This method also introduces numerical dissipation, as can be seen by examining the energy of the system for the standing wave case. Plotting the inner part of (8.43) (i.e. $HM_{ij}(u_iu_j+v_iv_j)+gM_{ij}h_ih_j$), both as separate kinetic and potential parts and as the total, we obtain figure 8.2, where the decay in energy is clearly seen.

The Crank-Nicolson Method

We thus seek a slightly better method for the time-stepping, for example the *Crank-Nicolson* scheme. This scheme combines the implicit Euler scheme mentioned above with an explicit Euler scheme (which can be unstable if used alone). If we rearrange our equation into the form

$$\partial_t \mathbf{\Psi} = F(\mathbf{\Psi}, \partial_x \mathbf{\Psi}, \partial_y \mathbf{\Psi}), \tag{8.46}$$

then the Crank-Nicolson method approximates ∂_t with

$$\frac{\mathbf{\Psi}^n - \mathbf{\Psi}^{n-1}}{\tau} = \frac{1}{2}(F^n + F^{n-1}) \tag{8.47}$$

(Cangiani and Georgoulis, 2012). Using this in equation (A.3) again, we obtain:

$$\underline{\boldsymbol{M}}\mathbf{\Psi}^n + \frac{\tau}{2}(\underline{\boldsymbol{S}}\mathbf{\Psi}^n + \underline{\boldsymbol{f}}\mathbf{\Psi}^n) = \underline{\boldsymbol{M}}\mathbf{\Psi}^{n-1} - \frac{\tau}{2}(\underline{\boldsymbol{S}}\mathbf{\Psi}^{n-1} + \underline{\boldsymbol{f}}\mathbf{\Psi}^{n-1}). \tag{8.48}$$

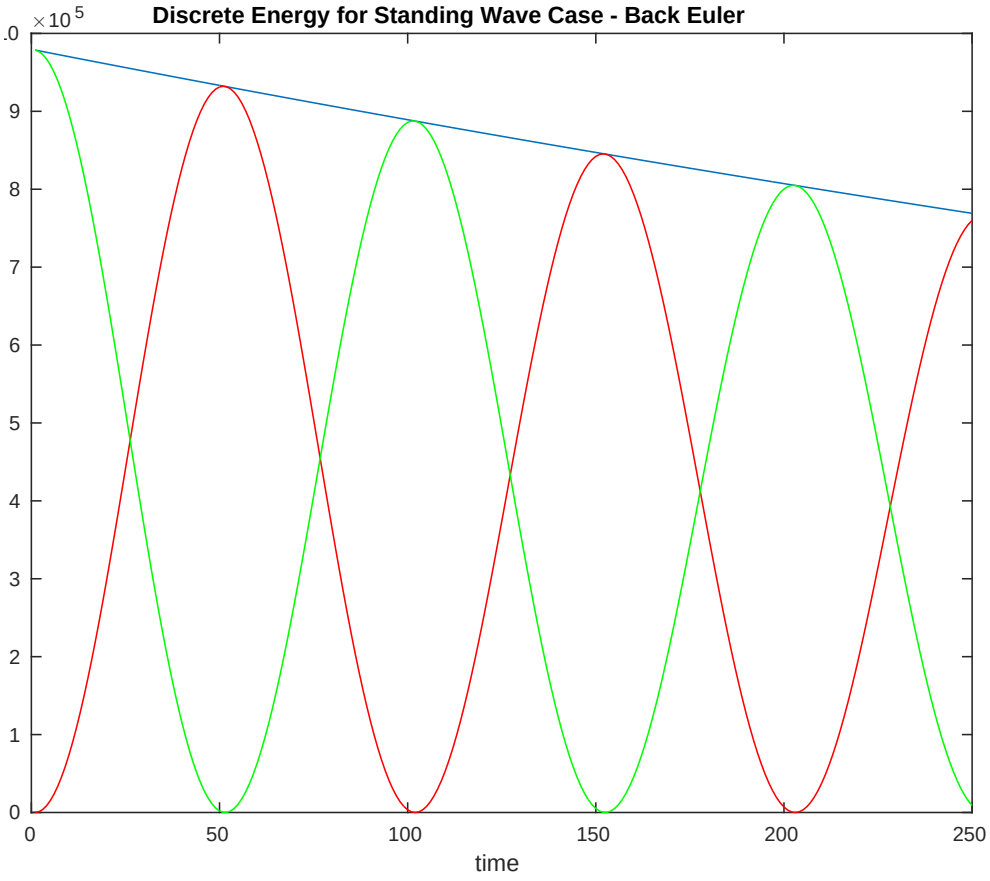

Figure 8.2: Energy Plot for the Backwards Euler Method. Red is Kinetic Energy, Green is Gravitational Potential Energy and Blue is the Total Energy.

This can be rearranged to give

$$(\underline{\boldsymbol{M}} + \frac{\tau}{2}(\underline{\boldsymbol{S}} + \underline{\boldsymbol{f}}))\boldsymbol{\Psi}^n = (\underline{\boldsymbol{M}} - \frac{\tau}{2}(\underline{\boldsymbol{S}} + \underline{\boldsymbol{f}}))\boldsymbol{\Psi}^{n-1}, \tag{8.49}$$

which is a linear system of equations in the form $Ax = b$, where A is a $3N_{nodes} \times 3N_{nodes}$ *mass matrix*, x is the solution vector of length $3N_{nodes}$ and b is a vector of length $3N_{nodes}$, and thus can be solved in MATLAB with the `mldivide` program (or in other programs by methods such as Gaussian elimination or LU decomposition).

The energy in this method does not decay like the backwards Euler method, as can be seen from figure 8.3.

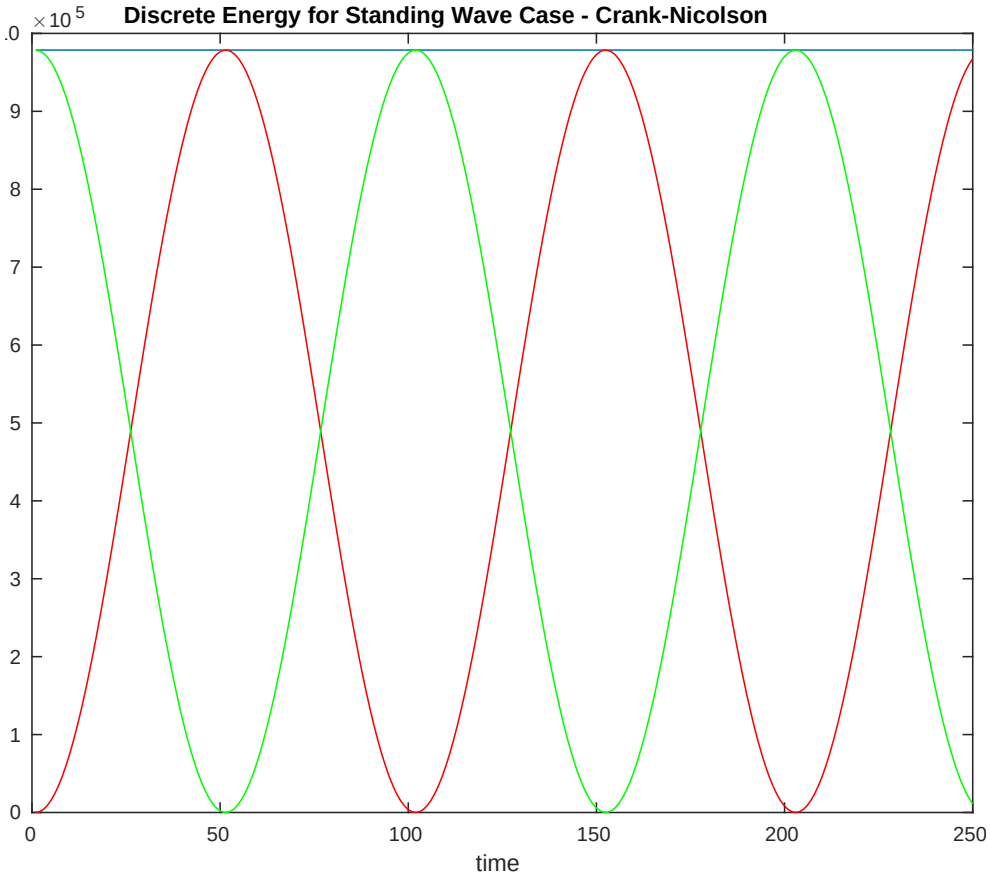

Figure 8.3: Energy Plot for the Crank-Nicolson Method. Red is Kinetic Energy, Green is Gravitational Potential Energy and Blue is the Total Energy.

8.6 Imposing Boundary Conditions

We now must consider how to impose the boundary conditions on the model domain. Some Neumann and Dirichlet conditions are taken care of within the weak formulation of the problem, however others are imposed after the fact, and are more complex to implement.

For these conditions, we must modify both the mass matrix (with elements $a_{ij}, i, j = 1 : 3N_{nodes}$) and the RHS vector (with elements $b_j, j = 1 : 3N_{nodes}$). Following Computational Science Education Project (1995, pp. 20-21), we use the following algorithm for each boundary node, presuming the value of u at the ith node (u_i) is known:

- For each element $b_j, j = 1 : N_{nodes}, j \neq i$ of the RHS vector, set $b_j = b_j - A_{ij}u_i$.

- Set all elements in the ith row and column of the mass matrix to zero.
- Set $a_{ii} = 0$.
- Set b_i=u_i.

If we know a value of v at the boundary node, we do the same algorithm, however we replace i and j with $i + N_{nodes}$ and $j + N_{nodes}$ respectively, to ensure that the part of the mass matrix and RHS vector acting on the v values is affected. Similarly, for a known value of h, we replace i and j with $i + 2N_{nodes}$ and $j + 2N_{nodes}$ respectively.

For imposing clamped tidal conditions, we proceed as above, however the known values of u,v and h may themselves need calculating at each time step before inclusion in the algorithm, or may be read from a data file of real-world tidal data.

Chapter 9

One Dimensional Finite Elements: The Lagrange Polynomial Basis

For a k times differentiable, one dimensional basis, we use *Lagrange polynomials*. These are polynomials $l_i(x)$ chosen for the property that, at each node x_k,

$$l_i(x_k) = \delta_{ik}, \tag{9.1}$$

where δ_{ik} is the *Kronecker delta*, defined as

$$\delta_{ik} = \begin{cases} 0 & i \neq j \\ 1 & i = j \end{cases} \tag{9.2}$$

(Kelmanson, 2014).

Each (kth order) Lagrange polynomial l_i is given by

$$l_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{k} \left(\frac{x - x_j}{x_i - x_j} \right). \tag{9.3}$$

For use as a basis for finite element analysis, we divide each element into $k+1$ nodes (including the endpoints of the element), and then map this element (and nodes) onto a reference element with coordinate $\xi \in [-1, 1]$ (Flaherty, 2005). The linear transformation from an element with $x \in [x_{j-1}, x_j]$ to $\xi \in [-1, 1]$ is given by

$$x(\xi) = \frac{1}{2}(x_j + x_{j-1}) + \frac{1}{2}(x_j - x_{j-1})\xi, \tag{9.4}$$

with shape functions $l_i(\xi)$.

For use as a basis, we must also know the derivatives of these functions. The transformation from x to ξ introduces a factor of $h/2$ into the integrals, (where $h = x_j - x_{j-1}$) and the derivatives transform as

$$\partial_\xi = \frac{1}{x_j}\partial_x. \tag{9.5}$$

Using this, we define, for kth order polynomials

$$\chi_i^k = \prod_{\substack{j=0 \\ j\neq i}}^{k} (\xi_i - \xi_j), \tag{9.6}$$

which does not change with differentiation, for ease of notation.

Example 9.1. *As an example, we expand the fourth order Lagrange polynomial for $i = 0$, given by*

$$l_0^4(\xi) = \frac{(\xi-\xi_1)(\xi-\xi_2)(\xi-\xi_3)(\xi-\xi_4)}{\chi_0} \tag{9.7}$$

$$\begin{aligned}\partial_\xi l_0^4(\xi) = \frac{1}{\chi_0^4}\big((\xi-\xi_1)(\xi-\xi_2)(\xi-\xi_3) + (\xi-\xi_1)(\xi-\xi_2)(\xi-\xi_4) \\ + (\xi-\xi_1)(\xi-\xi_3)(\xi-\xi_4) + (\xi-\xi_2)(\xi-\xi_3)(\xi-\xi_4)\big)\end{aligned} \tag{9.8}$$

$$\begin{aligned}\partial_{\xi\xi} l_0^4(\xi) = \frac{2}{\chi_0^4}\big(\xi_1(\xi_2+\xi_3+\xi_4-3\xi) + \xi_2(\xi_3+\xi_4-3\xi) \\ +\xi_3\xi_4 - 3\xi_3\xi - 3\xi_4\xi + 6\xi^2\big)\end{aligned} \tag{9.9}$$

$$\partial_{\xi\xi\xi} l_0^4(\xi) = \frac{-6}{\chi_0^4}(\xi_1+\xi_2+\xi_3+\xi_4-4\xi) \tag{9.10}$$

$$\partial_{\xi\xi\xi\xi} l_0^4(\xi) = \frac{24}{\chi_0^4}. \tag{9.11}$$

For the basis of the Green-Naghdi finite element model, we use second order elements on an evenly spaced grid. This means that h is a constant. The second order Lagrange polynomials $l_i^2, i = 0:2$ on the reference $[-1, 1]$

are as follows:

$$l_0^2(\xi) = \frac{(\xi - \xi_1)(\xi - \xi_2)}{\chi_0^2} \tag{9.12}$$

$$= \frac{\xi^2 - \xi}{2} \tag{9.13}$$

$$l_1^2(\xi) = \frac{(\xi - \xi_0)(\xi - \xi_2)}{\chi_1^2} \tag{9.14}$$

$$= \frac{\xi^2 - 1}{-1} \tag{9.15}$$

$$= 1 - \xi^2 \tag{9.16}$$

$$l_2^2(\xi) = \frac{(\xi - \xi_0)(\xi - \xi_1)}{\chi_2^2}, \tag{9.17}$$

$$= \frac{\xi^2 + \xi}{2} \tag{9.18}$$

and their derivatives are

$$\partial_\xi l_0^2(\xi) = \frac{2\xi - (\xi_1 + \xi_2)}{\chi_0^2} \tag{9.19}$$

$$= \frac{2\xi - 1}{2} \tag{9.20}$$

$$\partial_\xi l_1^2(\xi) = \frac{2\xi - (xi_0 + \xi_1)}{\chi_1^2} \tag{9.21}$$

$$= \frac{2\xi}{-1} \tag{9.22}$$

$$= -2\xi \tag{9.23}$$

$$\partial_\xi l_2^2(\xi) = \frac{2\xi - (xi_0 + \xi_1)}{\chi_2^2} \tag{9.24}$$

$$= \frac{2\xi + 1}{2}, \tag{9.25}$$

A similar method of element and node numbering and "hat" elements as used in the 2D triangular case is then used to build the mass and stress matrices for the finite element problem. The algorithm used is

1: **for all** elements e **do**
2: **for** $\alpha = 1 : 3$ **do**
3: $k := index(\alpha, e)$
4: **for** $\beta = 1 : 3$ **do**
5: $l =: index(\beta, e)$
6: $M_{kl} = M_{kl} + \hat{M}_{\alpha\beta}$
7: $A_{kl} = A_{kl} + \hat{A}_{\alpha\beta}$
8: **end for**
9: **end for**

10: **end for**

with

$$\hat{M}_{\alpha\beta} = \frac{h}{2}\int_{-1}^{1} w_\alpha(\xi) w_\beta(\xi) d\xi \tag{9.26}$$

$$\hat{A}_{\alpha\beta} = \frac{h}{2}\int_{-1}^{1} \partial_\xi w_\alpha(\xi) \partial_\xi w_\beta(\xi) d\xi \tag{9.27}$$

Chapter 10

Results and Model Output

10.1 Testing the Two–Dimensional Model

We first test the two dimensional model alone, with a simplified, rectangular boundary and no Coriolis effects. A standing wave solution (figure 10.1), such as that obtained when we use an initial condition $h(x, y, 0) = cos\,({}^{m\pi x}/_{L_x})$, is easy to find. In this case, we have

$$h = \alpha \cos\left(\frac{m\pi x}{L_x}\right)\cos(2\pi\omega t), \tag{10.1}$$

for amplitude α, $i = 1, 2, 3\ldots$, and $\omega = {}^{m}\sqrt{({}^{gH}/_{2L_x})}$. From this, we can find a similar equation for u,

$$u = \frac{\pi L_x^2 A\sqrt{g}}{\sqrt{H}}, \tag{10.2}$$

and note that $v = 0$. To analyse the variation in our model compared to the exact solution, we must define the *energy norm*, given by

$$\|\boldsymbol{\vartheta}\|_{\mathcal{H}} = \left(\int_{\Omega}(\nabla\boldsymbol{\vartheta})^2 d\Omega\right)^{\frac{1}{2}}, \tag{10.3}$$

for all $\boldsymbol{\vartheta} \in \mathcal{H}$ (Cangiani and Georgoulis, 2012). Using this, as in Cangiani and Georgoulis (2012) (avoiding the rather lengthy proof of Galerkin Orthogonality for brevity), we can obtain the equation for the *optimality of the finite element solution*:

$$\|\psi - \psi_h\|_{\mathcal{H}} = \min_{w_h \in \mathcal{H}_h} \|\psi - w_h\|. \tag{10.4}$$

This states that the finite element solution ψ_h is a closer approximation to the exact (weak) solution ψ, than any other function $\in \mathcal{H}_h$.

We now consider the norm for our solution, and rather than using the energy norm, which can be awkward to calculate, we consider instead the $\mathcal{L}_\infty$

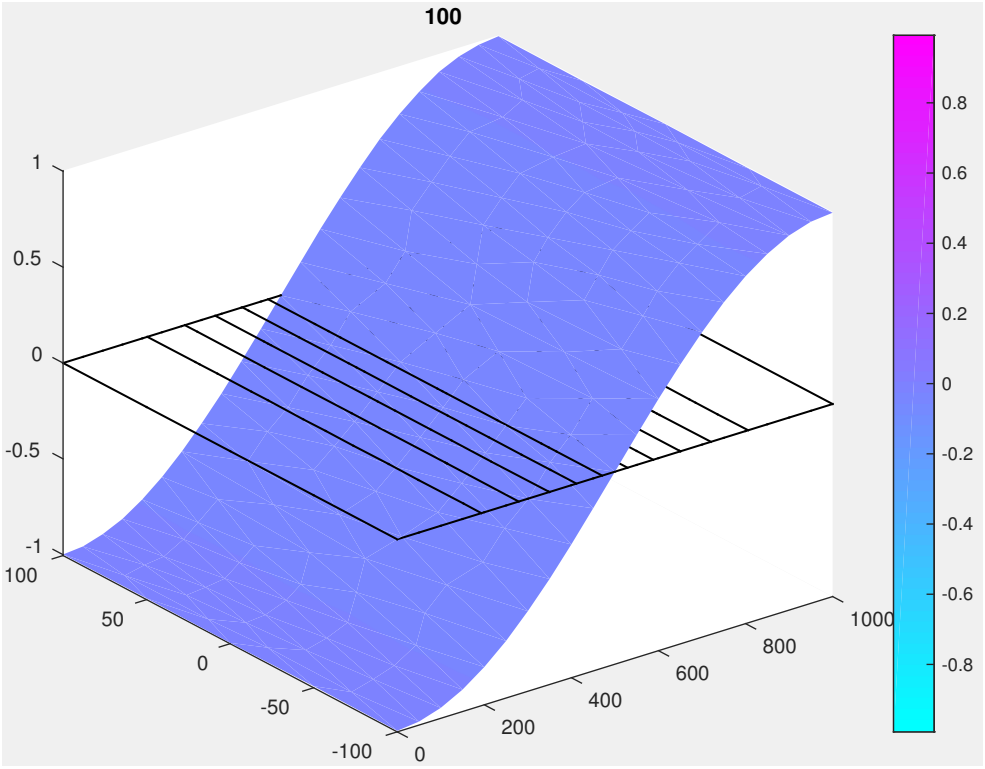

Figure 10.1: A standing wave solution for the two–dimensional shallow water equations in a closed basin.

norm, given by

$$\|f(x,y)\|_\infty = \max_{ij} f_{ij}, \tag{10.5}$$

i.e. the maximum value across all nodes. Plotting this norm against time, we can analyse the way the error changes as we change the level of mesh refinement (figure 10.2).

10.2 The Coupled Model

We now implement the two finite element models, coupled with the boundary conditions discussed in chapter 7. As we have used boundary conditions of flow only from the two dimensional to the one dimensional model, and as the driven flow at the estuary mouth takes a period of time to reach the coupling point, we at first run only the two–dimensional model. To "engage" the one–dimensional model, we put a check in the model program for changes in η and ϕ at the coupling point, and run the full system once one of the variables changes by more than this tolerance, or after a certain number of

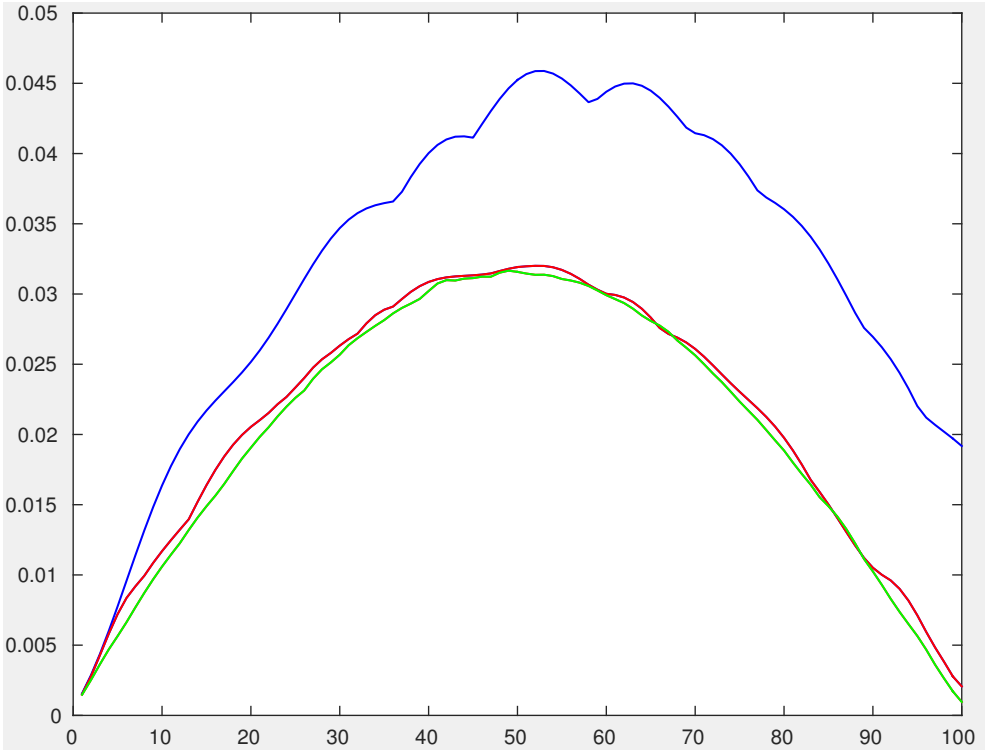

Figure 10.2: Plots of the L_∞ norms for the two–dimensional model against time for 0 mesh refinements (blue), 1 refinement (red) and two refinements(green).

iterations have already been performed. This allows us to run one–dimensional models with more elements without having quite so adverse an effect on the computation time. We also introduce our scaled variables (see section 2.1) into the model at this point.

The model domain is now scaled with the length scale L =15 km, and consists of the two dimensional basin shown in figures 10.3 and 10.4, coupled to a one–dimensional river at $^2/_{15}$ the depth (to simulate the shallower Trent) at the western point. We also take equal time steps τ, and take the spatial step in the Green Naghdi model, h, to be equal to the mean radius of the circles that circumscribe the triangular elements in the basin model, to attempt to ensure scaling is preserved across the model.

Running the model with appropriate values for the Humber estuary, we can obtain a form of bore in the nonlinear section, as shown in figures 10.5 to 10.8.

"Undoing" the scaling from earlier, we obtain a bore height of around 1.5 m, which is within reasonable bounds of height for a bore.

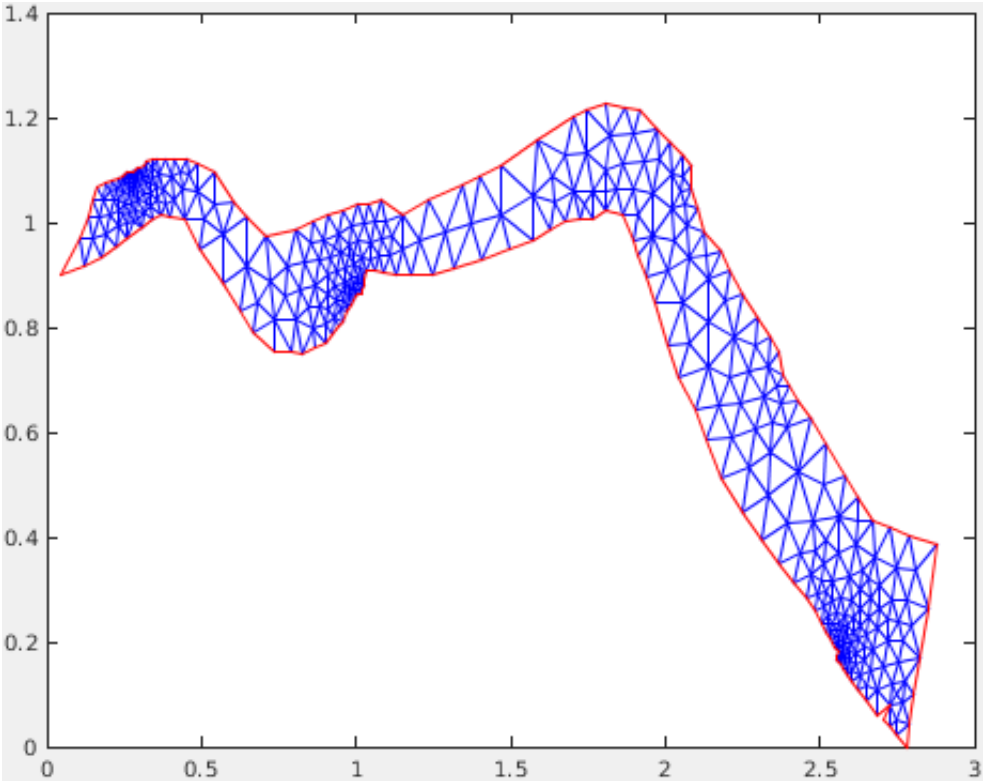

Figure 10.3: Model domain and grid for the two–dimensional finite element model, with 0 refinements.

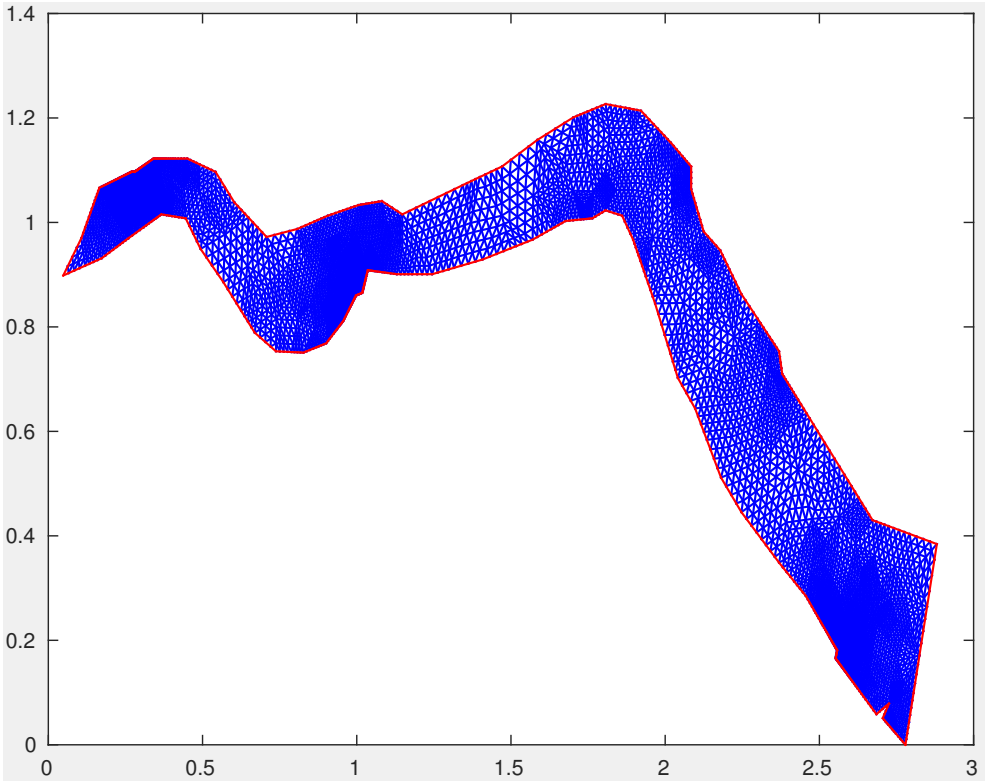

Figure 10.4: Model domain and grid for the two–dimensional finite element model, with 2 refinements.

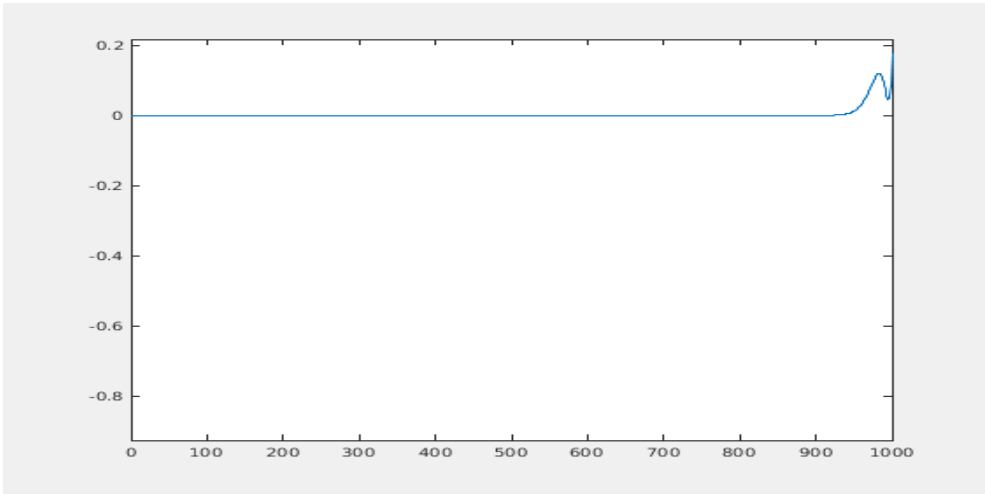

Figure 10.5: One–dimensional model output showing a bore translating up-river (from right to left).

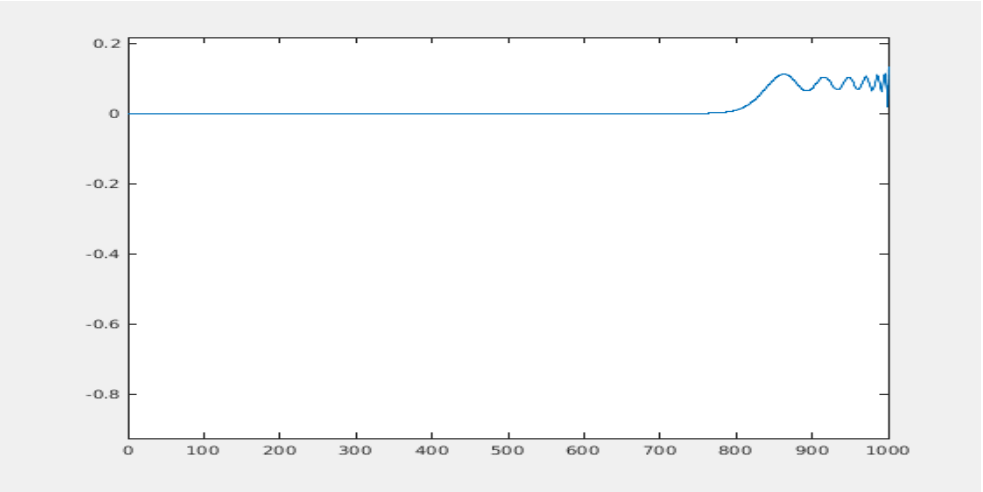

Figure 10.6: Model output after a further ten iterations.

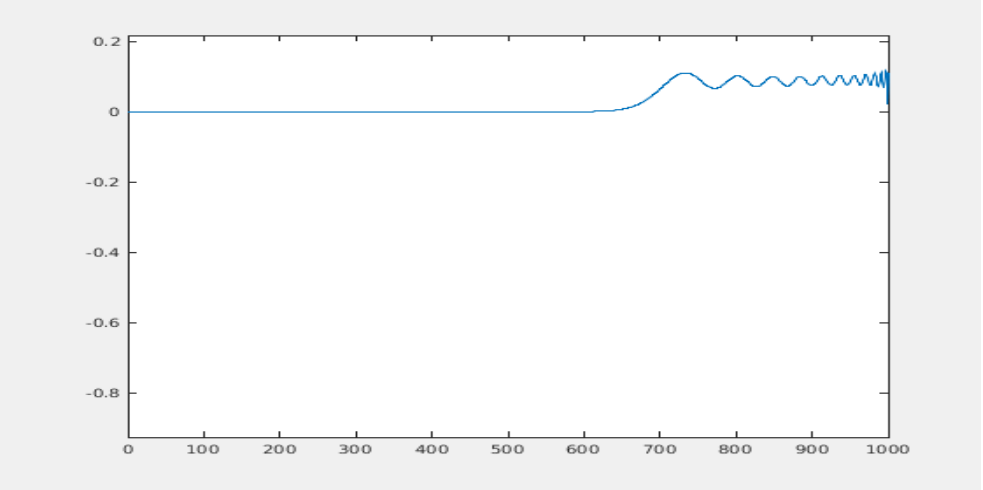

Figure 10.7: Model output after ten further iterations.

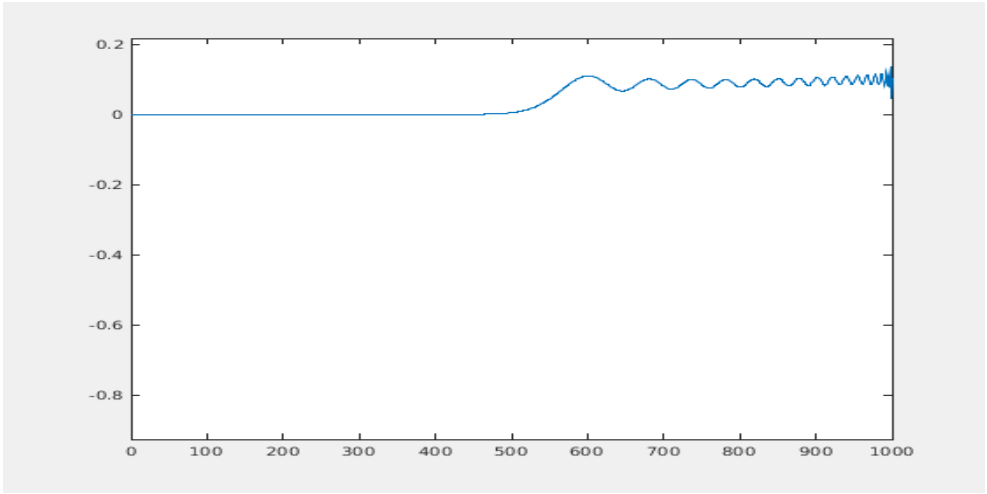

Figure 10.8: Model output after ten more iterations. The translation of the bore can be clearly seen.

Chapter 11

Conclusions and Further Work

In conclusion, we have presented a coupled shallow-water-Green-Naghdi model for the purposes of modelling the Trent Aegir, and although a series of rather crude assumptions have been made for purposes of simplification, we have obtained a model which is (at least in disjoint parts) conservative of energy, and appears to show a bore formation with scaled real-world parameters. The scaling should therefore be relatively simple to adapt to a different situation, such as a tabletop model. The process of the finite element method has been explored in detail, including the parameters and techniques for mesh generation. The MATLAB code used for the model is presented in appendix B, for use or improvement in further work.

Further work and improvements could be made in a variety of areas. The primary sources of inaccuracy in the model will likely stem from the internal boundary conditions, and the simplifications to the Green-Naghdi model, regarding the loss of higher order terms. In the case of the former, the problem could be approached by using a flux term across the boundary, as in Ambati et al. (2009), or with a good grasp of the calculus of variations the variational method presented in Kristina et al. (2014) would also significantly improve the results. Regarding the simplified Green-Naghdi/Variational Boussinesq model, a discontinous Galerkin method could be used with the Ritz method for the higher powers of h, as in Gagarina et al. (2013), or one could follow Bonneton et al. (2010), and implement a finite volume method, although this has the potential to be complex. The use of a symplectic method such as the Stormer-Verlet method could also improve the time derivatives.

The study of hydraulic jumps and bores themselves also lends itself to further study, mostly in the areas of turbulence and the dissipation of energy from the system, as worked on by Docherty and Chanson (2012),Wang et al. (2014) or, Chanson et al. (2012). Finally, we have assumed a flat bottom profile for all of the models in this report, however the rise in bottom profile can be important for the generation of bores, so smooth bottom profiles, or potentially frictional effects could also be worked in to an upgraded model.

Appendix A

Discrete Energy Calculation

To obtain the discrete energy formulation, we work from the discretised form of the momentum and continuity equations, (8.31), (8.32), (8.33), assuming zero Coriolis, and write them in terms of the mass and stress matrices, using Einstein summation notation.

$$M_{ij}\frac{d\boldsymbol{u}_j}{dt} + gS_{ij}h_j = 0, \tag{A.1}$$

$$-M_{ij}\frac{dh_j}{dt} + HS_{ji}\cdot\boldsymbol{u}_j = 0. \tag{A.2}$$

Following Nurijanyan (2013), we multiply (A.1) by $\boldsymbol{u}_i$ and (A.2) by h_i, continuing to use the Einstein notation (representing a sum over all nodes i, j). From this we obtain

$$M_{ij}\frac{d\boldsymbol{u}_j}{dt}\cdot\boldsymbol{u}_i + gS_{ij}h_j\cdot\boldsymbol{u}_i = 0, \tag{A.3}$$

$$-\frac{M_{ij}}{H}\frac{dh_j}{dt}h_i + S_{ji}\cdot h_i\boldsymbol{u}_j = 0. \tag{A.4}$$

To proceed from here, we note that $S_{ij}\cdot\boldsymbol{u}_i h_j = S_{ji}\cdot\boldsymbol{u}_j h_i$, so we can now write

$$HM_{ij}\frac{d\boldsymbol{u}_j}{dt}\cdot\boldsymbol{u}_i + gM_{ij}\frac{dh_j}{dt}h_i = 0. \tag{A.5}$$

We can also note that, as we sum over all nodes, ${}^{d\alpha_i}/_{dt}\alpha_j = {}^{d\alpha_j}/_{dt}\alpha_i$, and thus

$$\frac{d}{dt}(\boldsymbol{u}_i\cdot\boldsymbol{u}_j) = \frac{d\boldsymbol{u}_i}{dt}\cdot\boldsymbol{u}_j + \frac{d\boldsymbol{u}_j}{dt}\cdot\boldsymbol{u}_i = 2\frac{d\boldsymbol{u}_j}{dt}\boldsymbol{u}_i, \tag{A.6}$$

$$\frac{d}{dt}h_ih_j = \frac{dh_i}{dt}h_j + \frac{dh_j}{dt}h_i \qquad = 2\frac{dh_j}{dt}h_i. \tag{A.7}$$

Using this, (A.5) can now be written

$$\frac{d}{dt}\left(HM_{ij}\boldsymbol{u}_i\cdot\boldsymbol{u}_j + gM_{ij}h_ih_j\right) = 0, \tag{A.8}$$

which is equivalent to (8.43).

Appendix B

Model Code

B.1 Main Model Code

```
clear all
%read in coordinates of Humber trace, for boundary.
est=dlmread('estuary.dat');
%remove duplicate points
est1=unique(est,'rows','stable');
%scale the points to be relative to the length scale
dis=sqrt(152^2+(1132-1092)^2);
mult=6000/(15000*dis);
est2=mult*est1;
xbpts=est2(:,1);
%flip the points in y, as the datafile is upside-down
ybpts=502*mult-est2(:,2);
%read the points into a CSG polygon with as many points as
    sides
poly1=[2;length(xbpts);xbpts;ybpts];
%undo the CSG so MATLAB can create the mesh
d1=decsg(poly1);
ref=input('number of refinements?');
%initiate mesh, initmesh performs Delaunay triangulation
[pSW,e,t]=initmesh(d1);
%refine mesh if required, by quartering each triangular
   element
if ref~=0
r=1;
while r<=ref
[pSW,e,t]=refinemesh(d1,pSW,e,t);
r=r+1;
end
end
f=-input('coriolis parameter=?');
%get a matrix saying which points are on the boundary.
```

```
30 [in,on]=inpolygon(pSW(1,:),pSW(2,:),xbpts,ybpts);
31 %find the point at the west end, for coupling to the Green
      -Naghdi model
32 for i=1:length(pSW)
33          if abs(pSW(1,i)-min(pSW(1,:)))<0.01;
34              leftend=i;
35          end
36
37 end
38 %find the other two nodes connected to the west end
39 pSW2=pSW;
40 for i=1:size(t,2)
41 for j=1:3
42 if t(i,j)==leftend
43 leftbutone=[t(i,mod(j+1,3)),t(i,mod(j+2,3))];
44 break
45 end
46 end
47 %get the radius of circumscribing circles for each element
48 for k=1:size(t,2)
49 circ(k)=circtri(pSW(1,t(1,k)),pSW(2,t(1,k)),pSW(1,t(2,k)),
      pSW(2,t(2,k)), ...
50                  pSW(1,t(3,k)),pSW(2,t(3,k)));
51 end
52 %take the Green-Naghdi space step as the mean of these
      radii
53 h=(mean(circ));
54 %build points and elements for Green-Naghdi model
55 xmin=0;
56 nelemGN=input(['number of GN elements? h=',num2str(h),'.
      nelemGN=']);
57 xmax=xmin+nelemGN*2*h
58 pGN=xmin:h:xmax;
59 % %%% h=input('h=');
60 tau=input('\tau=');
61 Nt=input('max time (for coupled model)?');
62 uriver=input('river velocity=')/1.5;
63 nelemGN2=(xmax-xmin)/(2*h);
64 NnGN=length(pGN);
65 el=zeros(3,nelemGN);
66 el(:,1)=[1,2,3]';
67     for j=2:nelemGN
68     el(:,j)=el(:,j-1)+2;
69     end
70 el2=[1:nelemGN;2:nelemGN+1];
71 %Initialise matrices and vectors for Green-Naghdi model
72 MGN=zeros(NnGN);
73 AGN=zeros(NnGN);
74 ZGN=zeros(NnGN);
```

```
IGN=eye(NnGN);
etaGN=zeros(NnGN,1);
phiGN=zeros(NnGN,1);
lambdaGN=zeros(NnGN,1);
qGN=zeros(NnGN,1);
PSIGN=[phiGN;etaGN;qGN;lambdaGN];
%create storage matrix
psistoreGN=zeros(length(PSIGN),1+Nt/tau);
psistoreGN(:,1)=PSIGN;
%scale gravity, depths and tidal parameters.
g=9.81/15000;
H=input('H(depth)=')/15;
HGN=H/7.5;
amp=input('tidal amplitude=')/15;
omega=input('tidal frequency=')/10000;
%read in parameters for switching on nonlinear model.
maxiter=input('max iterations before second model engaged=
    ');
phitol=input('change in \phi to engage second model');
etatol=input('change in \eta to engage second model')
%Initialise matrices for shallow water model;
NnSW=max(max(t));
NNodesSW=sum(in)-sum(on);
AxSW=zeros(NnSW);
AySW=zeros(NnSW);
MSW=zeros(NnSW);
USW=0*ones(NnSW,1);
VSW=USW;
etaSW=USW;
PSISW=[USW;VSW;etaSW];
  psistoreSW=zeros(length(PSISW),1+Nt/tau);
psistoreSW(:,1)=PSISW;
%get index of nodes on the tidal boundary
for i=1:size(e,2)
    if e(5,i)==3;
        rightedge(i)=1;
    else
        rightedge(i)=0;
    end
end
rind=e(1:2,rightedge==1);
rightb=zeros(length(PSISW),1);
for i=1:length(PSISW)/3
  if any(any(abs(i-rind)<0.0001))==1;
        rightb(i+(2/3)*length(PSISW))=1;
  end
  end
%build mass and stress matrices for both models
for k=1:size(t,2)
```

```
123 for alpha=1:3
124 i=t(alpha,k);
125 for beta=1:3
126 j=t(beta,k);
127 MSW(i,j)=MSW(i,j)+mhat(alpha,beta,pSW(1,t(1,k)),pSW(2,t(1,
        k)),pSW(1,t(2,k)),pSW(2,t(2,k)),pSW(1,t(3,k)),pSW(2,t
        (3,k)));
128 AxSW(i,j)=AxSW(i,j)+ahatx2(alpha,beta,pSW(1,t(1,k)),pSW(2,
        t(1,k)),pSW(1,t(2,k)),pSW(2,t(2,k)),pSW(1,t(3,k)),pSW
        (2,t(3,k)));
129 AySW(i,j)=AySW(i,j)+ahaty2(alpha,beta,pSW(1,t(1,k)),pSW(2,
        t(1,k)),pSW(1,t(2,k)),pSW(2,t(2,k)),pSW(1,t(3,k)),pSW
        (2,t(3,k)));
130 end
131 end
132
133 end
134 for k=1:nelemGN
135     for alpha=1:3
136 i=el(alpha,k);
137 for beta=1:3
138 j=el(beta,k);
139         MGN(i,j)=MGN(i,j)+m1hat(alpha,beta,h);
140             AGN(i,j)=AGN(i,j)+a1hat(alpha,beta,h);
141 end
142 end
143 end
144
145 %create block matrices
146 MGNL=[-(2/tau)*MGN,g*MGN,ZGN,ZGN;ZGN,MGN,ZGN,ZGN;ZGN,ZGN,
        MGN,ZGN;ZGN,ZGN,(1/3)*(HGN^3)*MGN,MGN];
147 MGNR=[(2/tau)*MGN,g*MGN,ZGN,ZGN;ZGN,-MGN,ZGN,ZGN;ZGN,ZGN,
        MGN,ZGN;ZGN,ZGN,(1/3)*(HGN^3)*MGN,MGN];
148 ASGN=[ZGN,ZGN,ZGN,ZGN;HGN*AGN,ZGN,ZGN,AGN;AGN,ZGN,ZGN,ZGN;
        ZGN,ZGN, ...
149        ZGN,ZGN];
150 MASW=blkdiag(MSW,MSW,-MSW);
151 ZSW=zeros(length(USW));
152 ISW=eye(length(USW));
153 FASW=[ZSW,-f*MSW,ZSW;f*MSW,ZSW,ZSW;ZSW,ZSW,ZSW];
154 SASW=[ZSW,ZSW,9.81*AxSW;ZSW,ZSW,9.81*AySW;H*AxSW',H*AySW',
        ZSW];
155 %run 2D model until there is movement at west edge or
        limit reached
156 mod1=PSISW(leftend);
157 mod2=PSISW(leftend+length(PSISW)/3);
158 iter=1;
159 while iter<maxiter
160     if abs(mod1)>phitol
```

```
161         break
162     elseif abs(mod2)>etatol
163         break
164     end
165 clear AAASW AAAGN
166 %form RHS vector and concacenated matrix
167 GSW=MASW*PSISW-(tau/2)*(SASW*PSISW+FASW*PSISW);
168     AAASW=MASW + (tau/2)*(SASW+FASW);
169 %get tidal height
170     tidalclamp=amp*sin(omega*iter*tau);
171     TBC=zeros(length(PSISW),1);
172 %Apply tidal boundary condition to right boundary using
    RHS of (8.22)
173 for k=1:size(t,2)
174     for alpha=1:3
175         i=t(alpha,k);
176         TBC(i+(2/3)*length(PSISW))=TBC(i+(2/3)*length(
              PSISW))
177 +Ghat(alpha,tidalclamp,pSW(1,t(1,k)),pSW(2,t(1,k)),pSW(1,t
    (2,k)),pSW(2,t(2,k)),pSW(1,t(3,k)),pSW(2,t(3,k)));
178     end
179 end
180 for i=1:length(TBC)
181 TBCS(i)=TBC(i)*rightb(i);
182 end
183 GSW=GSW+TBCS';
184 %Solve for shallow water PSI forward in time
185 PSISWn=AAASW\GSW;
186 %Store value for later
187 psistoreSW(:,iter)=PSISWn;
188 PSISW=PSISWn;
189 mod1=PSISW(leftend);
190 mod2=PSISW(leftend+length(PSISW)/3);
191 iter=iter+1
192 end
193 %now run both models
194 for time=2:1+Nt/tau
195 clear AAASW AAAGN GSW GGN PSIGNn PSISWn NBC
196 %form matrices and RHS vector, and tidal boundary
    condition
197 GSW=MASW*PSISW-(tau/2)*(SASW*PSISW+FASW*PSISW);
198     AAASW=MASW + (tau/2)*(SASW+FASW);
199     GGN=(tau/2)*(MGNR+ASGN)*PSIGN;
200 AAAGN=-(tau/2)*(MGNL+ASGN);
201 tidalclamp=amp*sin(omega*(time+iter)*tau);
202 TBC=zeros(length(PSISW),1);
203 for k=1:size(t,2)
204     for alpha=1:3
205         i=t(alpha,k);
```

```
206         TBC(i+(2/3)*length(PSISW))=TBC(i+(2/3)*length(
                PSISW))+Ghat(alpha,tidalclamp,pSW(1,t(1,k)),pSW
                (2,t(1,k)),pSW(1,t(2,k)),pSW(2,t(2,k)),pSW(1,t
                (3,k)),pSW(2,t(3,k)));
207     end
208 end
209 for i=1:length(TBC)
210 TBCS(i)=TBC(i)*rightb(i);
211 end
212 GSW=GSW+TBCS';
213 %apply boundary conditions to the Green Naghdi model -
       coupling and river
214 NBC=zeros(length(PSIGN),1);
215 for k=1:nelemGN
216     for alpha=1:3
217 i=el(alpha,k);
218  NBC(i)=NBC(i)+g1hat(alpha,PSISW(leftend),h);
219 NBC(i)=NBC(i)+g1hat(alpha,uriver,h);
220  end
221 end
222 for i=1:length(NBC)
223     if i==length(NBC)/4
224         NBCS(i)=NBC(i);
225  elseif i==1
226         NBCS(i)=NBC(i);
227     else
228         NBCS(i)=0;
229     end
230 end
231
232 GGN=GGN+NBCS';
233 %apply boundary condition on eta
234 etaend=PSISW(leftend+2*length(PSISW)/3);
235 for j=1+length(PSIGN)/4:length(PSIGN)/2
236     GGN(j)=GGN(j)-AAAGN(NnGN,j)*etaend;
237 end
238
239 GGN(length(PSIGN)/4+NnGN)=etaend;
240 AAAGN(:,length(PSIGN)/4+NnGN)=0;
241 AAAGN(length(PSIGN)/4+NnGN,:)=0;
242 AAAGN(length(PSIGN)/4+NnGN,length(PSIGN)/4+NnGN)=1;
243 %Solve and store solutions
244 PSIGNn=AAAGN\GGN;
245 psistoreGN(:,time+iter)=PSIGNn;
246 PSIGN=PSIGNn;
247 PSISWn=AAASW\GSW;
248 psistoreSW(:,time+iter)=PSISWn;
249 PSISW=PSISWn;
250 time
```

```
251 end
252 %split block storage matrices back into each variable
253 ustoreSW=psistoreSW(1:length(PSISW)/3,:);
254 vstoreSW=psistoreSW(1+length(PSISW)/3:2*length(PSISW)/3,:)
    ;
255 etastoreSW=psistoreSW(1+2*length(PSISW)/3:length(PSISW),:)
    ;
256 phistoreGN=psistoreGN(1:length(PSIGN)/4,:);
257 etastoreGN=psistoreGN(1+length(PSIGN)/4:length(PSIGN)/2,:)
    ;
258 qstoreGN=psistoreGN(1+length(PSIGN)/2:3*length(PSIGN)/4,:)
    ;
259 lambdastoreGN=psistoreGN(1+3*length(PSIGN)/4:length(PSIGN)
    ,:);
```

B.2 Circumscribing the Elements

```
1   function r=circtri(x1,y1,x2,y2,x3,y3)
2 %this program finds the smallest circle containing points
3 %(x1,y1),(x2,y2),(x3,y3), in this case the triangle
    vertices
4 xa = x2-x1;
5 ya = y2-y1;
6 xb = x3-x1;
7 yb = y3-y1;
8 da = xa^2+ya^2;
9 db = xb^2+yb^2;
10 A = 2*(xa*yb-ya*xb);
11 r = sqrt(da*db*((x3-x2)^2+(y3-y2)^2))/abs(A);
12 end
```

B.3 Matrix Generating Programs

```
1  function out=mhat(alpha,beta,x0,y0,x1,y1,x2,y2)
2 %create shallow water mass matrix elements
3 %coefficients of canonical element basis and derivatives
4 dc=[-1,1,0;-1,0,1];
5 c=[1,-1,-1;0,1,0;0,0,1];
6 %jacobian of transform to canonical element
7 detJ=((y2-y0)*(x1-x0))-((y1-y0)*(x2-x0));
8 %define subfunction as the triangular basis
9 function hout=hf(p,q)
10 hout=(c(alpha,1)+c(alpha,2)*p+c(alpha,3)*q)*(c(beta,1)+c(
    beta,2)*p+c(beta,3)*q)*abs(detJ);
11 end
12 %then integrate using Gauss quadrature
13 out=(1/6)*(hf(0.5,0)+hf(0,0.5)+hf(0.5,0.5));
14 end
```

```
function out=Ahatx(alpha,beta,x0,y0,x1,y1,x2,y2)
%create x derivative part of stress matrix for shallow
    water
d=[-1,1,0;-1,0,1];
c=[1,-1,-1;0,1,0;0,0,1];
detJ=((y2-y0)*(x1-x0))-((y1-y0)*(x2-x0));
function hout=hf(p,q)
hout=(c(alpha,1)+c(alpha,2)*p+c(alpha,3)*q)*((y2-y0)*d(1,
    beta)+(y0-y1)*d(2,beta))*(1/detJ)*abs(detJ);
end
out=(1/6)*(hf(0.5,0)+hf(0,0.5)+hf(0.5,0.5));
end
```

```
  function out=Ahaty(alpha,beta,x0,y0,x1,y1,x2,y2)
%create y derivative part of stress matrix for shallow
    water
d=[-1,1,0;-1,0,1];
c=[1,-1,-1;0,1,0;0,0,1];
detJ=((y2-y0)*(x1-x0))-((y1-y0)*(x2-x0));
function hout=hf(p,q)
hout=(c(alpha,1)+c(alpha,2)*p+c(alpha,3)*q)*((x0-x2)*d(1,
    beta)+(x1-x0)*d(2,beta))*(1/detJ)*abs(detJ);
end
out=(1/6)*(hf(0.5,0)+hf(0,0.5)+hf(0.5,0.5));
end
```

```
function out=m1hat(alpha,beta,h)
%create Green-Naghdi mass matrix elements
c=[1/2,-1/2,0;-1,0,1;1/2,1/2,0];

function hout=hf(p)
%2nd order lagrange polynomials
hout=(c(alpha,1)*p^2+c(alpha,2)*p+c(alpha,3))*(c(beta,1)*p
    ^2+c(beta,2)*p+c(beta,3));

   hout=(h/2)*hout;
end
%higher order Gauss quadrature
out=(5/9)*hf(-sqrt(3/5))+(8/9)*hf(0)+(5/9)*hf(sqrt(3/5));
end
```

```
function out=a1hat(alpha,beta,h)
%Form Green-Naghdi stress matrix elements
c=[1,-1/2;-2,0;1,1/2];

function hout=hf(p)
hout=(c(alpha,1)*p+c(alpha,2))*(c(beta,1)*p+c(beta,2));

   hout=(h/2)*hout;
end
```

```
10 out=(5/9)*hf(-sqrt(3/5))+(8/9)*hf(0)+(5/9)*hf(sqrt(3/5));
11 end
```

Bibliography

Adams, R. and Fournier, J. (2003). *Sobolev Spaces*, Oxford: Elsevier.

Ambati, V., Klaver, F., van der Vegt, J. and Bokhove, O. (2009). Coupling Numerical Models for Shallow and Deep Water Waves, Slides from Presentation - Castro-Urdiales, Cantabria, Spain. Available online at `http://www.math.sciences.univ-nantes.fr/~berthon/conferences/castro_2009/talks/contributed/ambati.pdf` [accessed 23/8/15].

Bokhove, O. (2014). Numerical Methods for Fluid Dynamics, Lecture Notes.

Bokhove, O. and Griffiths, S. (2015). MATH5458M geophysical fluid dynamics, lecture notes, [online]. [Accessed 01/02/15] Available from `http://vlebb.leeds.ac.uk`.

Bonneton, P., Barthelemy, E., Carter, J., Chazel, F., Cienfuegos, R., Lannes, D., Marche, F. and Tissier, M. (2010). Fully Nonlinear Weakly Dispersive Modelling of Wave Transformation, Breaking and Runup, *arXiv preprint* **1004.3480**.

Cangiani, A. and Georgoulis, M. (2012). MA4011 computational methods for partial differential equations, lecture notes, [online]. [Accessed 25/5/14] Available from `http://blackboard.le.ac.uk`.

Carter, E. (No Date). The Aegir on the Trent, Photograph [online] [accessed 27/8/15] Available at `http://www.snicholson.freeserve.co.uk/gainsborough/gainstour6.html`.

Carter, G. S. and Merrifield, M. A. (2007). Open Boundary Conditions for Regional Tidal Simulations, *Ocean Modelling* **18**: 194–209.

Chanson, H. (2012). *Tidal Bores, Aegir, Eagre, Mascaret, Pororoca: Theory and Observations*, Singapore: World Scientific.

Chanson, H. and Koch, C. (2006). *Unsteady Turbulence Characteristics in a Tidal Bore*, London: Taylor & Francis, chapter A1: Turbulent Open Channel Flow and Transport Phenomena, pp. 79–88.

Chanson, H., Lubin, P. and Glockner, S. (2012). Unsteady Turbulence in a Shock: Physical and Numerical Modelling in Tidal Bores and Hydraulic Jumps, *in* R. J. Marcuso (ed.), *Turbulence: Theory, Types and Simulation*, New York: Nova Science, chapter 3.

Computational Science Education Project (1995). *Direct and Inverse Bioelectric Field Problems*, Ebook: Available at `http://www.phy.ornl.gov/csep/bf/bf.html`.

Coxeter, H. (1947). *Regular Polytopes*, London: Methuen & Co.

Dawson, C. and Mirabito, C. M. (2008). The Shallow Water Equations, [online]. [Accessed 15/06/15] Available from `http://users.ices.utexas.edu/~arbogast/cam397/dawson_v2.pdf`.

Delauney, B. (1934). Sur la Sphère vide. A la Mémoire de Georges Voronoï, *Bulletin de l'Académie de Sciences de l'URSS. Classe des Sciences Mathématiques et Naturelles* (6): 793–800.

Dirichlet, G. (1850). Über die Reduction der Positiven Quadratischen Formen mit Drei Unbestimmten Ganzen Zahlen. [On the Reduction of Positive Quadratic Form with Three Indeterminate Integers. Trans. K.N. Tiyapan], *Journal für die Reine und Angewandte Mathematik* **40**: 209–227. Originally a Lecture to the Academy, 13th July 1848. [Translation in *Voronoi Translated 2, Lulu.com, 2001, pp. 7-25*].

Docherty, N. J. and Chanson, H. (2012). Physical Modeling of Unsteady Turbulence in Breaking Tidal Bores, *Journal of Hydraulic Engineering* **138**(4): 412–419.

Edwards, A., Sayers, D. and Woodward, G. (1988). The Water Quality Management of the Humber Estuary Committee, *in* P. Newman and A. Agg (eds), *Environmental Protection of the North Sea*, Oxford: Heinemann Professional.

Environment Agency (2015). River and Sea Levels, Website. Available at `http://apps.environment-agency.gov.uk/river-and-sea-levels/default.aspx`. [Accessed 26/8/15].

Evans, L. C. (1998). *Partial Differential Equations*, Providence, RI: American Mathematical Society.

Flaherty, J. E. (2005). MATH6860 finite element analysis, lecture notes - rensselaer polytechnic institute, [online]. [Accessed 01/08/15] Available from `http://www.cs.rpi.edu/~flaherje/`.

Flather, R. (1976). A Tidal Model of the North West Europeak Continental Shelf, *Memoires de la Societe Royale de Sciences de Liege* **6**(10): 141–164.

Fort, T. (2008). River Trent: Going with the Flow of History, *The Daily Telegraph* . [online] [accessed 27/8/15] Available at `http://www.telegraph.co.uk/news/earth/3336220/River-Trent-Going-with-the-flow-of-history.html`.

Gagarina, E., van der Vegt, J. and Bokhove, O. (2013). Horizontal Circulation and Jumps in Hamiltonian Wave Models, *Nonlinear Processes in Geophysics* **20**: 483–500.

George, P.-L. and Borouchaki, H. (1998). *Delaunay Triangulation and Meshing: Application to Finite Elements*, Paris: Hermès.

Green, A. and Naghdi, P. (1976). A Derivation of Equations for Wave Propagation in Water of Variable Depth, *Journal of Fluid Mechanics* **78**(2): 237–246.

Kalogirou, A. and Bokhove, O. (2015). Variational Water Wave Modelling: From Continuum to Experiment, *in* T. J. Bridges, M. D. Groves and D. P. Nicholls (eds), *Lectures on the Theory of Water Waves*, London Mathematical Society Lecture Notes, Cambridge: University Press.

Kelmanson, M. (2014). MATH3474 numerical methods, lecture notes, [online]. [Accessed 25/01/15] Available from `http://vlebb.leeds.ac.uk`.

Kristina, W., Bokhove, O. and van Groesen, E. (2014). Effective Coastal Boundary Conditions for Tsunami Wave Run-Up over Sloping Bathymetry, *Nonlinear Processes in Geophysics* **21**: 987–1005.

Miles, J. and Salmon, R. (1985). Weakly Dispersive Nonlinear Gravity Waves, *Journal of Fluid Mechanics* **157**: 519–531.

Nurijanyan, S. (2013). *Discrete and Continuous Hamiltonian Systems for Wave Modelling*, PhD thesis, University of Twente.

Owen, A. (2015). Variational and Hamiltonian Geophysical Fluid Dynamics, Term Paper - MATH5458 Geophysical Fluid Dynamics.

Paterson, A. (1983). *A First Course in Fluid Dynamics*, Cambridge: University Press.

Proudman, J. and Doodson, A. (1924). The Principal Constituent of the Tides of the North Sea, *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character* **224**: 185–219.

Savenije, H. (2006). *Salinity and Tides in Alluvial Estuaries*, Amsterdam: Elsevier.

Smith, D. (2015). River Severn Bore: Watch Surfers Ride Spectacular 'Super Tide' on Britain's Longest River, *The Daily Mirror* . [online] [accessed 25/8/15] Available at `http://www.mirror.co.uk/news/uk-news/river-severn-bore-watch-surfers-5374225`.

The Trent Aegir at Gainsborough, Lincolnshire (2005). Image, Online, available at `https://en.wikipedia.org/wiki/Trent_Aegir#/media/File:Trent_Aegir_3.JPG`. In Wikimedia Commons, uploaded by Chris075. [accessed 27/8/15].

Townend, I., Wang, Z. and Rees, J. (2007). Millennial to Annual Volume Changes in the Humber Estuary, *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* **463**(2079): 837–854.

Tricker, R. (1964). *Bores, Breakers, Waves and Wakes: An Introduction to the Study of Waves on Water*, London: Mills & Boon.

UK Hydrographic Office (1993). Admiralty Chart No. 109 (River Humber, River Ouse and Trent), Admiralty Chart.

University of Western Australia, School of Computer Science and Software Engineering (2010). Voronoï diagrams and delaunay triangulations, Image, Online. Available at `http://undergraduate.csse.uwa.edu.au/units/CITS3210/project.html` [accessed 22/7/15].

van Rijn, L. (2011). Comparison Hydrodynamics and Salinity of Tide Estuaries; Elbe, Humber, Scheldt and Weser, *Technical report*, Deltares.

Voronoi, G. (1908). Nouvelles Applications des Paramètres Continus à la Théorie des Formes Quadratiques. Deuxième Mémoire. Recherches sur les Parallélloèdres Primitifs., *Journal für die Reine und Angewandte Mathematik* **134**: 198–287.

Wang, H., Felder, S. and Chanson, H. (2014). An Experimental Study of Turbulent Two-Phase Flow in Hydraulic Jumps and Application of a Triple Decomposition Technique, *Experiments in Fluids* **55**(7): 1–18.