FUNWAVE-TVD

Fully Nonlinear Boussinesq Wave Model with TVD Solver
Documentation and User's Manual
(Version 3.0)
Release Dec 2016

Fengyan Shi, Center for Applied Coastal Research, University of Delaware
James T. Kirby, Center for Applied Coastal Research, University of Delaware
Babak Tehranirad, Center for Applied Coastal Research, University of Delaware
Jeffrey C. Harris, Department of Ocean Engineering, University of Rhode Island,

Guest authors
Young-Kwang Choi, Korea Institute of Ocean Science and Technology
Matt Malej, U.S. Army Engineer Research and Development Center

CENTER FOR APPLIED COASTAL RESEARCH
Ocean Engineering Laboratory
University of Delaware
Newark, Delaware 19716

Acknowledgements

This model development was a part of the surfzone optics project sponsored by the Office of Naval Research, Coastal Geosciences Program through grant N00014-10-1-0088. The work of wave runup benchmark tests was sponsored by the National Tsunami Hazard Mitigation Program(NTHMP).

Special thanks to Drs. Seung-Nam Seo and Young-Kwang Choi from Korea Institute of Ocean Science and Technology (KIOST) who conducted line-by-line checking on the code and provided some important revisions on Version 3.0. Some examples were contributed by Drs. Seo and Choi.

Abstract

The report documents a new version of the fully nonlinear Boussinesq wave model (FUNWAVE) initially developed by Kirby et al. (1998). The development of the present version was motivated by recent needs for modeling of surfzone-scale optical properties in a Boussinesq model framework, and modeling of Tsunami wave in both a regional/coastal scale for prediction of coastal inundation and a basin scale for wave propagation. This version features several theoretical and numerical improvements, including 1) a more complete set of fully nonlinear Boussinesq equations; 2) MUSCL-TVD solver with adaptive Runge-Kutta time stepping; 3) Both viscosity wave breaking scheme and shock-capturing breaking scheme; 4) wetting-drying moving boundary condition with incorporation of HLL construction method into the scheme; 5) an option for parallel computation. The documentation provides derivations of the conservation form of theoretical equations, re-arrangement of pressure gradient term in order to obtain a numerically well-balanced form, detailed numerical schemes, users' manual and examples. Users can also refer to Shi et al. (2012a) for the Cartesian version and Kirby et al. (2012) for the spherical version of the model.

The examples in the previous report (Shi et al., 2011) have been updated using the present version (Version 3.0). This version also includes new examples contributed by Drs. Seung-Nam Seo and Young-Kwang Choi. The tsunami benchmark tests in spherical coordinates are reported separately in Tehranirad et al. (2013).

Source code update:

https://github.com/fengyanshi/FUNWAVE-TVD

Contents

List of Figures

List of Tables

Differences between Version 1.0 and Version 1.1

The major updates from FUNWAVE-TVD Version 1.0 to Version 1.1 are new model capabilities in grid nesting, incorporation of wind effect, wave generation in the spherical mode and calculation of time-averaged properties. Version 1.1 has been fully tested using the existing examples in the previous report for Version 1.0 and tsunami benchmark tests in spherical coordinates reported by Shi et al. (2012b). Detailed updates are listed below.

1. One-way grid nesting capability was added in Version 1.1 (subroutine OneWayCoupling). The nesting schemes and an example are shown in section 4 and section 5, respectively.

2. Wind effect was added based on Chen et al. (2004). Users should specify -DWIND in Makefile in order to use this option. Details are described in section 4.5.

3. Output of mean velocities.

4. Wavemakers were added in the spherical mode in order to conduct benchmark tests (Shi et al., 2012b).

5. Coding for VANLEER_LIMITER () was modified for a better computational efficiency.

6. At a wet-dry interface, water depth at cell interface (DepthX and DepthY) was evaluated using the depth at the cell center of the wet point. This approach can avoid an extreme large value of slope caused by the raw topographic data (e.g., NGDC data).

7. Bugs were fixed for MASK9 where the previously defined array has a different size from that of U and V.

Differences between Version 2.0 and Version 1.1

The major change from Version 1.1 to Version 2.0 is for the spherical mode. Recently, Kirby et al. (2012) formulated a more complete set of Boussinesq equations in spherical coordinates. The equations include both weakly nonlinear and fully nonlinear forms based on velocities at a reference level z_α. In version 2.0, we implemented the weakly nonlinear formulations. Version 2.0 has been fully tested using the existing examples in the previous report for Version 1.0/1.1 and tsunami benchmark tests in spherical coordinates (Shi et al., 2012). Other updates are listed below.

1. Compared to Version 1.1, a more accurate scheme for grid nesting was implemented. Affected subroutines include PHI_COLL, CAL_DISPERSION and GET_Eta_U_V_HU_HV. Boundary conditions for dispersion terms were removed at nesting boundaries (in PHI_COLL and CAL_DISPERSION). Boundary conditions at nesting boundaries are added in the tridiagonal solver (GET_Eta_U_V_HU_HV). To activate this option, set -DZALPHA in Makefile.

2. The wall clock was added in the main program.

3. Input format for gauges was changed to geographic coordinates (Lat/Lon) in the spherical mode (Version 1.1 and earlier used grid numbers I and J).

4. Several useless subroutines were removed from the code.

Differences between Version 2.1 and Version 2.0

The major change from Version 2.0 to Version 2.1 is for the friction factor specifications. Inside previous versions the model used a fixed friction factor for the whole modeling domain. However, in version 2.1 the model is capable of reading an initial friction field (friction coefficient matrix) that can be defined differently for each grid by the user. Also, the code is now able to use Manning coefficient as well as conventional friction factor. Moreover, output system efficiency is upgraded inside version 2.1 and several variable are added to the recording list of FUNWAVE-TVD. Updates are listed below.

1. Version 2.1 is capable of reading an input file for friction coefficient for cases in which friction field is not homogeneous.

2. Version 2.1 is able to use Manning coefficient formulation (in addition to friction coefficient formulation) to calculate friction force. To activate this option, set -DMANNING in Makefile.

3. The efficiency of writing the output files is upgraded in this version.

4. In version 2.1, user can print the maximum momentum flux, maximum velocity, and minimum surface elevation as well as maximum vorticity in addition to other output variable in prior versions.

Differences between Version 2.2 and Version 2.1

There are some problems found in application of Larsen-Dancy-type (Larsen and Dancy, 1983, Chen et al., 1999) sponge layers for long-term simulations. The direct damping method combined with the TVD scheme generates sawtooth noises with a $2dx$ wave length. The sawtooth noises are usually not noticeable due to small magnitudes. However, they grow gradually with time and may become significant in a long term simulation. In Version 2.2, we complemented two additional slope layer methods: the friction-type sponge layer and the viscous-type sponge layer.

In addition, Version 2.2 added the eddy-viscosity type breaking algorithm (Kennedy et al., 2000).

Differences between Version 3.0 and Version 2.2

In Version 3.0, the codes were reorganized into the modular form. Several bugs have been found and fixed (thanks to Young-Kwang Choi's work). We completely removed the option for the implicit iteration procedure because it is unnecessary based on tests. The periodic boundary condition was implemented in the parallel code using the Sherman-Morrison algorithm. New examples are added.

1 Introduction

Boussinesq wave models have become a useful tool for modeling surface wave transformation from deep water to the swash zone, as well as wave-induced circulation inside the surfzone. Improvements in the range of model applicability have been obtained with respect to classical restrictions to both weak dispersion and weak nonlinearity. Madsen et al (1991) and Nwogu (1993) demonstrated that the order of approximation in reproducing frequency dispersion effects could be increased using either judicious choices for the form (or reference point) for Taylor series expansions for the vertical structure of dependent variables, or operators effecting a rearrangement of dispersive terms in already-developed model equations. These approaches, combined with use either of progressively higher order truncated series expansions (Gobbi et al, 2000; Agnon et al, 1999) or multiple level representations (Lynett and Liu, 2004), have effectively eliminated the restriction of this class of model to relatively shallow water, allowing for their application to the entire shoaling zone or deeper. At the same time, the use of so-called "fully-nonlinear" formulations (e.g., Wei et al (1995) and many others) effectively eliminates the restriction to weak nonlinearity by removing the wave height to water depth ratio as a scaling or expansion parameter in the development of approximate governing equations. This approach has improved model applicability in the surf and swash zones particularly, where surface fluctuations are of the order of mean water depth at least and which can represent the total vertical extent of the water column in swash conditions. Representations of dissipative wave-breaking events, which do not naturally arise as weak discontinuous solutions in the dispersive Boussinesq formulation, have been developed usually following an eddy viscosity formulation due to Zelt (1991), and have been shown to be highly effective in describing surf zone wave height decay. The resulting class of models has been shown to be highly effective in modeling wave-averaged surf zone flows over both simple (Chen et al, 2003; Feddersen et al, 2011) and complex (Geiman et al, 2011) bathymetries. Kim et al (2009) have further extended the formulation to incorporate a consistent representation of boundary layer turbulence effects on vertical flow structure.

Existing approaches to development of numerical implementations for Boussinesq models include a wide range of finite difference, finite volume, or finite element formulations. In this paper, we describe the development of a new numerical approach for the FUNWAVE model (Kirby et al, 1998), which has been widely used as a public domain open-source code since its initial development. FUNWAVE was originally developed using an unstaggered finite difference formulation for spatial derivatives together with an iterated 4th order Adams-Bashforth-Moulton (ABM) scheme for time stepping (Wei and Kirby, 1995), applied to the fully nonlinear model equations of Wei et al (1995). In this scheme, spatial differencing is handled using a mixed-order approach, employing fourth-order accurate centered differences for first derivatives and second-order accurate differences for third derivatives. This choice was made in order to move leading order truncation errors to one order higher than the $O(\mu^2)$ dispersive terms (where μ is ratio of a characteristic water depth to a horizontal length, a dimensionless parameter characterizing frequency dispersion), while maintaining the tridiagonal structure of spatial derivatives within time-derivative terms. This scheme is straightforward to develop and has been widely utilized in other Boussinesq models.

Kennedy et al (2000) and Chen et al (2000) describe further aspects of the model system aimed at generalizing it for use in modeling surf zone flows. Breaking is handled using a generalization to two horizontal dimensions of the eddy viscosity model of Zelt (1991). Similar approaches have been used by other Boussinesq model developers, such as Nwogu and Demirbilek (2001), who used a more sophisticated eddy viscosity model in which the eddy viscosity is expressed in terms of turbulent kinetic energy and a length scale. The presence of a moving shoreline in the swash zone is handled using a "slot" or porous-beach method, in which the entire domain remains wetted using a network of slots at grid resolution which are narrow but which extend down to a depth lower than the minimum expected excursion of the modelled free surface. These generalizations form the basis for the existing public domain version of the code (Kirby et al, 1998). Subsequently, several extensions have been made in research versions of the code. Kennedy et al (2001) improved nonlinear performance of the model by utilizing an adaptive reference level for vertical series expansions, which is allowed to move up and down with local surface fluctuations. Chen et al (2003) extended the model to include longshore periodic boundary conditions and described it's application to modeling longshore currents on relatively straight coastlines. Shi et al (2001) generalized the model coordinate system to non-orthogonal curvilinear coordinates. Finally, Chen et al (2003) and Chen (2006) provided revised model equations which correct deficiencies in the representation of higher order advection terms, leading to a set of model equations which, in the absence of dissipation effects, conserve depth-integrated potential vorticity to $O(\mu^2)$, consistent with the level of approximation in the model equations.

The need for a new formulation for FUNWAVE arose from recognition of several crucial problems with the existing code. First, the original model proved to be "noisy", or at least weakly unstable to high wavenumbers near the grid Nyquist limit. In addition, both the implementation of the eddy viscosity model for surf zone wave breaking and the interaction of runup with beach slots proved to be additional sources of noise in model calculations. In the original unstaggered grid finite difference scheme, these effects led to the need for periodic application of dissipative filters, with the frequency of filtering increased in areas with active breaking. The overall grid-based noise generation was found to stem from several sources. First, Kennedy (2001, personal communications), during development of a High Performance Fortran (HPF) version of the code on an early linux cluster, discovered that the noise was partially due to implementation of boundary conditions and could be alleviated to an extent. At the same time, the curvilinear model developed by Shi et al (2001) was implemented using a staggered grid for spatial differencing, and the resulting code was found to be less susceptible to the general grid noise problem. A comparison of noise levels in the original FUNWAVE, the staggered grid scheme of Shi et al, and the corrected boundary condition formulation of Kennedy may be found in Zhen (2004).

Additional sources of noise related either to the eddy viscosity formulation or interaction between the flow and beach "slots" remain less well analyzed or understood. In addition, the performance of the slots themselves has been called into question in several cases involving inundation over complex bathymetry. Slots which are too wide relative to the model grid spacing may admit too much fluid before filling during runup, and cause both a reduction in amplitude and a phase lag in modeled runup events. At the other extreme, slots which are too narrow tend to induce a

great deal of numerical noise, leading to the need for intermittent or even fairly frequent filtering of swash zone solutions. Poor model performance in comparison to data for the case of Swigler and Lynett (2011), described below in the example section, was the determining factor in the decision to pursue a new model formulation.

A number of recently development Boussinesq-type wave models have used a hybrid method combining the finite-volume and finite-difference TVD (Total Variation Diminishing)-type schemes (Toro, 2009), and have shown robust performance of the shock-capturing method in simulating breaking waves and coastal inundation (Tonelli and Petti, 2009, 2010, Roeber et al., 2010, Shiach and Mingham, 2009, Erduran et al., 2005, and others). The use of the hybrid method, in which the underlying components of the nonlinear shallow water equations (which form the basis of the Boussinesq model equations) are handled using the TVD finite volume method while dispersive terms are implemented using conventional finite differencing, provides a robust framework for modeling of surf zone flows. In particular, wave breaking may be handled entirely by the treatment of weak solutions in the shock-capturing TVD scheme, making the implementation of an explicit formulation for breaking wave dissipation unnecessary. In addition, shoreline movement may be handled quite naturally as part of the Reiman solver underlying the finite volume scheme.

In this version, we describe the development of a hybrid finite volume - finite difference scheme for the fully nonlinear Boussinesq model equations of Chen (2006), extended to incorporate a moving reference level as in Kennedy et al (2001). The use of a moving reference elevation is more consistent with a time-varying representation of elevation at a moving shoreline in modeling of a swash zone dynamics and coastal inundation. A conservative form of the equations is derived. Dispersive terms are reorganized with the aim of constructing a tridiagonal structure of spatial derivatives within time-derivative terms. The surface elevation gradient term are also rearranged to obtain a numerically well-balanced form, which is suitable for any numerical order. In contrast to previous high-order temporal schemes, which usually require uniform time-stepping, we use adaptive time stepping based on a third-order Runge-Kutta method. Spatial derivatives are discretized using a combination of finite-volume and finite-difference methods. A high-order MUSCL(Monotone Upstream-centered Schemes for Conservation Laws) reconstruction technique, which is accurate up to the fourth-order, is used in the Riemann solver. The wave breaking scheme follows the approach of Tonelli and Petti (2009), who used the ability of the nonlinear shallow water equations (NLSWE) with a TVD solver to simulate moving hydraulic jumps. Wave breaking is modeled by switching from Boussinesq to NSWE at cells where the Froude number exceeds a certain threshold. The original eddy viscosity breaking scheme is retained in Version 3.0. A wetting-drying scheme is used to model a moving shoreline.

The model was parallelized using the domain decomposition technique. The Message Passing Interface (MPI) with non-blocking communication is used for data communication between processors.

This report provides derivations of the conservation form of theoretical equations with a well-balanced pressure gradient term, numerical schemes, and users' manual. The last part of report illustrates the model's applications to problems of wave breaking and runup in the context of a standard suite of benchmark tests. In addition, we include a documentation of the spherical Boussinesq

model used for Tsunami wave simulations. The spherical Boussinesq model was based on Kirby et al. (2004, 2012) and implemented in the same model framework.

2 Theory

In this section, we describe the development of a set of Boussinesq equations which are accurate to $O(\mu^2)$ in dispersive effects. Here, μ is a parameter characterizing the ratio of water depth to wave length, and is assumed to be small in classical Boussinesq theory. We retain dimensional forms below but will refer to the apparent $O(\mu^2)$ ordering of terms resulting from deviations from hydrostatic behavior in order to identify these effects as needed. The model equations used here follow from the work of Chen (2006). In this and earlier works starting with Nwogu (1993), the horizontal velocity is written as

$$\mathbf{u} = \mathbf{u}_\alpha + \mathbf{u}_2(z) \tag{1}$$

Here, $\mathbf{u}_\alpha$ denotes the velocity at a reference elevation $z = z_\alpha$, and

$$\mathbf{u}_2(z) = (z_\alpha - z)\nabla A + \frac{1}{2}\left(z_\alpha^2 - z^2\right)\nabla B \tag{2}$$

represents the depth-dependent correction at $O(\mu^2)$, with A and B given by

$$\begin{aligned} A &= \nabla \cdot (h\mathbf{u}_\alpha) \\ B &= \nabla \cdot \mathbf{u}_\alpha \end{aligned} \tag{3}$$

The derivation follows Chen (2006) except for the additional effect of letting the reference elevation z_α vary in time according to

$$z_\alpha = \zeta h + \beta\eta \tag{4}$$

where h is local still water depth, η is local surface displacement and ζ and β are constants, as in Kennedy et al (2001). This addition does not alter the details of the derivation, which are omitted below.

2.1 Governing equations

The equations of Chen (2006) extended to incorporate a possible moving reference elevation follow. The depth-integrated volume conservation equation is given by

$$\eta_t + \nabla \cdot \mathbf{M} = 0 \tag{5}$$

where

$$\mathbf{M} = H\left\{\mathbf{u}_\alpha + \overline{\mathbf{u}}_2\right\} \tag{6}$$

is the horizontal volume flux. $H = h + \eta$ is the total local water depth and $\overline{\mathbf{u}}_2$ is the depth averaged $O(\mu^2)$ contribution to the horizontal velocity field, given by

$$\overline{\mathbf{u}}_2 = \frac{1}{H}\int_{-h}^{\eta} \mathbf{u}_2(z)dz = \left(\frac{z_\alpha^2}{2} - \frac{1}{6}(h^2 - h\eta + \eta^2)\right)\nabla B + \left(z_\alpha + \frac{1}{2}(h-\eta)\right)\nabla A \tag{7}$$

The depth-averaged horizontal momentum equation can be written as

$$\mathbf{u}_{\alpha,t} + (\mathbf{u}_\alpha \cdot \nabla)\mathbf{u}_\alpha + g\nabla\eta + \mathbf{V}_1 + \mathbf{V}_2 + \mathbf{V}_3 + \mathbf{R} = 0 \tag{8}$$

where g is the gravitational acceleration and $\mathbf{R}$ represents diffusive and dissipative terms including bottom friction and subgrid lateral turbulent mixing. $\mathbf{V}_1$ and $\mathbf{V}_2$ are terms representing the dispersive Boussinesq terms given by

$$\mathbf{V}_1 = \left\{\frac{z_\alpha^2}{2}\nabla B + z_\alpha \nabla A\right\}_t - \nabla\left[\frac{\eta^2}{2}B_t + \eta A_t)\right] \tag{9}$$

$$\mathbf{V}_2 = \nabla\left\{(z_\alpha - \eta)(\mathbf{u}_\alpha \cdot \nabla)A + \frac{1}{2}(z_\alpha^2 - \eta^2)(\mathbf{u}_\alpha \cdot \nabla)B + \frac{1}{2}[A + \eta B]^2\right\} \tag{10}$$

The form of (9) allows for the reference level z_α to be treated as a time-varying elevation, as suggested in Kennedy et al (2001). If this extension is neglected, the terms reduce to the form given originally by Wei et al (1995). The expression (10) for $\mathbf{V}_2$ was also given by Wei et al (1995), and is not altered by the choice of a fixed or moving reference elevation.

The term $\mathbf{V}_3$ in (8) represents the $O(\mu^2)$ contribution to the expression for $\boldsymbol{\omega} \times \mathbf{u} = \omega \mathbf{i}^z \times \mathbf{u}$ (with $\mathbf{i}^z$ the unit vector in the z direction) and may be written as

$$\mathbf{V}_3 = \omega_0 \mathbf{i}^z \times \overline{\mathbf{u}}_2 + \omega_2 \mathbf{i}^z \times \mathbf{u}_\alpha \tag{11}$$

where

$$\omega_0 = (\nabla \times \mathbf{u}_\alpha) \cdot \mathbf{i}^z = v_{\alpha,x} - u_{\alpha,y} \tag{12}$$

$$\omega_2 = (\nabla \times \mathbf{u}_2) \cdot \mathbf{i}^z = z_{\alpha,x}(A_y + z_\alpha B_y) - z_{\alpha,y}(A_x + z_\alpha B_x) \tag{13}$$

Note that there is a typo in (13) in Shi et al. (2012) ($\overline{\mathbf{u}}_2$ should be $\mathbf{u}_2$). Following Nwogu (1993), z_α is usually chosen in order to optimize the apparent dispersion relation of the linearized model relative to the full linear dispersion in some sense. In particular, the choice $\alpha = (z_\alpha/h)^2/2 + z_\alpha/h = -2/5$ recovers a Padé approximant form of the dispersion relation, while the choice $\alpha = -0.39$, corresponding to the choice $z_\alpha = -0.53h$, minimizes the maximum error in wave phase speed occurring over the range $0 \le kh \le \pi$. Kennedy et al (2001) showed that, allowing z_α to move up and down with the passage of the wave field, allowed a greater degree of flexibility in optimizing nonlinear behavior of the resulting model equations. In the examples chosen here, where a great deal of our focus is on the behavior of the model from the break point landward, we adopt Kennedy et al's "datum invariant" form

$$z_\alpha = -h + \beta H = (\beta - 1)h + \beta\eta = \zeta h + (1 + \zeta)\eta \tag{14}$$

with $\zeta = -0.53$ as in Nwogu (1993) and $\beta = 1 + \zeta = 0.47$ This corresponds in essence to a σ coordinate approach, which places the reference elevation at a level 53% of the total local depth below the local water surface. This also serves to keep the model reference elevation within the actual water column over the entire wetted extent of the model domain.

2.2 Treatment of the surface gradient term

The hybrid numerical scheme requires a conservative form of continuity equation and momentum equations, thus requiring a modification of the leading order pressure term in the momentum equation. A numerical imbalance problem occurs when the surface gradient term is conventionally split into an artificial flux gradient and a source term that includes the effect of the bed slope for a non-uniform bed. To eliminate errors introduced by the traditional depth gradient method (DGM), a so-called surface gradient method (SGM) proposed by Zhou et al. (2001) was adopted in the TVD based-Boussinesq models in the recent literatures. Zhou et al. discussed an example of SGM in 1-D and verified that the slope-source term may be canceled out by part of the numerical flux term associated with water depth, if the bottom elevation at the cell center is constructed using the average of bottom elevations at two cell interfaces. Zhou et al. also showed a 2D application, but without explicitly describing 2D numerical schemes. Although this scheme can be extended into 2D following the same procedure as in 1D, it was found that the 2D extension may not be trivial in terms of the bottom construction for a 2D arbitrary bathymetry. Kim et al. (2008) pointed out that the water depth in the slope-source term should be written in a discretized form rather than the value obtained using the bottom construction, implying that their revised SGM is valid for general 2D applications.

For the higher-order schemes, such as the fourth-order MUSCL-TVD scheme (Yamamoto and Daiguji, 1993, Yamamoto et al., 1998) used in the recent Boussinesq applications, the original SGM and the revised SGM may not be effective in removing the artificial source. This problem was recently noticed by some authors, such as Roeber et al. (2010), who kept a first-order scheme (second-order for normal conditions) for the numerical flux term and the slope-source term in order to ensure a well-balanced solution, without adding noise for a rapidly varying bathymetry.

In fact, the imbalance problem can be solved by a reformulation of this term in terms of deviations from an unforced but separately specified equilibrium state (see general derivations in Rogers et al., 2003 and recent application in Liang and Marche, 2009). Using this technique, the surface gradient term may be split as

$$gH\nabla\eta = \nabla\left[\frac{1}{2}g(\eta^2 + 2h\eta)\right] - g\eta\nabla h \tag{15}$$

which is well-balanced for any numerical order under an unforced stationary condition (still water condition).

2.3 Conservative form of fully nonlinear Boussinesq equations in Cartesian coordinates

For Chen's (2006) equations or the minor extension considered here, $H\mathbf{u}_\alpha$ can be used as a conserved variable in the construction of a conservative form of Boussinesq equations, but this results in a source term in the mass conservation equation, such as in Shiach and Mingham (2009) and Roeber et al. (2010). An alternative approach is to use $\mathbf{M}$ as a conserved variable in terms of the physical meaning of mass conservation. In this study, we used $\mathbf{M}$, instead of $H\mathbf{u}_\alpha$, in the following derivations of the conservative form of the fully nonlinear Boussinesq equations.

Using $\mathbf{M}$ from (6) together with the vector identity

$$\nabla \cdot (\mathbf{uv}) = \nabla \mathbf{u} \cdot \mathbf{v} + (\nabla \cdot \mathbf{v})\mathbf{u} \tag{16}$$

allows (8) to be rearranged as

$$\begin{aligned} &\mathbf{M}_t + \nabla \cdot \left(\frac{\mathbf{MM}}{H} \right) + gH\nabla\eta \\ &= H\left\{ \overline{\mathbf{u}}_{2,t} + \mathbf{u}_\alpha \cdot \nabla \overline{\mathbf{u}}_2 + \overline{\mathbf{u}}_2 \cdot \nabla \mathbf{u}_\alpha - \mathbf{V}_1 - \mathbf{V}_2 - \mathbf{V}_3 - \mathbf{R} \right\} \end{aligned} \tag{17}$$

Following Wei et al. (1995), we separate the time derivative dispersion terms in $\mathbf{V}_1$ according to

$$\mathbf{V}_1 = \mathbf{V}'_{1,t} + \mathbf{V}''_1 \tag{18}$$

where

$$\mathbf{V}'_1 = \frac{z_\alpha^2}{2}\nabla B + z_\alpha \nabla A - \nabla \left[\frac{\eta^2}{2} B + \eta A \right] \tag{19}$$

and

$$\mathbf{V}''_1 = \nabla \left[\eta_t (A + \eta B) \right] \tag{20}$$

Using (15), (19) and (20), the momentum equation can be rewritten as

$$\begin{aligned} &\mathbf{M}_t + \nabla \cdot \left[\frac{\mathbf{MM}}{H} \right] + \nabla \left[\frac{1}{2} g(\eta^2 + 2h\eta) \right] = \\ &= H\left\{ \overline{\mathbf{u}}_{2,t} + \mathbf{u}_\alpha \cdot \nabla \overline{\mathbf{u}}_2 + \overline{\mathbf{u}}_2 \cdot \nabla \mathbf{u}_\alpha - \mathbf{V}'_{1,t} - \mathbf{V}''_1 - \mathbf{V}_2 - \mathbf{V}_3 - \mathbf{R} \right\} + g\eta\nabla h \end{aligned} \tag{21}$$

A difficulty usually arises in applying the adaptive time-stepping scheme to the time derivative dispersive terms $\bar{u}_{2,t}$ and $\mathbf{V}'_{1,t}$, which was usually calculated using values stored in several time levels in the previous Boussinesq codes such as in Wei et al. (1995) and Shi et al. (2001). To prevent this, the equation can be re-arranged by merging the time derivatives on the right hand side into the time derivative term on the left hand side, giving

$$\begin{aligned} \mathbf{V}_t \quad &+\nabla \cdot \left[\frac{\mathbf{MM}}{H} \right] + \nabla \left[\frac{1}{2} g(\eta^2 + 2h\eta) \right] = \eta_t (\mathbf{V}'_1 - \overline{\mathbf{u}}_2) \\ &+H\left(\mathbf{u}_\alpha \cdot \overline{\mathbf{u}}_2 + \overline{\mathbf{u}}_2 \cdot \nabla \mathbf{u}_\alpha - \mathbf{V}''_1 - \mathbf{V}_2 - \mathbf{V}_3 - \mathbf{R} \right) + g\eta\nabla h \end{aligned} \tag{22}$$

where

$$\mathbf{V} = H(\mathbf{u}_\alpha + \mathbf{V}_1') \tag{23}$$

In (23) η_t can be calculated explicitly using (5) as in Roeber et al. (2010). Equations (5) and (22) are the governing equations solved in this study. As $\mathbf{V}$ is obtained, the velocity $\mathbf{u}_\alpha$ can be found by solving a system of equation with tridiagonal matrix formed by (23), in which all cross-derivatives are moved to the right-hand side of the equation.

2.4 Weakly nonlinear Boussinesq equations in spherical coordinates

The weakly nonlinear Boussinesq equations in spherical coordinates are solved in the same model framework. We used the spherical Boussinesq equations derived by Kirby et al. (2004, 2012):

$$H_t + \frac{1}{r_0 \cos\theta}\left\{(Hu)_\phi + (Hv\cos\theta)_\theta\right\} = 0 \tag{24}$$

$$\begin{aligned}
&u_t - fv + \frac{1}{r_0\cos\theta}uu_\phi + \frac{1}{r_0}vu_\theta \ + \frac{g}{r_0\cos\theta}\eta_\phi \\
&+\frac{1}{r_0^2\cos^2\theta}\left\{\frac{h^2}{6}\left[u_{\phi\phi t} + (v\cos\theta)_{\phi\theta t}\right] - \frac{h}{2}\left[(hu_t)_{\phi\phi} + (h\cos\theta v_t)_{\phi\theta}\right]\right\} \\
&-\tau_b^\phi + \frac{1}{r_0\cos\theta}(BFT)_\phi = 0
\end{aligned} \tag{25}$$

$$\begin{aligned}
&v_t + fu + \frac{1}{r_0\cos\theta}uv_\phi + \frac{1}{r_0}vv_\theta + \frac{g}{r_0}\eta_\theta \\
&+\frac{1}{r_0^2}\left\{\frac{h^2}{6}\left[\frac{1}{\cos\theta}\{u_{\phi t} + (v\cos\theta)_{\theta t}\}\right]_\theta - \frac{h}{2}\left[\frac{1}{\cos\theta}\{(hu_t)_\phi + (h\cos\theta v_t)_\theta\}\right]_\theta\right\} \\
&-\tau_b^\theta + \frac{1}{r_0}(BFT)_\theta = 0
\end{aligned} \tag{26}$$

where θ and ϕ denote latitude and longitude, respectively, r_0 is the earth radius, f is the Coriolis parameter, $H = h + \eta$, (u, v) represent the depth-averaged velocity. BFT denotes forcing terms resulting from motion of the ocean bottom and it is not taken into account in the present program.

To facilitate solving the spherical equations in the same model framework as in the Cartesian coordinates, we solve the spherical governing equations by transforming the equations into an equivalent set of equations in Cartesian coordinates using a standard cylindrical projection. We define

$$\begin{cases} \xi_1 = & r_0\cos\theta_0(\phi - \phi_0) \\ \xi_2 = & r_0(\theta - \theta_0) \end{cases} \tag{27}$$

where (ϕ_0, θ_0) are the reference longitude and latitude, respectively. The differentials of ξ_1 and ξ_2

are

$$\begin{cases} d\xi_1 = & r_0 \cos\theta_0 d\phi \\ d\xi_2 = & r_0 d\theta \end{cases} \tag{28}$$

$d\phi$ and $d\theta$ in (24) - (26) are replaced by $d\xi_1$ and $d\xi_2$. Detailed derivations for each term are described below.

$$\frac{1}{r_0\cos\theta}(Hu)\phi = S_p(Hu)_{\xi_1} \tag{29}$$

$$\frac{1}{r_0\cos\theta}(Hv\cos\theta)_\theta = \frac{1}{r_0\cos\theta}\left[\cos\theta(Hv)_\theta - Hv\sin\theta\right] = (Hv)_{\xi_2} - \frac{1}{r_0}\tan\theta Hv \tag{30}$$

$$\frac{1}{r_0\cos\theta}uu_\phi = S_p uu_{\xi_1} \tag{31}$$

$$\frac{1}{r_0}vu_\theta = vu_{\xi_2} \tag{32}$$

$$\frac{1}{r_0\cos\theta}g\eta_\phi = S_p g\eta_{\xi_1} \tag{33}$$

$$\frac{h^2}{6}\frac{1}{r_0^2\cos^2\theta}u_{\phi\phi t} = S_p^2\frac{h^2}{6}u_{\xi_1\xi_1 t} \tag{34}$$

$$\frac{h^2}{6}\frac{1}{r_0^2\cos^2\theta}(v\cos\theta)_{\phi\theta t} = \frac{h^2 S_p}{6}\left(v_{\xi_1\xi_2} - \frac{1}{r_0}\tan\theta v_{\xi_1}\right)_t \tag{35}$$

$$-\frac{h}{2}\frac{1}{r_0^2\cos^2\theta}(hu)_{\phi\phi t} = -\frac{hS_p^2}{2}(hu)_{\xi_1\xi_1 t} \tag{36}$$

$$-\frac{h}{2}\frac{1}{r_0^2\cos^2\theta}(hv\cos\theta)_{\phi\theta t} = -\frac{hS_p}{2}\left[(hv)_{\xi_1\xi_2} - \frac{1}{r_0}\tan\theta(hv)_{\xi_1}\right]_t \tag{37}$$

$$\frac{1}{r_0\cos\theta}uv_\phi = S_p uv_{\xi_1} \tag{38}$$

$$\frac{1}{r_0}vv_\theta = vv_{\xi_2} \tag{39}$$

$$\frac{1}{r_0}g\eta_\theta = g\eta_{\xi_2} \tag{40}$$

$$\frac{h^2}{6}\frac{1}{r_0^2}\left(\frac{u_{\phi t}}{\cos\theta}\right)_\theta = \frac{h^2 S_p}{6}\left(u_{\xi_1\xi_2} + \frac{1}{r_0}\tan\theta u_{\xi_1}\right)_t \tag{41}$$

$$-\frac{h}{2}\frac{1}{r_0^2}\left[\frac{(hu)_{\phi t}}{\cos\theta}\right]_\theta = -\frac{hS_p}{2}\left[(hu)_{\xi_1\xi_2} + \frac{1}{r_0}\tan\theta(hu)_{\xi_1}\right]_t \tag{42}$$

$$\frac{h^2}{6}\frac{1}{r_0^2}\left[\frac{(v\cos\theta)_{\theta t}}{\cos\theta}\right]_\theta = \frac{h^2}{6}\left[v_{\xi_2\xi_2} - \frac{1}{r_0}\tan\theta v_{\xi_2} - \frac{1}{r_0^2\cos^2\theta}v\right]_t \tag{43}$$

$$-\frac{h}{2}\frac{1}{r_0^2}\left[\frac{(hv\cos\theta)_{\theta t}}{\cos\theta}\right]_\theta = -\frac{h}{2}\left[(hv)_{\xi_2\xi_2} - \frac{1}{r_0}\tan\theta(hv)_{\xi_2} - \frac{1}{r_0^2\cos^2\theta}hv\right]_t \tag{44}$$

where S_p is a spherical coordinate correction factor expressed by

$$S_p = \frac{\cos\theta_0}{\cos\theta}. \tag{45}$$

The equations in (ξ_1, ξ_2) coordinates are

$$H_t + S_p(Hu)_{\xi_1} + (Hv)_{\xi_2} = \frac{1}{r_0}\tan\theta Hv \tag{46}$$

$$u_t + S_p u u_{\xi_1} + v u_{\xi_2} - fv + S_p g\eta_{\xi_1} + F_t - \tau_b^{\xi_1} + S_p(BFT)_{\xi_1} = 0 \tag{47}$$

$$v_t + S_p u v_{\xi_1} + v v_{\xi_2} + fu + g\eta_{\xi_2} + G_t - \tau_b^{\xi_2} + (BFT)_{\xi_2} \tag{48}$$

where

$$F = \frac{h^2 S_p^2}{6} u_{\xi_1\xi_1} + \frac{h^2 S_p}{6}(v_{\xi_1\xi_2} - \frac{1}{r_0}\tan\theta v_{\xi_1}) - \frac{hS_p^2}{2}(hu)_{\xi_1\xi_1} - \frac{hS_p}{2}\left[(hv)_{\xi_1\xi_2} - \frac{1}{r_0}\tan\theta(hv)_{\xi_1}\right] \tag{49}$$

$$\begin{aligned} G = {} & \frac{h^2 S_p}{6}\left(u_{\xi_1\xi_2} + \frac{1}{r_0}\tan\theta u_{\xi_1}\right) + \frac{h^2}{6}\left(v_{\xi_2\xi_2} - \frac{1}{r_0}\tan\theta v_{\xi_2} - \frac{1}{r_0^2\cos^2\theta}v\right) \\ & - \frac{hS_p}{2}\left[(hu)_{\xi_1\xi_2} + \frac{1}{r_0}\tan\theta(hu)_{\xi_1}\right] - \frac{h}{2}\left[(hv)_{\xi_2\xi_2} - \frac{1}{r_0}\tan\theta(hv)_{\xi_2} - \frac{1}{r_0^2\cos^2\theta}(hv)\right] \end{aligned} \tag{50}$$

The conservation forms of the spherical equations can be rewritten in the same form of the Cartesian equations (57) and $\mathbf{\Psi}$,$\mathbf{\Theta}$ and $\mathbf{S}$ are defined as

$$\mathbf{\Psi} = \begin{pmatrix} \eta \\ U_s \\ V_s \end{pmatrix}, \qquad \mathbf{\Theta} = \begin{pmatrix} S_p P\mathbf{i} + Q\mathbf{j} \\ \left[\frac{S_p P^2}{h+\eta} + \frac{1}{2}S_p g(\eta^2 + 2\eta h)\right]\mathbf{i} + \frac{PQ}{h+\eta}\mathbf{j} \\ \frac{S_p PQ}{h+\eta}\mathbf{i} + \left[\frac{Q^2}{h+\eta} + \frac{1}{2}g(\eta^2 + 2\eta h)\right]\mathbf{j} \end{pmatrix}. \tag{51}$$

$$\mathbf{S} = \begin{pmatrix} \frac{1}{r_0}\tan\theta(h+\eta)v \\ S_p g\eta\frac{\partial h}{\partial\xi_1} + f(h+\eta)v + \tau_b^{\xi_1} + \psi_1 \\ g\eta\frac{\partial h}{\partial\xi_2} - f(h+\eta)u + \tau_b^{\xi_2} + \psi_2 \end{pmatrix}, \tag{52}$$

where $P = (h+\eta)u$, $Q = (h+\eta)v$,

$$U_s = (h+\eta)(u+F) \tag{53}$$

$$V_s = (h+\eta)(v+G) \tag{54}$$

$$\psi_1 = \eta_t F \tag{55}$$

$$\psi_2 = \eta_t G \tag{56}$$

3 Numerical schemes

3.1 Compact form of governing equations

We define

$$\begin{aligned}
&\mathbf{u}_\alpha = (u, v),\\
&\overline{\mathbf{u}}_2 = (U_4, V_4),\\
&\mathbf{M} = (P, Q) = H\left[u + U_4, v + V_4\right],\\
&\mathbf{V}_1' = (U_1', V_1'),\\
&\mathbf{V}_1'' = (U_1'', V_1''),\\
&\mathbf{V}_2 = (U_2, V_2),\\
&\mathbf{V} = (U, V) = H\left[(u + U_1'), (v + V_1')\right].
\end{aligned}$$

The generalized conservative form of Boussinesq equations can be written as

$$\frac{\partial \mathbf{\Psi}}{\partial t} + \nabla \cdot \mathbf{\Theta}(\mathbf{\Psi}) = \mathbf{S} \tag{57}$$

where $\mathbf{\Psi}$ and $\mathbf{\Theta}(\mathbf{\Psi})$ are the vector of conserved variables and the flux vector function, respectively, and are given by

$$\mathbf{\Psi} = \begin{pmatrix} \eta \\ U \\ V \end{pmatrix}, \qquad \mathbf{\Theta} = \begin{pmatrix} P\mathbf{i} + Q\mathbf{j} \\ \left[\frac{P^2}{H} + \frac{1}{2}g(\eta^2 + 2\eta h)\right]\mathbf{i} + \frac{PQ}{H}\mathbf{j} \\ \frac{PQ}{H}\mathbf{i} + \left[\frac{Q^2}{H} + \frac{1}{2}g(\eta^2 + 2\eta h)\right]\mathbf{j} \end{pmatrix}. \tag{58}$$

$$\mathbf{S} = \begin{pmatrix} 0 \\ g\eta\frac{\partial h}{\partial x} + \psi_x + HR_x \\ g\eta\frac{\partial h}{\partial y} + \psi_y + HR_y \end{pmatrix}, \tag{59}$$

where

$$\psi_x = \eta_t(U_1' - U_4) + H\left(uU_{4,x} + vU_{4,y} + U_4u_x + V_4u_y - U_1'' - U_2 - U_3\right) \tag{60}$$

$$\psi_y = \eta_t(V_1' - V_4) + H\left(uV_{4,x} + vV_{4,y} + U_4v_x + V_4v_y - V_1'' - V_2 - V_3\right) \tag{61}$$

The expanded forms of $(U_1', V_1'), (U_1'', V_1''), (U_2, V_2), (U_3, V_3)$ and (U_4, V_4) can be found in Appendix A. For the term $\mathbf{R}$, the bottom stress is approximated using a quadratic friction equation. A Smagorinsky (1963)-like subgrid turbulent mixing algorithm is implemented following Chen et al. (1999).

3.2 Spatial discretization

A combined finite-volume and finite-difference method is applied to the spatial discretization. For the flux terms and the first-order derivative terms, both high-order and low-order MUSCL-TVD schemes are implemented. In previous version of FUNWAVE-TVD model (lower than Version 3.0), the fourth-order MUSCL-TVD scheme by Yamamoto et al. (1998) was used. However, in some cases, numerical instability occurred in long time simulations.

In present version, the fourth-order MUSCL-TVD scheme by Erduran et al. (2005) who modified Yamamoto et al.'s (1998) fourth-order approach is applied for high-order scheme. In x-direction, for example, the interface construction by the fourth-order MUSCL-TVD scheme can be written as follows:

$$\phi^L_{i+1/2} = \phi_i + \frac{1}{6}\left[\chi(r)\Delta^*\phi_{i-1/2} + 2\,\chi(1/r)\Delta^*\phi_{i+1/2}\right] \tag{62}$$

$$\phi^R_{i-1/2} = \phi_i - \frac{1}{6}\left[2\,\chi(r)\Delta^*\phi_{i-1/2} + \chi(1/r)\Delta^*\phi_{i+1/2}\right] \tag{63}$$

where $\phi^L_{i+1/2}$ is the constructed value at the left-hand side of the interface $i+\frac{1}{2}$ and $\phi^R_{i-1/2}$ is the value at the right-hand side of the interface $i-\frac{1}{2}$. The values of $\Delta^*\phi$ are evaluated as follows:

$$\begin{aligned}
&\Delta^*\phi_{i+1/2} = \Delta\phi_{i+1/2} - \Delta^3\bar{\phi}_{i+1/2}/6,\\
&\Delta\phi_{i+1/2} = \phi_{i+1} - \phi_i,\\
&\Delta^3\bar{\phi}_{i+1/2} = \Delta\bar{\phi}_{i+3/2} - 2\Delta\bar{\phi}_{i+1/2} + \Delta\bar{\phi}_{i-1/2},\\
&\Delta\bar{\phi}_{i-1/2} = \text{minmod}(\Delta\phi_{i-1/2}, \Delta\phi_{i+1/2}, \Delta\phi_{i+3/2}),\\
&\Delta\bar{\phi}_{i+1/2} = \text{minmod}(\Delta\phi_{i+1/2}, \Delta\phi_{i+3/2}, \Delta\phi_{i-1/2}),\\
&\Delta\bar{\phi}_{i+3/2} = \text{minmod}(\Delta\phi_{i+3/2}, \Delta\phi_{i-1/2}, \Delta\phi_{i+1/2})
\end{aligned} \tag{64}$$

In (64), minmod represents the Minmod limiter and is given by

$$\text{minmod}(j,k,l) = \text{sign}(j)\max\{0, \min[|j|, 2\text{sign}(j)k, 2\text{sign}(j)l]\}. \tag{65}$$

$\chi(r)$ in (62) and (63) is the limiter function. The original scheme introduced by Yamamoto et al. (1998) uses the Minmod limiter as used in (64). Erduran et al. (2005) found that the use of the van-Leer limiter for the third-order part gives more accurate results. The van-Leer limiter can be expressed as

$$\chi(r) = \frac{r+|r|}{1+r}, \tag{66}$$

where

$$r = \frac{\Delta^*\phi_{i+1/2}}{\Delta^*\phi_{i-1/2}}. \tag{67}$$

For the low-order scheme, the second-order MUSCL-TVD scheme with van-Leer limiter is applied. The reconstruction at cell interface can be written as (Zhou et al., 2001)

$$\phi = \phi_i + (x - x_i)\sigma\phi_i. \tag{68}$$

or,

$$\phi^L_{i+1/2} = \phi_i + \frac{1}{2}\Delta x\,\sigma\phi_i, \quad \phi^R_{i-1/2} = \phi_i - \frac{1}{2}\Delta x\,\sigma\phi_i, \tag{69}$$

where

$$\sigma\phi_i = \Upsilon\left(\frac{\phi_{i+1} - \phi_i}{x_{i+1} - x_i}, \frac{\phi_i - \phi_{i-1}}{x_i - x_{i-1}}\right). \tag{70}$$

The van Leer limiter in the second-order MUSCL-TVD scheme can be written as

$$\Upsilon(a, b) = \frac{a|b| + |a|b}{|a| + |b|}. \tag{71}$$

If a numerical solution from the fourth-order scheme can not be converged, for example, in a case of modeling extremely large wave height, the second-order MUSCL-TVD scheme is recommended. However, a grid resolution higher than that used for the fourth-order scheme may be required to ensure model accuracy.

The numerical fluxes are computed using a HLL approximate Riemann solver

$$\mathbf{\Theta}(\mathbf{\Psi}^L, \mathbf{\Psi}^R) = \begin{cases} \mathbf{\Theta}(\mathbf{\Psi}^L) & \text{if} \quad s_L \geq 0 \\ \mathbf{\Theta}^*(\mathbf{\Psi}^L, \mathbf{\Psi}^R) & \text{if} \quad s_L < 0 < s_R \\ \mathbf{\Theta}(\mathbf{\Psi}^R) & \text{if} \quad s_R \leq 0, \end{cases} \tag{72}$$

where

$$\mathbf{\Theta}^*(\mathbf{\Psi}^L, \mathbf{\Psi}^R) = \frac{s_R\mathbf{\Theta}(\mathbf{\Psi}^L) - s_L\mathbf{\Theta}(\mathbf{\Psi}^R) + s_L s_R(\mathbf{\Psi}^R - \mathbf{\Psi}^L)}{s_R - s_L} \tag{73}$$

The wave speeds of the Riemann solver are given by

$$s_L = \min(\mathbf{V}^L \cdot \mathbf{n} - \sqrt{g(h+\eta)^L}, u_s - \sqrt{\varphi_s}), \tag{74}$$

$$s_R = \max(\mathbf{V}^R \cdot \mathbf{n} + \sqrt{g(h+\eta)^R}, u_s + \sqrt{\varphi_s}), \tag{75}$$

in which u_s and φ_s are estimated as

$$u_s = \frac{1}{2}(\mathbf{V}^L + \mathbf{V}^R) \cdot \mathbf{n} + \sqrt{g(\eta+h)^L} - \sqrt{g(\eta+h)^R} \tag{76}$$

$$\sqrt{\varphi_s} = \frac{\sqrt{g(\eta+h)^L} + \sqrt{g(\eta+h)^R}}{2} + \frac{(\mathbf{V}^L - \mathbf{V}^R) \cdot \mathbf{n}}{4} \tag{77}$$

and $\mathbf{n}$ is the normalized side vector for a cell face.

Higher derivative terms in ψ_x and ψ_y were discretized using a central difference scheme at the cell centroids, as in Wei et al. (1995). No discretization of dispersion terms at the cell interfaces is needed due to using $\mathbf{M}$ as a flux variable. The Surface Gradient Method (Zhou et al, 2001) was used to eliminate unphysical oscillations. Because the pressure gradient term is re-organized as in section 2.2, there is no imbalance issue for the high-order MUSCL scheme.

3.3 Time stepping

The third-order Strong Stability-Preserving (SSP) Runge-Kutta scheme for nonlinear spatial discretization (Gottlieb et al., 2001) was adopted for time stepping. The scheme is given by

$$\begin{aligned}
\mathbf{\Psi}^{(1)} &= \mathbf{\Psi}^n + \Delta t(-\nabla\cdot\mathbf{\Theta}(\mathbf{\Psi}^n) + \mathbf{S}^{(1)}) \\
\mathbf{\Psi}^{(2)} &= \frac{3}{4}\mathbf{\Psi}^n + \frac{1}{4}\left[\mathbf{\Psi}^{(1)} + \Delta t\left(-\nabla\cdot\mathbf{\Theta}(\mathbf{\Psi}^{(1)}) + \mathbf{S}^{(2)}\right)\right] \\
\mathbf{\Psi}^{n+1} &= \frac{1}{3}\mathbf{\Psi}^n + \frac{2}{3}\left[\mathbf{\Psi}^{(2)} + \Delta t\left(-\nabla\cdot\mathbf{\Theta}(\mathbf{\Psi}^{(2)}) + \mathbf{S}^{n+1}\right)\right]
\end{aligned} \tag{78}$$

in which $\mathbf{\Psi}^n$ denotes $\mathbf{\Psi}$ at time level n. $\mathbf{\Psi}^{(1)}$ and $\mathbf{\Psi}^{(2)}$ are values at intermediate stages in the Runge-Kutta integration. As $\mathbf{\Psi}$ is obtained at each intermediate step, the velocity (u, v) can be solved by a system of tridiagonal matrix equations formed by (23). $\mathbf{S}$ needs to be updated using (u, v, η) at the corresponding time step and an iteration is needed to achieve convergence.

An adaptive time step is chosen, following the Courant-Friedrichs-Lewy (CFL) criterion:

$$\Delta t = C\min\left(\min\frac{\Delta x}{|u_{i,j}| + \sqrt{g(h_{i,j}+\eta_{i,j})}}, \min\frac{\Delta y}{|v_{i,j}| + \sqrt{g(h_{i,j}+\eta_{i,j})}}\right) \tag{79}$$

where C is the Courant number and $C = 0.5$ was used in the following examples.

3.4 Wave breaking

There are two breaking algorithms implemented in the model. One takes the advantage of the shock-capturing scheme in TVD. It follows the approach of Tonelli and Petti (2009), who successfully used the ability of NSWE with a TVD scheme to model moving hydraulic jumps. Thus, the fully nonlinear Boussinesq equations are switched to NSWE at cells where the Froude number exceeds a certain threshold. Following Tonelli and Petti, the ratio of wave height to total water depth is chosen as the criterion to switch from Boussinesq to NSWE, with threshold value set to 0.8, as suggested by Tonelli and Petti.

The other one is the original eddy-viscosity scheme used in the previous version of FUNWAVE (Kennedy et al., 2000). To fit the eddy-viscosity method in the TVD scheme, the artificial eddy viscosity terms are modified as below

$$\mathbf{R}_{bx} = \frac{\partial}{\partial x}(\nu\frac{\partial P}{\partial x}) + \frac{\partial}{\partial y}(\nu\frac{\partial P}{\partial y}) \tag{80}$$

$$\mathbf{R}_{by} = \frac{\partial}{\partial y}(\nu\frac{\partial Q}{\partial y}) + \frac{\partial}{\partial x}(\nu\frac{\partial Q}{\partial x}) \tag{81}$$

Note that the form of (80) and (81) is slightly different from that in Kennedy et al. (2000). The present form was found to give a more stable numerical solution as the cross-derivatives removed. In (80) and (81), ν is the artificial eddy viscosity defined by

$$\nu = B\delta_b^2(h+\eta)\eta_t \tag{82}$$

where $\delta_b = 1.2$. In Kennedy et al. (2000), B varies smoothly from 0 to 1 so as to avoid an impulsive start of breaking and the resulting instability. In the present TVD model, because there is no instability problem found, we adopt a constant value $B = 1$ as breaking is initiated

$$B = \begin{cases} 1 & \eta_t \geq \eta_t^* \\ 0 & \eta_t < \eta_t^* \end{cases} \tag{83}$$

The parameter η_t^* determines the onset and cessation of breaking. Following Kennedy et al., a breaking event begins when η_t exceeds some initial threshold value $\eta_t^{(I)}$, as breaking develops, the wave will continue to break until η_t drops below $\eta_t^{(F)}$. However, we do not use the smooth transition as in Kennedy et al. because the present TVD scheme did not encounter any instability problem. The default values of $\eta_t^{(I)}$ and $\eta_t^{(F)}$ are $0.65 \times 2\sqrt{gh}$ and $0.15\sqrt{gh}$ as in the previous FUNWAVE version (i.e., Cbrk1 = 0.65 and Cbrk2=0.15 in the present version, see the example in the next section). However, the recent tests showed $\eta_t^{(I)}$ should be slightly larger to match the laboratory data.

It should be mentioned that, instead of using U and V in the diffusion terms, we used P and Q as shown in (80) and (81). The formulations (80) and (81) combined with the TVD scheme make the model more stable than using U and V.

3.5 Wetting-drying schemes for shallow water

The wetting-drying scheme for modeling a moving boundary is straightforward. The normal flux $\mathbf{n} \cdot \mathbf{M}$ at the cell interface of a dry cell is set to zero. A mirror boundary condition is applied to the fourth-order MUSCL-TVD scheme and discretization of dispersive terms in ψ_x, ψ_y at dry cells. It may be noted that the wave speeds of the Riemann solver (74) and (75) for a dry cell are modified as

$$s_L = \mathbf{V}^L \cdot \mathbf{n} - \sqrt{g(h+\eta)^L}, \quad s_R = \mathbf{V}^L \cdot \mathbf{n} + 2\sqrt{g(h+\eta)^L} \quad \text{(right dry cell)} \tag{84}$$

and

$$s_L = \mathbf{V}^R \cdot \mathbf{n} - \sqrt{g(h+\eta)^R}, \quad s_R = \mathbf{V}^R \cdot \mathbf{n} + 2\sqrt{g(h+\eta)^R} \quad \text{(left dry cell)} \tag{85}$$

3.6 Boundary conditions

We implemented various boundary conditions including wall boundary condition, absorbing boundary condition following Kirby et al. (1998) and periodic boundary condition following Chen et al. (2003).

3.6.1 Sponge layer

The sponge layer technique introduced by Larsen and Dancy (L-D type, 1983) is implemented in the code. In this method, the variables ϕ(i.e., η, u, v) are directly attenuated at every time step:

$$\phi = \phi / C_s \tag{86}$$

where C_s is a damping coefficient function defined by

$$C_s = \alpha_s^{\gamma_x^{i-1}}, \quad i = 1, 2, ..., n \tag{87}$$

where α_s and γ_s are two free parameters. i represents grid numbers. Chen et al. (1999) suggested that $\alpha_s = 2$, $\gamma_s = 0.88 - 0.92$, and $n = 50 - 100$. The length of the sponge layer is usually taken to be one or two times the typical wavelength. Chen et al also pointed out that the damping coefficients for optimal absorption are somewhat case-sensitive.

Recently, some problem was found in application of L-D type sponge layer for long-term simulations. The direct damping method combined with the TVD scheme generates sawtooth noises with a $2dx$ wave length. The sawtooth noises are usually not noticeable due to small magnitudes. However, they grow gradually with time and may become significant in a long term simulation. If this problem occurs, we suggest using the following friction-type or viscous type sponge layers. Using the combination of L-D and friction/viscous sponge layers may remove the sawtooth noises and also make the wave damping more efficient.

The friction-type and viscous type sponge layers directly use the friction terms and diffusion terms existing in the model. The source term for the friction-type sponge can be described as

$$F_{frc} = -C_{sponge} |\mathbf{u}_\alpha| (u_\alpha, v_\alpha) h \tag{88}$$

Note that depth h is added in (88) to make the source term depth-independent in terms of the flux-type momentum equations. For the diffusion-type sponge, the description of diffusion term follows exactly the eddy-viscosity breaking formulation with spatial varying viscosity coefficients ν_{sponge}. Both coefficients are smoothly ramped in space at the sponge layer boundaries. For example, for a sponge layer on the left end of the domain, C_{sponge} can be written as

$$C_{sponge} = C_{max} \left(1 - \tanh \frac{10(i-1)}{I_{\text{width}} - 1} \right) \tag{89}$$

where C_{max} is the maximum value of C_{sponge} used in the sponger layer. i and I_{width} represent point number and the layer width in points. Similar expressions can be obtained for sponge layers on three other ends of the domain as well as the viscous sponge layer.

The width of the sponge layer is usually taken to be two or three wave lengths for the friction-type and viscous sponge layers. Narrow sponge layers can be used for L-D type sponge layer with a good efficiency but sawtooth noises generated by the method is a concern for long-term simulation.

3.6.2 Periodic boundary condition

The periodic boundary condition in y (south/north) direction was implemented in the code.

3.7 Wavemaker

3.7.1 Internal wavemaker theory

Internal wavemaker was implemented based on Wei and Kirby's (1999) two-way internal wavemaker and Chawla and Kirby's (2000) one-way internal wavemaker (under development). Here, we briefly summarize the formulations used in the wavemakers. Detailed theory can be found in Wei and Kirby (1999) and Chawla and Kirby (2000).

Wei and Kirby (1999) followed the approach of Larsen and Dancy (1983) who used an ad-hoc source mechanism where water mass is added and subtracted along a straight source/sink line inside the computing domain. This approach works well in a staggered-grid differencing scheme, where water is essentially being added to or drained from a single grid block. In applying this technique to the Boussinesq model on an unstaggered grid, however, Wei and Kirby found that use of a single source line accused high frequency noise, leading to blowup of the model. They then used a partially distributed mass source $f(x, y, t)$

$$f(x, y, t) = g(x)s(y, t) \tag{90}$$

where $g(x)$ is a Gaussian shape function and $s(y, t)$ the input time series of the magnitude of source function with an assumption that the center of the source region is parallel to the y-axis. The functions $g(x)$ and $s(y, t)$ are defined as

$$g(x) = \exp[-\beta(x - x_s)^2] \tag{91}$$

$$s(y, t) = D\sin(\lambda y - \omega t) \tag{92}$$

where β is the shape coefficient for the source function, and x_s is the central location of the source in the x direction, for a source oriented parallel to the y axis, as shown in Figure 1. D is the magnitude of the source function, $\lambda = k\sin(\theta)$ the wavenumber in the y direction, and k is the linear wavenumber.

For a monochromatic wave or a single wave component of a random wave train, the magnitude D of source function can be determined by

$$D = \frac{2a_0 \cos(\theta)(\omega^2 - \alpha_1 g k^4 h^3)}{\omega k I[1 - \alpha(kh)^2]} \tag{93}$$

where $\alpha = -0.390, \alpha_1 = \alpha + 1/3$, and I is the integral given by

$$I = \int_{-\infty}^{\infty} \exp(-\beta x^2) \exp(-ilx) dx = \sqrt{\frac{\pi}{\beta}} \exp(-l^2/4\beta) \tag{94}$$

where $l = k\cos(\theta)$ is the wavenumber in x direction. In theory, the shape coefficient β can be any number. The larger the value β is, the narrower the source function becomes. The definition of the source function width W is not unique, and here we define W to be the distance between two

coordinates x_1 and x_2 where the corresponding source function heights are equal to $\exp(-5) = 0.0067$ times the maximum height D. Then x_1 and x_2 must satisfy the quadratic equation

$$\beta(x - x_s)^2 = 5 \tag{95}$$

from which the width of source function is given by

$$W = |x_2 - x_1| = 2\sqrt{\frac{5}{\beta}} \tag{96}$$

In the previous version of FUNWAVE (Kirby et al., 1998), it is suggested that W equals about half of the wavelength for monochromatic wave. If L is the wavelength, the requirement of $W = \delta L/2$ (where δ is of order 1) results in

$$\beta = \frac{5}{(\delta L/4)^2} = \frac{80}{\delta^2 L^2} \tag{97}$$

For random waves, the value of β is determined according to the peak frequency component and then used for all components in the wave train. FUNWAVE-TVD follows the criteria for determining β though a narrow W does not seem to cause any problem.

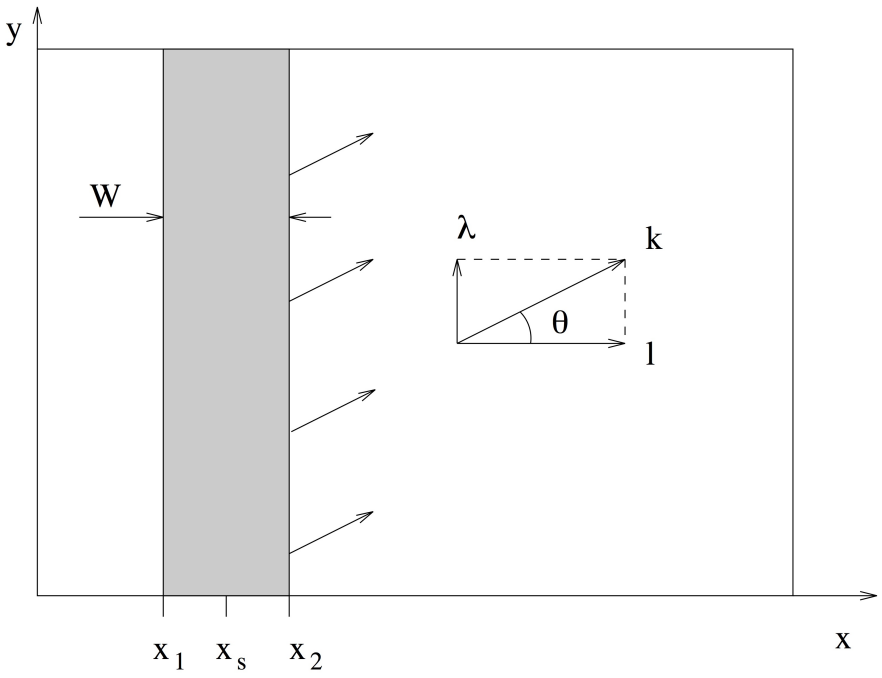

Figure 1: Source function definition in the computing domain.

For the irregular wavemaker, an extension was made to incorporate an alongshore periodicity into wave generation, in order to eliminate a boundary effect on wave simulations. The technique

exactly follows the strategy in Chen et al. (2003), who adjusted the distribution of wave directions in each frequency bin to obtain alongshore periodicity. This approach is effective in modeling of breaking wave-induced nearshore circulation such as alongshore currents and rip currents.

3.7.2 Regular wave generation

The generation of monochromatic wave using the internal wavemaker is straightforward. Following the formulations given in 3.7.1, the magnitude of source function D is calculated by (93) for given wave amplitude a_0, wave angle θ, water depth h and wave period. The source function can be obtained using (92).

3.7.3 Irregular wave generation using directional spectral data

Irregular waves can be generated by integrating wave components split by frequency and direction and with random phases Each wave component contains wave amplitude a_0 converted from wave energy, wave angle θ and wave period. The source function for each component can be obtained using (92).

3.7.4 Irregular wave generation using analytical spectrum function

The input for the wavemaker can be wave bulk parameters or directional spectral data. A TMA shallow-water spectrum and a wrapped-normal directional-spreading function are used to simulate a directional sea state. The combined spectrum function can be expressed as

$$S(f,h,\theta) = E_{TMA}(f,h)G(\theta) \tag{98}$$

In (98), E_{TMA} is the TMA shallow-water frequency distribution as follows

$$E_{TMA}(f,h) = \alpha g^2 f^{-5}(2\pi)^{-4}\Phi(2\pi f,h)e^{-5/4(f/f_p)^{-4}}\gamma^{e^{[-(f/f_p-1)^2/2\sigma^2]}} \tag{99}$$

in which f_p is the peak frequency. γ presents a frequency spreading parameter, and α and σ are coefficients which may be found in Bouws *et al.* (1985). In the model, $\alpha = 1.0$ and

$$\sigma = \begin{cases} 0.07 & \text{if } f \le f_p; \\ 0.09 & \text{if } f > f_p. \end{cases}$$

Φ may be expressed as

$$\Phi(2\pi f,h) = \begin{cases} \frac{1}{2}\omega_h^2 & \text{for } \omega_h \le 1; \\ 1-\frac{1}{2}(2-\omega_h)^2 & \text{for } 2 > \omega_h > 1; \\ 1 & \text{for } \omega_h \ge 2. \end{cases}$$

where

$$\omega_h = 2\pi f(\frac{h}{g})^{1/2}.$$

$G(\theta)$ is the wrapped normal directional spreading function written as

$$G(\theta) = \frac{1}{2\pi} + \frac{1}{\pi}\sum_{n=1}^{N} e^{[-\frac{(n\sigma_\theta)^2}{2}]}\cos n\theta \tag{100}$$

where σ_θ denotes circular deviation of the wrapped normal spreading function. To avoid the computational underflow, $N = 20$ in the model.

In the TMA wavemaker, the directional spectrum is divided into 2100 components with random phases and equal energy at each frequency block. The source function technique (Wei, *et al.*, 1999) is then used for each component and the final surface elevation function can be written as

$$\eta = \sum_{m=1}^{M} C_m \cos\omega_m t + \sum_{m=1}^{M} S_m \sin\omega_m t \tag{101}$$

where

$$\begin{aligned} C_m &= \sum_{n=1}^{k} D_{mn}\cos(k_{mn}y + \varepsilon_{mn}) \\ S_m &= \sum_{n=1}^{k} D_{mn}\sin(k_{mn}y + \varepsilon_{mn}) \end{aligned} \tag{102}$$

in which y-axis is oriented along the main axis of the wave maker. D_{mn}, k_{mn} and ε_{mn} are respectively the amplitude, wave number in the y direction and phase of a component. The phase can be random.

3.8 Other implementations

3.8.1 Wind effect

Wind effects are modeled using the wind stress forcing proposed by Chen et al. (2004). The wind stress is expressed by

$$\mathbf{R}_w = \frac{\rho_a}{\rho} C_{dw} |\mathbf{U}_{10} - \mathbf{C}|(\mathbf{U}_{10} - \mathbf{C}) \tag{103}$$

where ρ_a and ρ represent air density and water density, respectively, $\mathbf{C}$ is wave celerity. The wind stress is only applied on wave crests. A free parameter representing a ratio of the forced crest height to maximum surface elevation is implemented in the model.

3.8.2 Wave height calculation

For spectral wave simulations, both average wave height and root-mean-square wave height are calculated using zero-crossing at each grid point. Significant wave height is evaluated using Goda's (2000) formula

$$H_{1/3} = 4.004\sqrt{m_0} \tag{104}$$

where

$$m_0 = \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} \eta^2 dt \tag{105}$$

3.9 Parallelization

In parallelizing the computational model, we used a domain decomposition technique to subdivide the problem into multiple regions and assign each subdomain to a separate processor core. Each subdomain region contains an overlapping area of ghost cells, three-row deep, as required by the fourth order MUSCL-TVD scheme. The Message Passing Interface (MPI) with non-blocking communication is used to exchange data in the overlapping region between neighboring processors. Velocity components are obtained from Equation (23), by solving tridiagonal matrices using the parallel pipelining tridiagonal solver described in Naik et al. (1993).

4 Users' Manual

4.1 Program outline and flow chart

The code was written using Fortran 90 with the c preprocessor (cpp) statements for separation of the source code. Arrays are dynamically allocated at runtime. Precision is selected using the *selected_real_kind* Fortran intrinsic function defined in the makefile. The default precision is single.

The present version of FUNWAVE-TVD includes a number of options including: choice of serial or parallel code; Cartesian or spherical coordinate (Tsunami propagation mode); samples; one-way nesting mode; wave breaking index and aging (bubble and foam mode); different bottom friction formulas and etc.

The flow chart is shown in Figure 2.

Note that, "ITERATION" described in Shi et al. (2011, 2012a,b) is not used any more. The Range-Kutta time stepping with combined CLF criteria is found to be more efficient than the iteration method.

4.2 Installation and compilation

FUNWAVE-TVD is distributed in a compressed fie. To install the programs, first, uncompress the package. Then use

$>$ tar xvf *.tar

to extract files from the uncompressed package. The exacted files will be distributed in two new directories: /src and /work.

To compile the program, go to /src and modify Makefile if needed. There are several necessary flags in Makefile needed to specify below.

-DDOUBLE_PRECISION: use double precision, default is single precision.

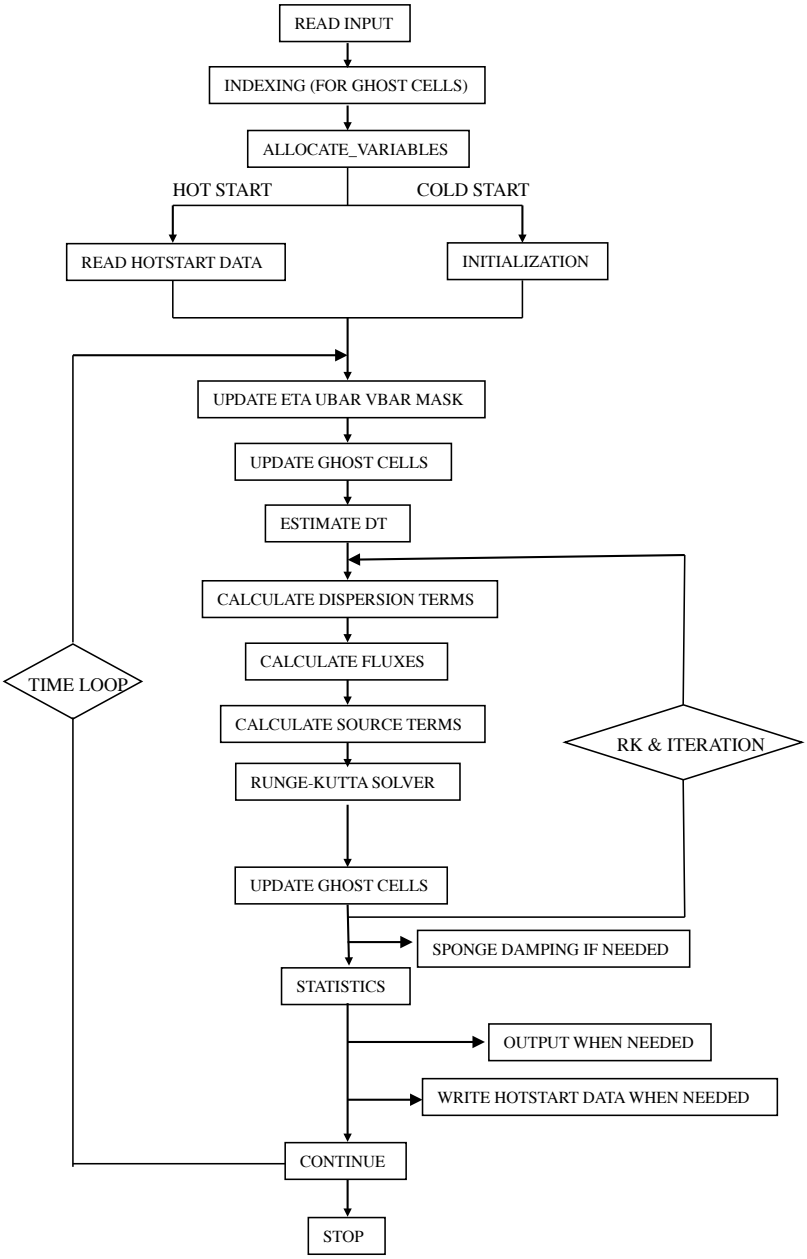

Figure 2: Flow chart of the main program (Note, the ITERATION option is not used any more).

-DPARALLEL: use parallel mode, default is serial mode.

-DSAMPLES: include all samples, default is no sample included.

-DCARTESIAN: Cartesian version, otherwise Spherical version

-DINTEL: if INTEL compiler is used, this option can make use of FPORT for the RAND() function

-DCRAY: for CRAY RAND() and system commands

-DMIXING: include Smagorinsky mixing.

-DCOUPLING: nesting mode.

-DMANNING: Using Manning coefficient for friction calculation.

CPP: path to CPP directory.

FC: Fortran compiler.

Note: -DSAMPLES should be set on in all cases when you are doing any surface wave case.
Then execute
> make

The executable file such as 'funwave' or 'mytvd' will be generated and copied from /src to /work/. If you do not find 'funwave' in your /work/, go to /src/ to find the newest executable file and copy it to your /work/ directory. Note: use 'make clean' after modifying Makefile.

To run the model, go to /work. Modify input.txt if needed and run.

4.3 Input

Following are descriptions of parameters in input.txt (**NOTE:** all parameter names are capital sensitive).

TITLE: title of your case, only used for log file.

SPECIFICATION OF HOT START

HOT_START: logical parameter, T for hot start, F for cold start. Note that hot start in the present version is not available. However, a user can use the option INI_UVZ to restart the model. For this mode, time derivative terms are ignored at the hot start.

FileNumber_HOTSTART: number of hotstart file used for a hot start, e.g., 1,2, ...

SPECIFICATION OF MULTI-PROCESSORS

PX: processor numbers in X

PY : processor numbers in Y

NOTE: PX and PY must be consistency with number of processors defined in mpirun command, e.g., mpirun -np n (where n = px$\times$py). PX (PY) should be a common factor of Mglob(Nglob) in the present version.

SPECIFICATION OF WATER DEPTH

DEPTH_TYPE: depth input type.

DEPTH_TYPE=DATA: from a depth file.

The program includes several simple bathymetry configurations such as

DEPTH_TYPE=FLAT: flat bottom, need DEPTH_FLAT

DEPTH_TYPE=SLOPE: plane beach along x direction. It needs three parameters: slope,SLP, slope starting point, Xslp and flat part of depth, DEPTH_FLAT

DEPTH_FILE: bathymetry file if DEPTH_TYPE=DATA, file dimension should be Mglob x Nglob with the first point as the south-west corner. The read format in the code is shown below.

```
DO J=1,Nglob
READ(1,*)(Depth(I,J),I=1,Mglob)
ENDDO
```

DEPTH_FLAT: water depth of flat bottom if DEPTH_TYPE=FLAT or DEPTH_TYPE=SLOPE (flat part of a plane beach).

SLP: slope if DEPTH_TYPE=SLOPE

Xslp: starting x (m) of a slope, if DEPTH_TYPE=SLOPE

SPECIFICATION OF RESULT FOLDER

RESULT_FOLDER: result folder name, e.g., RESULT_FOLDER = /Users/fengyanshi/tmp/

SPECIFICATION OF DIMENSION

Mglob: global dimension in x direction.

Nglob: global dimension in y direction.

SPECIFICATION OF TIME

TOTAL_TIME: simulation time in seconds

PLOT_INTV: output interval in seconds (Note, output time is not exact because adaptive dt is used.)

SCREEN_INTV: time interval (s) of screen print.

PLOT_INTV_STATION: time interval (s) of gauge output

SPECIFICATION OF GRID SIZE

DX: grid size(m) in x direction.

DY: grid size(m) in y direction.

SPECIFICATION OF INITIAL CONDITION

INT_UVZ : logical parameter for initial condition, default is FALSE

ETA_FILE: name of file for initial η, e.g., ETA_FILE= /Users/fengyanshi/work/input/CVV_H.grd, data format is the same as depth data.

U_FILE: name of file for initial u, e.g.,U_FILE= /Users/fengyanshi/work/input/CVV_U.grd, data format is the same as depth data.

V_FILE: name of file for initial v, e.g., V_FILE= /Users/fengyanshi/work/input/CVV_V.grd, data format is the same as depth data.

MASK_FILE: name of file for initial MASK, e.g., MASK_FILE= /Users/fengyanshi/work/input/CVV_MASK.grd, data format is the same as depth data. Usually a MASK_FILE is from a model output and the format is REAL numbers. If there is no MASK_FILE, MASK values will be re-specified according to ETA and DEPTH.

SPECIFICATION OF WIND EFFECT

WindForce: logical parameter representing if wind effect is taken into account. T or F. Note: spatially uniform wind field is assumed in this version.

WIND_FILE: file name for wind data. The following is an example of data format.

wind data

100 - number of data

0.0 , 10.0 0.0 — time(s), wu, wv (m/s)

2000.0, 10.0, 0.0

8000.0, 10.0, 0.0

...

Cdw: wind stress coefficient for the quadratic formula.

WindCrestPercent: ratio of the forced wave crest height to the maximum surface elevation.

SPECIFICATION OF WAVEMAKER

WAVEMAKER: wavemaker type.

WAVEMAKER = INI_REC: initial rectangular hump, need Xc,Yc and WID

WAVEMAKER = LEF_SOL: left boundary solitary, need AMP,DEP, and LAGTIME

WAVEMAKER = INI_SOL: initial solitary wave, WKN B solution, need AMP, DEP, and XWAVEMAKER

WAVEMAKER = INI_OTH: other initial distribution specified by users

WAVEMAKER = WK_REG: Wei and Kirby 1999 internal wave maker, need Xc_WK, Yc_WK, Tperiod, AMP_WK, DEP_WK, Theta_WK, and Time_ramp (factor of period)

WAVEMAKER = WK_IRR: Wei and Kirby 1999 TMA spectrum wavemaker, need Xc_WK, Yc_WK, DEP_WK, Time_ramp, Delta_WK, FreqPeak, FreqMin,FreqMax, Hmo, GammaTMA, ThetaPeak

WAVEMAKER = WK_TIME_SERIES: *fft* a time series to get each wave component and then use Wei and Kirby's (1999) wavemaker. The wave angle is zero (x direction) for all wave components. Need input WaveCompFile (including 3 columns: per,amp,pha) and NumWaveComp,PeakPeriod,DEP_WK, Xc_WK,Ywidth_WK

WAVEMAKER = WAVE_DATA: 2D directional spectrum data specified in WaveCompFile. Need Xc_WK, Yc_WK, DEP_WK, Delta_WK. See WaveCompFile for file format.

WAVEMAKER = GAUSIAN: initial Gausian hump, need AMP, Xc, Yc, and WID.

WaveCompFile: Wave component file when WAVEMAKER = WAVE_DATA is selected. The format is

nfreq(integer) ndir(integer)
peak_freq (real)
freq1(real)
freq2(real)
...
dir1(real, in degrees)
dir2(real, in degrees)
...
2D array of wave magnitude ((amp(I,J),I=1,ndir),J=1,nfreq)

AMP: amplitude (m) of initial η, if WAVEMAKER = INI_REC, WAVEMAKER = INI_SOL, WAVEMAKER = LEF_SOL.

DEP: water depth at wavemaker location, if WAVEMAKER = INI_SOL, WAVEMAKER = LEF_SOL.

LAGTIME, time lag (s) for the solitary wave generated on the left boundary, e.g., WAVEMAKER = LEF_SOL.

XWAVEMAKER: x (m) coordinate for WAVEMAKER = INI_SOL.

Xc: x (m) coordinate of the center of a rectangular hump if WAVEMAKER = INI_REC.

Yc: y (m) coordinate of the center of a rectangular hump if WAVEMAKER = INI_REC.

WID: width (m) of a rectangular hump if WAVEMAKER = INI_REC, or INI_GAU.

Time_ramp: time ramp (s) for Wei and Kirby (1999) wavemaker.

Delta_WK: width parameter δ for Wei and Kirby (1999) wavemaker. $\delta = 0.3 \sim 0.6$

DEP_WK: water depth (m) for Wei and Kirby (1999) wavemaker.

Xc_WK: x coordinate (m) for Wei and Kirby (1999) wavemaker.

Ywidth_WK: width (m) in y direction for Wei and Kirby (1999) wavemaker.

Tperiod: period (s) of regular wave for Wei and Kirby (1999) wavemaker.

AMP_WK: amplitude (m) of regular wave for Wei and Kirby (1999) wavemaker.

Theta_WK: direction (degrees) of regular wave for Wei and Kirby (1999) wavemaker. Note: it may be adjusted for a periodic boundary case by the program. A warning will be given if adjustment is made.

FreqPeak: peak frequency (1/s) for Wei and Kirby (1999) irregular wavemaker.

FreqMin: low frequency cutoff (1/s) for Wei and Kirby (1999) irregular wavemaker.

FreqMax: high frequency cutoff (1/s) for Wei and Kirby (1999) irregular wavemaker.

Hmo: Hmo (m) for Wei and Kirby (1999) irregular wavemaker.

GammaTMA, TMA parameter γ for Wei and Kirby (1999) irregular wavemaker.

ThetaPeak: peak direction (degrees) for Wei and Kirby (1999) irregular wavemaker.

Sigma_Theta: parameter of directional spectrum for Wei and Kirby (1999) irregular wavemaker.

SPECIFICATION OF PERIODIC BOUNDARY CONDITION

(Note: only south-north periodic condition was implemented, ONLY FOR SERIAL CODE IN THE PRESENT VERSION.)

PERIODIC: logical parameter for periodic boundary condition, T - periodic, F - wall boundary condition.

SPECIFICATION OF SPONGE LAYER

DIRECT_SPONGE: logical parameter for L-D type sponge, T - sponge layer, F - no sponge layer.

FRICTION_SPONGE: logical parameter for friction type sponge, T - sponge layer, F - no sponge layer.

DIFFUSION_SPONGE: logical parameter for diffusion type sponge, T - sponge layer, F - no sponge layer.

Csp: The maximum diffusion coefficient for diffusion type sponge.

CDsponge: The maximum Cd for friction type sponge.

Sponge_west_width: width (m) of sponge layer at west boundary.

Sponge_east_width: width (m) of sponge layer at east boundary.

Sponge_south_width: width (m) of sponge layer at south boundary.

Sponge_north_width width (m) of sponge layer at north boundary

R_sponge: decay rate in L-D type sponge layer. Its values are between $0.85 \sim 0.95$.

A_sponge: maximum damping magnitude in L-D type sponge. The value is ~ 5.0.

SPECIFICATION OF OBSTACLES

OBSTACLE_FILE: name of obstacle file. 1 - water point, 0 - permanent dry point. Data dimension is (Mglob $\times$ Nglob). Data format is the same as the depth data.

SPECIFICATION OF PHYSICS

DISPERSION: logical parameter for inclusion of dispersion terms. T - calculate dispersion, F - no dispersion terms

Gamma1: parameter for linear dispersive terms. 1.0 - inclusion of linear dispersive terms, 0.0 - no linear dispersive terms.

Gamma2: parameter for nonlinear dispersive terms. 1.0 - inclusion of nonlinear dispersive terms, 0.0 - no nonlinear dispersive terms.

Gamma1=1.0, Gamma2=0.0 for NG's equations. Gamma1=1.0, Gamma2=1.0 for the fully nonlinear Boussinesq equations.

Gamma3: parameter for linear shallow water equations (Gamma3 = 1.0). When Gamma3 = 0.0, Gamma1 and Gamma2 automatically become zero.

Beta_ref: parameter β defined for the reference level. β = -0.531 for NG's and FUNWAVE equations.

VISCOSITY_BREAKING : logical parameter for viscous breaking. When this option is selected, Cbrk1 and Cbrk2 needed. Default is shock-capturing type breaking

SWE_ETA_DEP: ratio of height/depth for switching from Boussinesq to NSWE for shock-capturing breaking. The value is ~ 0.80.

SPECIFICATION OF FRICTION

FRICTION_MATRIX: logical parameter for homogeneous and inhomogeneous frction feild. T - inhomogeneous, F - homogeneous

FRICTION_FILE: file file if FRICTION_MATRIX= T , file dimension should be Mglob x Nglob with the first point as the south-west corner. The read format in the code is shown below.

DO J=1,Nglob

READ(1,*)(Cd(I,J),I=1,Mglob)

ENDDO

Cd_fixed: fixed bottom friction coefficient.

SPECIFICATION OF NUMERICS

Time_Scheme: stepping option, Runge_Kutta or Predictor_Corrector (not suggested for this version).

HIGH_ORDER: spatial scheme option, FOURTH for the fourth-order, THIRD for the third-order, and SECOND for the second-order (not suggested for Boussinesq modeling). NOTE: Abadie et al. (2012) pointed out that the fourth-order TVD scheme in RUNWAVE-TVD has an stability problem for simulations in deep water. We also encountered the similar problem in some cases. For this reason, the third-order scheme is suggested for the present version.

SM Abadie, JC Harris, ST Grilli, R Fabre, 2012, Numerical modeling of tsunami waves generated by the flank collapse of the Cumbre Vieja Volcano (La Palma, Canary Islands): Tsunami source and near field effects, Journal of Geophysical Research: Oceans (19782012) 117 (C5)

CONSTRUCTION: construction method, HLL for HLL scheme, otherwise for averaging scheme.

CFL: CFL number, CFL ~ 0.5.

FroudeCap: cap for Froude number in velocity calculation for efficiency. The value could be $5 \sim 10.0$.

MinDepth: minimum water depth (m) for wetting and drying scheme. Suggestion: MinDepth = 0.001 for lab scale and 0.01 for field scale.

MinDepthFrc: minimum water depth (m) to limit bottom friction value. Suggestion: MinDepthFrc = 0.01 for lab scale and 0.1 for field scale.

SHOW_BREAKING: logical parameter to calculate breaking index. Note that, if VISCOSITY_BREAKING is not selected, breaking is calculated using shock wave capturing scheme. The index calculated here is based on Kennedy et al. (2000).

Cbrk1: parameter C1 in Kennedy et al. (2000).

Cbrk2: parameter C2 in Kennedy et al. (2000).

WAVEMAKER_Cbrk: breaking parameter inside wavemaker. For some cases, wave breaks inside the wavemaker. This parameter provides Cbrk inside the wavemaker domain. For most of cases, set WAVEMAKER_Cbrk = Cbrk1 or higher.

STEADY_TIME: starting time (t_1 in (105)) for calculating mean values, significant/RMS wave height (when WaveHeight = T, output parameter below).

T_INTV_mean: time interval ($t_2 - t_1$ in (105)) for calculating mean values, significant/RMS wave height (when WaveHeight = T, output parameter below).

SPECIFICATION OF OUTPUT VARIABLES

NumberStations: number of station for output. If NumberStations > 0, need input i,j in STATION_FILE

DEPTH_OUT: logical parameter for output depth. T or F.

U: logical parameter for output u. T or F.

V: logical parameter for output v. T or F.

ETA: logical parameter for output η. T or F.

MASK: logical parameter for output wetting-drying MASK. T or F.

MASK9: logical parameter for output MASK9 (switch for Boussinesq/NSWE). T or F.

SourceX: logical parameter for output source terms in x direction. T or F.

SourceY: logical parameter for output source terms in y direction. T or F.

P: logical parameter for output of momentum flux in x direction. T or F.

Q: logical parameter for output of momentum flux in y direction. T or F.

Fx: logical parameter for output of numerical flux F in x direction. T or F.

Fy: logical parameter for output of numerical flux F in y direction. T or F.

Gx: logical parameter for output of numerical flux G in x direction. T or F.

Gy: logical parameter for output of numerical flux G in y direction. T or F.

AGE: logical parameter for output of breaking age. T or F.

HMAX: logical parameter for output of recorded maximum surface elevation . T or F.

HMIN: logical parameter for output of recorded minimum surface elevation . T or F.

UMAX: logical parameter for output of recorded maximum velocity . T or F.

VORMAX: logical parameter for output of recorded maximum vorticity . T or F.

MFMAX: logical parameter for output of recorded maximum momentum flux . T or F.

WaveHeight: logical parameter for output of wave height, Hsig, Hrms, Havg. T or F.

4.4 Input for the spherical code

All input parameters, except the following grid information, are the same as for the Cartesican code.

Lon_West: longitude (degrees) of west boundary.

Lat_South: latitude (degrees) of south boundary.

Dphi: $d\phi$ (degrees)

Dtheta: $d\theta$ (degrees)

In addition, it is not necessary to specify Gamma2 (for nonlinear dispersive terms) in the spherical code.

Another feature of the spherical code is that a computational grid can be a stretched grid. For a stretched grid, a user should set StretchGrid = T and provide grid files for DX and DY and a file for Coriolis parameters at each grid point. For example,

DX_FILE = dx_str.txt
DY_FILE = dy_str.txt
CORIOLIS_FILE = cori_str.txt

However, use of a stretched grid is not recommended in terms of decrease in numerical accuracy for higher order numerical schemes.

4.5 Model nesting

The present version has a capability for one-way nesting. The nesting scheme passes surface elevation and velocity components calculated from a large domain to a nested small domain through ghost cells at nesting boundaries. To run a nested model, the following procedures should be performed.

1. The coupling option in Makefile should be defined as '-Dcoupling". The program should be re-compiled.

2. Prepare nesting data using the output of a large-domain model. The following is an example of the data format.

   coupling data
   boundary info: num of points (negative means no point), start point
   EAST
   -1 1 (*no data on east side*)
   WEST
   5 1 (*5 points at west boundary, start from 1*)
   SOUTH
   -1 0 (*no data on south side*)
   NORTH
   -1 0 (*no data on north side*)
   TIME SERIES
   68.443001 (*first step, NOTE: model clock will be initialized as this time*)
   EAST SIDE (*no nesting on east side*)
   WEST SIDE
   0.141220E-02 0.141220E-02 0.141220E-02 0.141220E-02 0.141220E-02 (u)
   -0.119260E-10 -0.119260E-10 -0.667390E-10 -0.667390E-10 -0.219240E-10 (v)
   0.141890E-02 0.141890E-02 0.141890E-02 0.141890E-02 0.141890E-02 (z)
   SOUTH SIDE (*no nesting on south side*)
   NORTH SIDE (*no nesting on north side*)
   68.641998 (*next time step*)
   ...

   The example above is a case that a model nesting takes place at the WEST (left) boundary of a small domain. Boundaries are defined with the order: EAST, WEST, SOUTH, and NORTH. If the num of points of a boundary is larger than zero, the program will read a time series of (u, v, η) below 'XXXX SIDE'. The read format is

   ```
   READ(11,*) TIME_COUPLING_2
   READ(11,119)(U_COUPLING_EAST(I,2),I=1,N_COUPLING_EAST)
   READ(11,119)(V_COUPLING_EAST(I,2),I=1,N_COUPLING_EAST)
   READ(11,119)(Z_COUPLING_EAST(I,2),I=1,N_COUPLING_EAST)
   ```

ENDDO
119 FORMAT(5E16.6)

where N_COUPLING_EAST is Num of points at the EAST boundary.

3. Specify the file of coupling data in input.txt

   ! ———————— COUPLING ————————————-
   ! if do coupling, have to set -DCOUPLING in Makefile
   COUPLING_FILE = coupling.txt
   where 'coupling.txt' is the file saved in procedure 2.

4.6 Output

The output files are saved in the result directory defined by RESULT_FOLDER in input.txt. For outputs in ASCII, a file name is a combination of variable name and an output series number such eta_00001, eta_00002, ... The format and read/write algorithm are consistent with a depth file. Output for stations is a series of numbered files such as sta_00001, sta_00002 ....

Other output formats are under development.

5 Examples

The model has been validated extensively using laboratory experiments for wave shoaling and breaking as in the FUNWAVE manual by Kirby et al. (1998). In addition, Tehranirad et al. (2011) used FUNWAVE-TVD Version 1.0 to carry out tsunami benchmark testing in conjunction with the National Tsunami Hazard Mitigation Program. Tests of mass conservation and convergence are included in Tehranirad et al. (2011).

5.1 Breaking waves on a beach

Hansen and Svendsen (1979) carried out laboratory experiments of wave shoaling and breaking on a beach. Waves were generated on a flat bottom a 0.36 m depth, and the beach slope was 1:34.26. The experiments included several cases including plunging breakers, plunging-spilling breakers and spilling breakers. In this manual, we simulate the plunging breaker case as a demonstration. The wave height and wave period are 4.3 cm and 3.33 s, respectively.

We adopted a grid size, dx = 0.025 m in the present case. Shi et al. (2011) also showed simulations in different grid sizes and verified a convergence with grid refinement. To compare two different breaking algorithms we conducted runs with the shock-capturing breaking method and the eddy-viscosity breaking method, respectively. Figure 3 shows comparisons of wave height and wave setup between measured data and numerical results from model runs with different breaking algorithms. The wave breaking location of wave setup/setdown predicted by the two runs are in agreement with the data. However, the predicted maximum wave heights are slightly different. Basically, the two models underpredict the peak wave height at breaking and overpredict wave

height inside of the surfzone. For the eddy-viscosity type breaking, we adopted Cbrk1 =0.65 and Cbrk2=0.15, which are default values from Kennedy et al and this model. The modeled mean water level is slightly lower than the measurement at the toe of the beach (x=0 m) due to the mass conservation in the limited length of computational domain.

Some necessary definitions in the input file, input.txt, for the plunging breaker case are

```
    ! ———————DEPTH———————————————-
DEPTH _TYPE = SLOPE
DEPTH _FLAT = 0.36
SLP = 0.0292
Xslp = 30.0

    ! ———————-PRINT———————————
RESULT_FOLDER = your output folder

    ! ———————DIMENSION———————————
Mglob = 2000
Nglob = 3
! ———————— TIME———————————————-
DX = 0.025
DY = 0.2 ! give any larger value than DX for 1-D case
! ———————-WAVEMAKER———————————
WAVEMAKER = WK _REG
Xc _WK = 45.0
Tperiod = 3.33
AMP _WK = 0.018 ! this value is smaller than the report of Hansen and Svendsen, but matches the measured data
DEP _WK = 0.36
Theta _WK = 0.0
Delta _WK = 0.5
! ———————- SPONGE LAYER ———————————
DIRECT _SPONGE = T
Sponge _west _width = 12.0
Sponge _east _width = 0.0
Sponge _south _width = 0.0
Sponge _north _width = 0.0
R _sponge = 0.85
A _sponge = 5.0
! ———————-PHYSICS———————————
DISPERSION = T
Gamma1 = 1.0
```

```
Gamma2 = 1.0
Gamma3=1.0
Beta _ref=-0.531
!————————-breaking————————————
SWE _ETA _DEP = 0.8
VISCOSITY_BREAKING = F ! switch to T for the other case
Cbrk1 = 0.65
Cbrk2 = 0.15
WAVEMAKER_Cbrk = 0.65

    !————————-Friction————————————
Friction_Matrix=F
Cd_file= cd.txt
Cd = 0.0 ! Cd is not sensitive for this case
! ————————-NUMERICS————————————-
Time _Scheme = Runge _Kutta
HIGH _ORDER = THIRD
CONSTRUCTION = HLLC
CFL = 0.5
! ————————WET-DRY————————————-
MinDepth=0.001
MinDepthFrc = 0.001
! ————————OUTPUT————————————
NumberStations = 119
STATIONS_FILE = gauge.txt
DEPTH _OUT = T
ETA = T
MASK = T
```

5.2 Random wave shoaling and breaking on a slope (in directory /mase_kirby_1d/)

To study random-wave properties of shoaling and breaking, Mase and Kirby (1992) conducted a laboratory experiment of random wave propagation over a planar beach. The experiment layout is shown in Figure 4, where a constant depth of 0.47 m on the left connects to a constant slope of 1:20 on the right. Two sets of random waves with peak frequencies of 0.6 Hz (run 1) and 1.0 Hz (run 2) were generated by the wavemaker on the left. The target incident spectrum was a Pierson-Moskowitz spectrum. Wave gauges at depths h = 47, 35, 30, 25, 20, 17.5, 15, 12.5, 10, 7.5, 5, and 2.5 cm collected time series of surface elevation.

Wei and Kirby (1995) carried out a simulation of run 2 without wave breaking. Later, Kirby et al. (1998) and Kennedy et al. (2000) carried out the same simulation with wave breaking. The

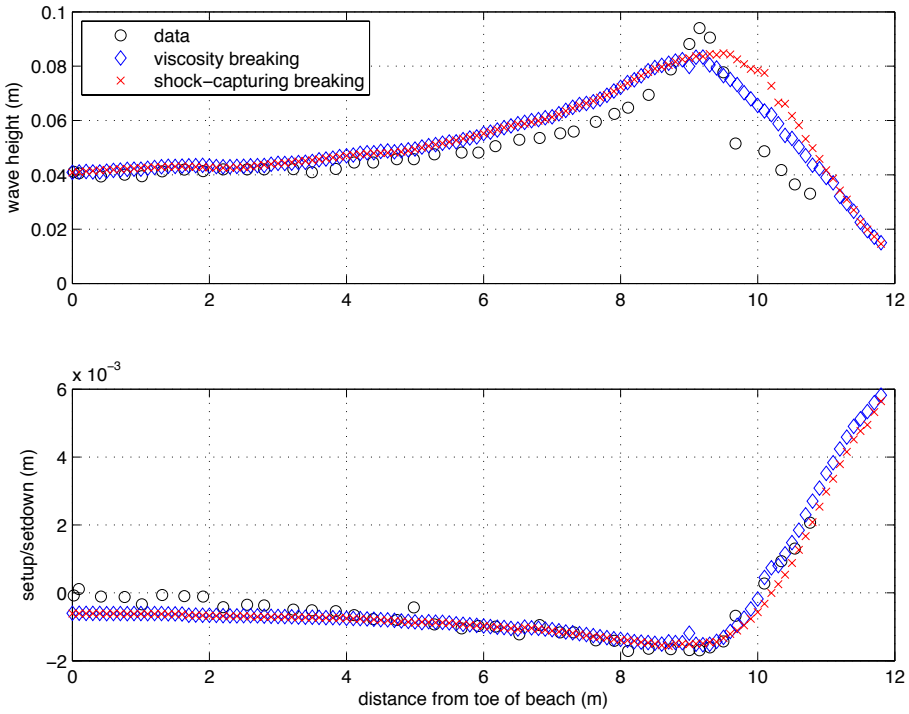

Figure 3: Comparisons of wave height (upper panel) and wave setup (lower panel) between measured data and model results.

present model was set up following Kirby et al. (1998), who used an internal wavemaker located at the toe of the slope where surface elevation is measured (gauge 1).

A *FFT* was used to transform between the time domain and frequency domain data required by the wavemaker. A MATLAB script, (*fft4wavemaker.m*) to perform this transform is included in the example. The script reads the measured data from the file called *r2d470.dat*, and saves calculated wave amplitude, period and phase information for each component in the file named as *wavemk_per_amp_pha.txt*. The low and high-frequency cutoffs are 0.2 and 10.0 Hz, respectively.

The simulation time is the same as the time length of data collection. The computational domain is from x = 0 m to 20 m with a grid size of 0.04 m. The toe of the slope starts at x = 10 m. A sponge layer is specified at the left side boundary, to absorb reflected waves, but no sponge layer is needed on the right boundary, which differs from Kirby et al. (1998) who used the slot method combined with a sponge layer at the end of the domain.

We present the model results for run 2 and compare with the experimental data measured at the other 11 gauges shown in Figure 4. Figure 5 shows model results (dashed lines) and measured data (solid lines) from t = 20 s to t = 40 s at those gauges. Both model and data show that most waves start breaking at the depth h= 15 cm. Except for small discrepancies for wave phases, the model reproduces the measured waveform quite well.

To further demonstrate the applicability of the model, we performed third moment computations of the resulting time series of surface elevation. Normalized wave skewness and asymmetry were calculated for both measured and modeled time series of surface elevation according to the following formulations,

$$\begin{aligned} \text{skew} &= \frac{<\eta^3>}{<\eta^2>^{3/2}} \\ \text{asym} &= \frac{<H(\eta)^3>}{<\eta^2>^{3/2}} \end{aligned} \tag{106}$$

where H denotes the Hilbert transform, $<>$ is the mean operator, and the mean has been removed from the time series of surface elevation.

Figure 6 shows the comparisons of skewness and asymmetry between the model results and experiment data. The model predicted skewness and asymmetry reasonably well with a slightly overprediction of wave skewness inside the surf zone.

It is worth mentioning that Kirby et al. (1998) employed more frequent use of numerical filtering, especially after wave breaking, so that the model run was stable over the entire data time series. The present model did not encounter any stability problem without filtering.

Some necessary definitions in the input file, input.txt, are

```
! ——————————DEPTH———————————————————-
DEPTH_TYPE = SLOPE
DEPTH_FLAT = 0.47
SLP = 0.05
```

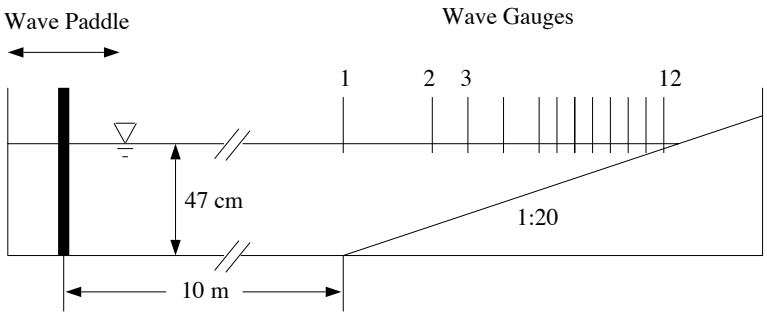

Figure 4: Experiment layout of Mase and Kirby (1992).

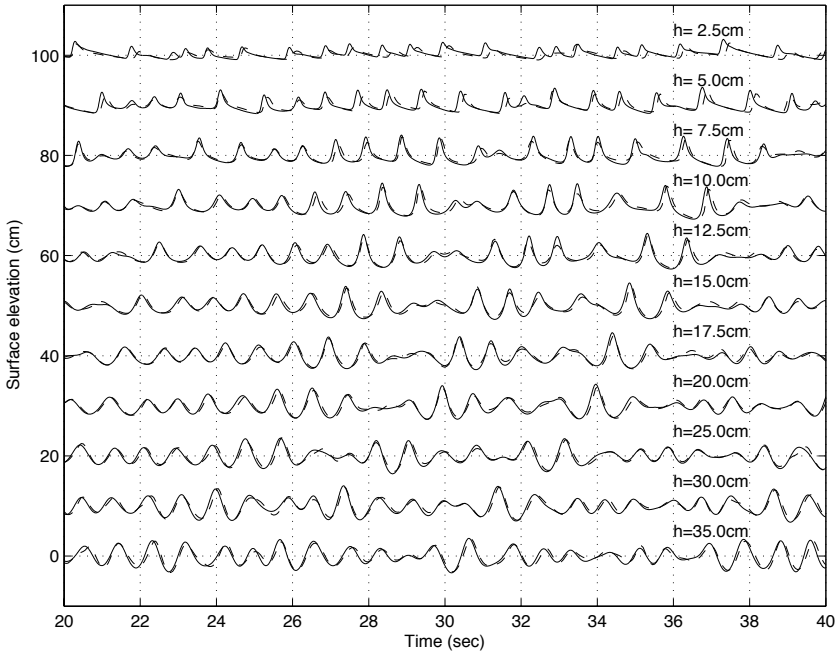

Figure 5: Time series comparison of η between model (dashed lines) and data (solid lines) at 11 wave gauges in Mase and Kirby (1992).

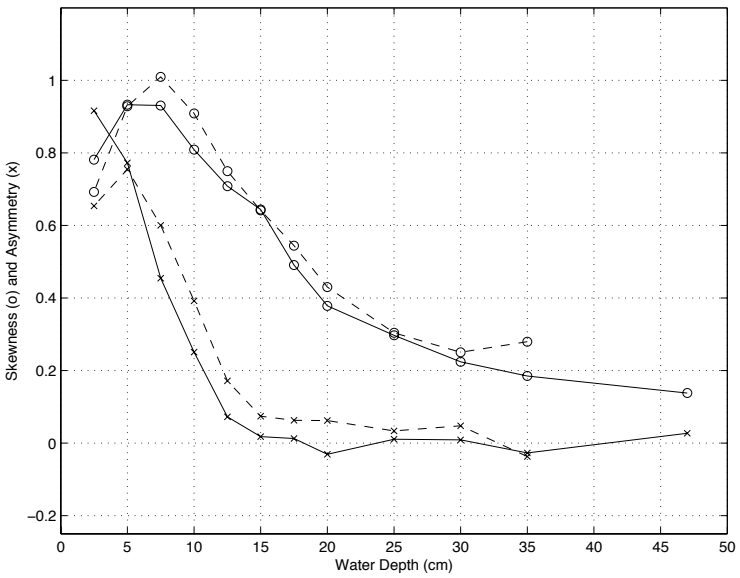

Figure 6: Comparison of skewness ($\circ$) and asymmetry ($\times$) at different water depths. Solid lines are experiment data (Mase and Kirby, 1992). Dashed lines are numerical results

```
Xslp = 10.0

    ! ——————————DIMENSION————————————————
Mglob = 500
Nglob = 3

    ! ——————————— TIME————————————————————-
TOTAL_TIME = 716.0
PLOT_INTV = 10.0
PLOT_INTV_STATION = 0.05
SCREEN_INTV = 1.0

    ! ——————————GRID————————————————————-
DX = 0.04
DY = 0.10

    ! ——————————-WAVEMAKER———————————————
WAVEMAKER = WK_TIME_SERIES
NumWaveComp = 1505
PeakPeriod = 1.0
WaveCompFile = ../fft/wavemk_per_amp_pha.txt
! Wei and Kirby 1999
Time_ramp = 1.0
Delta_WK = 0.4 ! width parameter 0.3-0.6
DEP_WK = 0.47
Xc_WK = 10.0
Ywidth_WK = 10000.0 ! give any bigger value than the width of tank

    ! ——————————- SPONGE LAYER ——————————
DIRECT _SPONGE = T
Sponge_west_width = 2.0
Sponge_east_width = 0.0
Sponge_south_width = 0.0
Sponge_north_width = 0.0
R_sponge = 0.90
A_sponge = 5.0

    ! ——————————-PHYSICS———————————————
DISPERSION = T
Gamma1 = 1.0
Gamma2 = 1.0
```

```
Gamma3 = 1.0
Beta_ref=-0.531
SWE_ETA_DEP = 0.80
Cd = 0.001 ! not sensitive for this case

    ! ————————-NUMERICS————————————-
Time_Scheme = Runge_Kutta
HIGH_ORDER = THRID
CONSTRUCTION = HLLC
CFL = 0.5
! ——————WET-DRY————————————-
MinDepth=0.001
MinDepthFrc = 0.001
! ————————OUTPUT————————————
NumberStations = 12
STATIONS_FILE = gauges_004.txt
DEPTH_OUT = T
ETA = T
```

Model output at 12 gauges are saved in *sta_0001, sta_0002, ..., sta_0012*. Use MATLAB scripts, *comp_mkskew.m* and *cmop_series.m* for plots.

5.3 Wave propagation over a shoal: Berkhoff et al. (1982) (in directory /berkhoff_2d/)

The laboratory experiment of wave propagation over a shoal conducted by Berkhoff et al. (1982) has served as a standard test for examining numerical model performances in predicting wave shoaling, refraction, diffraction and nonlinear dispersion. Kirby et al. (1998) showed that the previous version of FUNWAVE accurately reproduces measured wave heights in the experiments. Here, in this manual, we repeat this test exactly following Kirby et al. (1998).

The bottom topography is shown in Figure 7, which is generated using the same program in Kirby et al., (1998). The topography consists of an elliptic shoal resting on a plane beach with a constant slope 1/50. Bottom contours on the slope are oriented at an angle of 20° to the y axis. Regular waves with period of $1s$ and amplitude of $2.32cm$ are generated by a wavemaker at $x = -10m$ and propagate across the domain. Experiment data are collected along 8 transects as shown in the figure. Two vertical side walls are located at $y = -10m$ and $y = 10m$. Detailed information on the geometry may be obtained in Berkhoff et al. (1982) or Kirby and Dalrymple (1984).

The computational domain used in the model is the same as in Figure 7 except for two sponge layers with a width of $2m$ sitting behind wavemaker and on the end of the beach. The source function for generating the corresponding monochromatic wave is located at the wavemaker.

Thirty waves are simulated in order to get a quasi-stable wave condition. The time series of

surface elevation in the last 5 seconds (5 wave periods) are used for the wave height estimation at each grid point. Matlab post-processing scripts, *calcheight.m* and *showheight.m* are provided in the example. Figure 8 shows comparisons between model results and experimental data along the eight transects where measurements were made. The model results of wave height agree well with experimental data, in both sections parallel or normal to incident wave direction. The results from the present model are also similar to that from the previous version of FUNWAVE in Kirby et al. (1998).

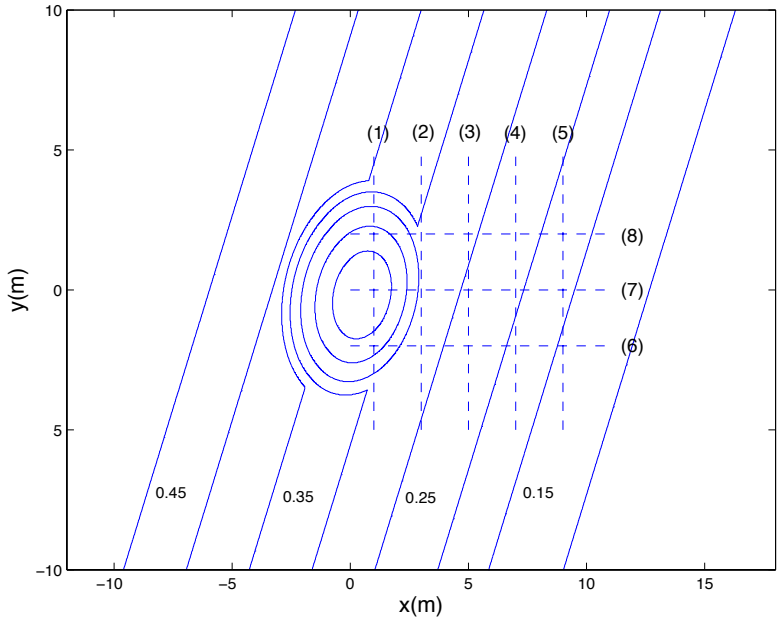

Figure 7: Experiment layout for wave focusing experiment of Berkhoff et al. (1982).

Parameters and other definitions specified in *input.txt* are listed below.

```
! ———————DEPTH————————————————-
DEPTH_TYPE = DATA
DEPTH_FILE = ../input/depth.txt

! ———————DIMENSION————————————
Mglob = 600
Nglob = 200

! ——————— TIME————————————————-
```

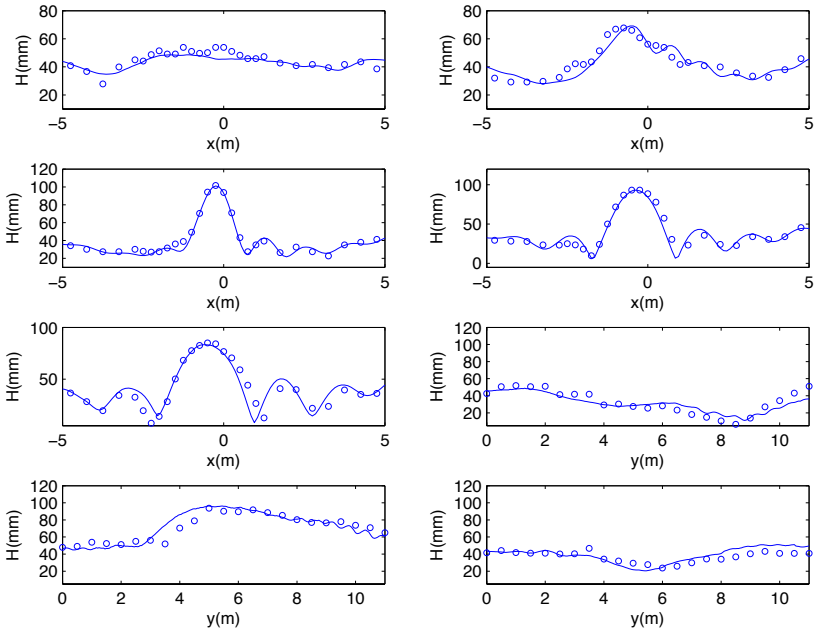

Figure 8: Comparisons of wave height along specified sections between the model (solid lines) and experiment data (circles).

```
TOTAL_TIME = 30.0
PLOT_INTV = 0.02
PLOT_INTV_STATION = 0.02
SCREEN_INTV = 0.1

    ! ————————GRID————————————-
DX = 0.05
DY = 0.10

    ! ————————-WAVEMAKER———————————
WAVEMAKER = WK_REG
Time_ramp = 1.0
Delta_WK = 0.5 ! width parameter 0.3-0.6
DEP_WK = 0.45
Xc_WK = 3.0
Ywidth_WK = 10000.0 ! give any larger value than the width of flume
Tperiod = 1.0
AMP_WK = 0.0232
Theta_WK = 0.0

    ! ————————- SPONGE LAYER ——————————
DIRECT _SPONGE = T
Sponge_west_width = 2.0
Sponge_east_width = 2.0
Sponge_south_width = 0.0
Sponge_north_width = 0.0
R_sponge = 0.90
A_sponge = 5.0

    ! ————————-PHYSICS———————————
DISPERSION = T
Gamma1 = 1.0
Gamma2 = 1.0
Gamma3 = 1.0
Beta_ref=-0.531
SWE_ETA_DEP = 0.80
Cd = 0.0

    ! ————————-NUMERICS———————————-
Time_Scheme = Runge_Kutta
HIGH_ORDER = FOURTH
```

```
CONSTRUCTION = HLLC
CFL = 0.5

    ! ————————WET-DRY————————————————-
MinDepth=0.001
MinDepthFrc = 0.001

    ! ———————————OUTPUT———————————————
NumberStations = 0
DEPTH_OUT = T
ETA = T
```

5.4 Wave refraction-diffraction over a mound (Vincent and Briggs, 1989, provided by Choi and Seo)

Vincent and Briggs (1989) performed experiments on monochromatic waves and frequency-direction spreading random waves which pass over a submerged elliptic shoal by using a directional spectral wave generator (DSWG) in Coastal and Hydraulics Laboratory's wave basin of the U.S. Army Engineer Research and Development Center.

Figure 9 demonstrates the computational domain based on the experiment layout of Vincent and Briggs (1989). Transect 4 represents one of the measurement transects where data are used to compare the model. FUNWAVE-TVD is set up for four cases, the non-breaking wave case, the breaking cases with the eddy-viscosity breaker and the shock-capturing breaker, and the irregular wave case. Here, we only present the non-breaking case and the breaking case with the eddy-viscosity breaker. The other cases can be found in the directories named by corresponding tests in the example package.

The grid resolution for this case is $dx = dy = 0.05$ m, resulting in a model dimension of 760 x 500. The total computational time is 120.0 sec. The wavemaker is placed at $x = 10$ m. Two sponge layers are used on both left and right sides with widths of 7.5 m and 5.0 m, respectively.

Figure 10 and 11 show model/data comparisons at Transect 4 for the non-breaking case and the breaking case, respectively. Parameters and other definitions for the breaking wave case are specified in input.txt, listed below.

```
    ! ———————————DEPTH————————————————————-
DEPTH_TYPE = DATA
DEPTH_FILE = ../depth/dep_ny501mx760.dat

    ! ———————————DIMENSION———————————————
Mglob = 760
Nglob = 501
```

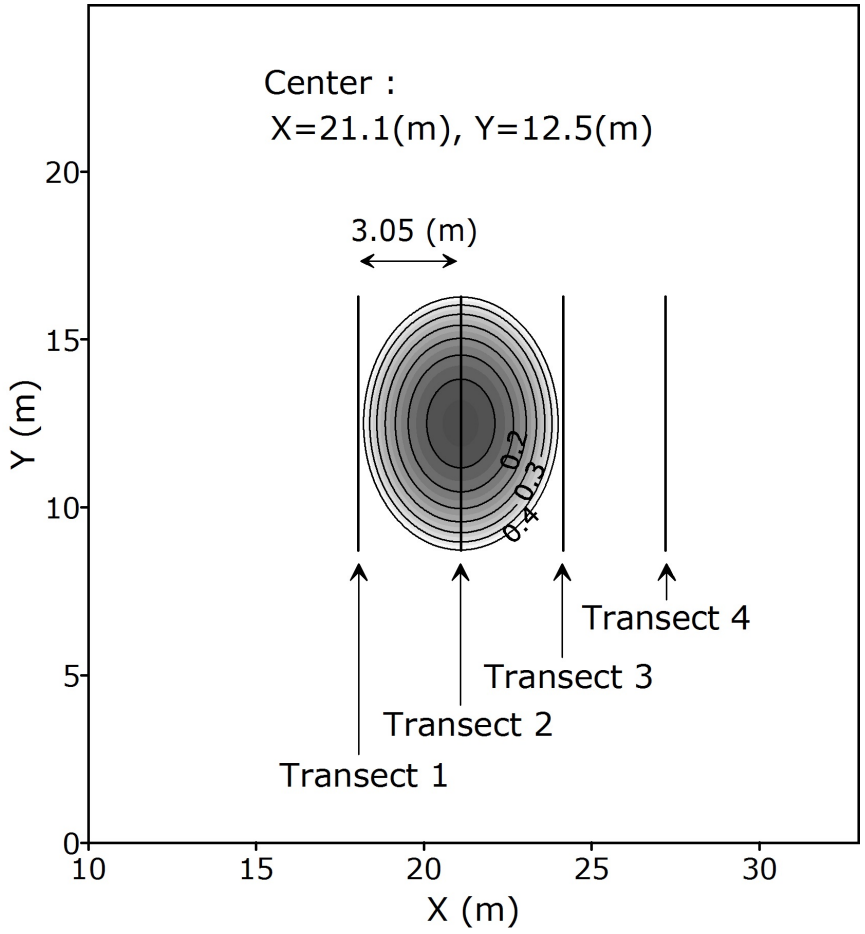

Figure 9: Computational domain for the case of Vincent and Briggs (1989).

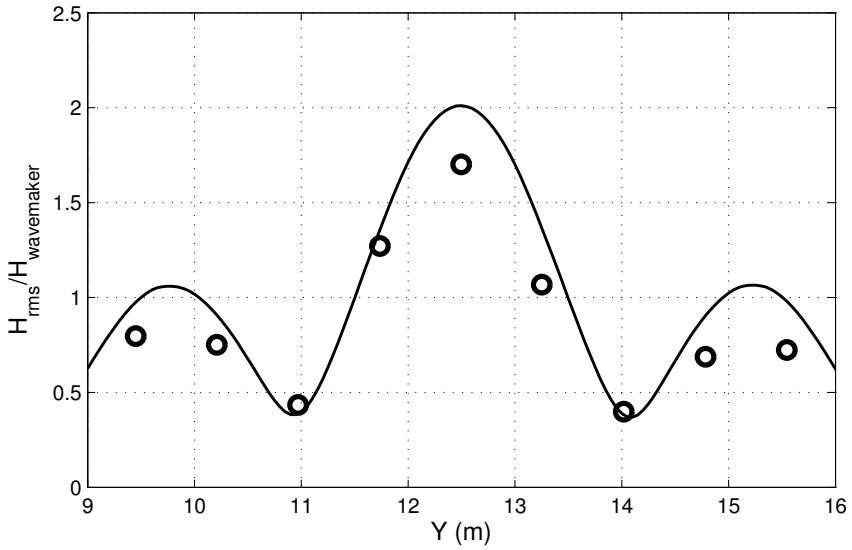

Figure 10: Comparison between data (circles) and model (solid line) in the non-breaking case of Vincent and Briggs (1989).

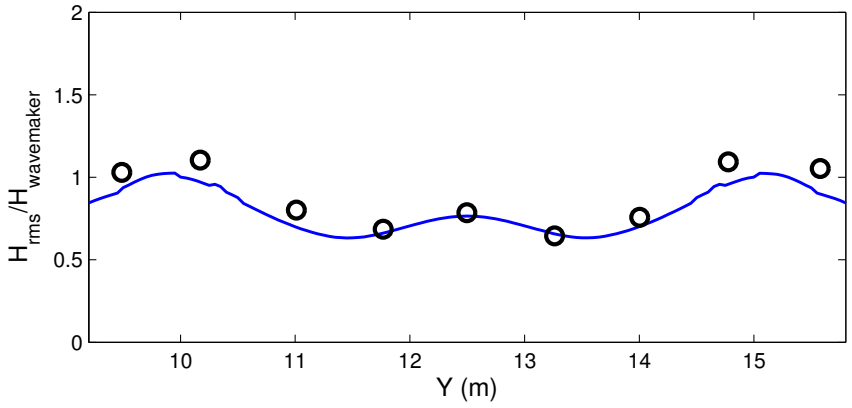

Figure 11: Comparison between data (circles) and model (solid line) in the breaking case of Vincent and Briggs (1989).

```
    ! ————————— TIME————————————————-
TOTAL_TIME = 120.0
PLOT_INTV = 0.05

    ! ————————GRID————————————————-
DX = 0.05
DY = 0.05

    ! ————————-WAVEMAKER———————————
WAVEMAKER = WK_REG
Delta_WK = 2.0 !(large number to avoid breaking inside wavemaker)
Xc_WK = 10.0
Yc_WK = 12.5
Ywidth_WK = 20000.0
Tperiod = 1.3
AMP_WK = 0.0675
Theta_WK = 0.0

    ! ————————- SPONGE LAYER ——————————
FRICTION_SPONGE = T
DIRECT_SPONGE = T
CDsponge = 1.0
Sponge_west_width = 7.5
Sponge_east_width = 5.0
R_sponge = 0.85
A_sponge = 5.0

    ! ————————-PHYSICS———————————————
DISPERSION = T
Gamma1 = 1.0
Gamma2 = 1.0
Gamma3 = 1.0
Beta_ref=-0.531
VISCOSITY_BREAKING = T

    !————————-Friction———————————————
Friction_Matrix=F
Cd = 0.0

    ! ————————-NUMERICS————————————————-
Time_Scheme = Runge_Kutta
```

HIGH_ORDER = FOURTH
CFL = 0.2

! ————————OUTPUT————————————
NumberStations = 0
ETA = T

Cases for Vincent and Briggs' experiments are included in the directory /vincent_briggs in the example package. The sub-directories are

/01_eddy_breaking : breaking case with eddy-viscosity breaker
/02_shock_breaking: breaking case with shock-capturing breaker
/03_nonbreaking: non-breaking case
/04_irregular: irregular wave case
/depth: depth files
/postprocessing: matlab scripts for post-processing

In addition, a readme file, README.txt, is included in the directory to describe how to post-process results and plot figures.

5.5 Periodic wave over submerged bar (Luth et al., 1994, provided by Choi and Seo)

It is well known that regular waves decompose into higher frequency free waves as they propagate past a submerged bar, as shown in experimental work by Beji and Battjes (1993), Luth et al. (1994), and Ohyama et al. (1994). Comparisons between several weakly nonlinear Boussinesq-type models and experimental data by Beji and Battjes (1993) and Luth et al. (1994) of waves propagating over a submerged bar were presented by Dingemans (1994).

The experiments performed by Beji and Battjes (1993) and Luth et al. (1994) have the same geometric characteristics, except for the length scale (Luth et al., 1994 is twice as large as in Beji and Battjes, 1993). In Luth et al. (1994) all gauge locations used in Beji and Battjes (1993) were repeated, and another run of measurements was performed with the gauges at different locations. In this section, we compare the present model with two laboratory experimental data sets, Case A and C, by Luth et al. (1994).

The layout of the experimental set-up with the locations of the measurement stations (to which we refer by their location, e.g., gauge 2.0 m, gauge 15.7 m, etc.) and the geometry of the flume are illustrated in Figure 12.

The model is set up in a 54 m long flume based on the experiment layout but extended on both sides in order to include sponge layers. The wavemaker is specified at x = 10.0 m and wave gauges are located at x = 20.0 m + (2.0 m, 4.0 m, ..., 23.0 m), respectively. The grid size $dx = dy = 0.02$ m.

Figure 13 and 14 show model/data comparisons of Case A and Case C, respectively. Parameters and other definitions specified for Case A are listed below.

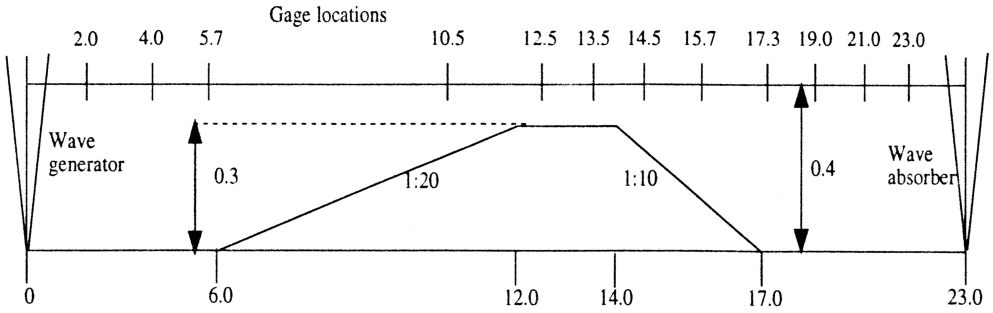

Figure 12: Sketch of wave flume of Delft experiments. All dimensions in (m).

! ——————DEPTH————————————————-
DEPTH_TYPE = DATA
DEPTH_FILE = ../depth/dep_ny6mx2700.dat

! ——————DIMENSION————————————
Mglob = 2700
Nglob = 3

! ——————GRID————————————————-
DX = 0.02
DY = 0.02

! ——————-WAVEMAKER————————————
WAVEMAKER = WK_REG
DEP_WK = 0.4
Xc_WK = 10.0
Yc_WK = 0.0
Ywidth_WK = 20000.0 (any number larger than the flume width)
Tperiod = 2.02
AMP_WK = 0.01
Theta_WK = 0.0

! ——————- SPONGE LAYER ——————————
FRICTION_SPONGE = T
DIRECT_SPONGE = T
CDsponge = 1.0
Sponge_west_width = 8.0

Sponge_east_width = 8.0
Sponge_south_width = 0.0
Sponge_north_width = 0.0
R_sponge = 0.85
A_sponge = 5.0

! ————————-PHYSICS————————————
DISPERSION = T
Gamma1 = 1.0
Gamma2 = 1.0
Gamma3 = 1.0

!————————-Friction————————————
Friction_Matrix=F
Cd = 0.0

! ————————-NUMERICS————————————-
HIGH_ORDER = FOURTH CFL = 0.2

! ————————OUTPUT————————————
NumberStations = 10
STATIONS_FILE = gauges.txt
ETA = T

The case is included in the example package directory /car_luth_AC/. Model setups for a finer grid resolution ($dx = 0.01m$ are also provided in the example package:

/01_CaseA : Case A with $dx = 0.02$ m

/01_CaseA_finer: Case A with $dx = 0.01$ m

/01_CaseC : Case C with $dx = 0.02$ m

/01_CaseC_finer: Case C with $dx = 0.01$ m

/depth: depth files

/lab_data: lab data

/postprocessing: matlab scrips to plot model/data comparisons such as plot_compare_A.m, plot_compare_C.m, plot_compare_A_finer.m and plot_compare_C_finer.m

5.6 Solitary wave on a conical island

Laboratory experiments on the interaction between solitary waves and a conical island were conducted by Briggs et al (1995). The three cases from this test illustrate the important fact that runup and inundation heights on the sheltered back sides of an island can exceed the incident wave height

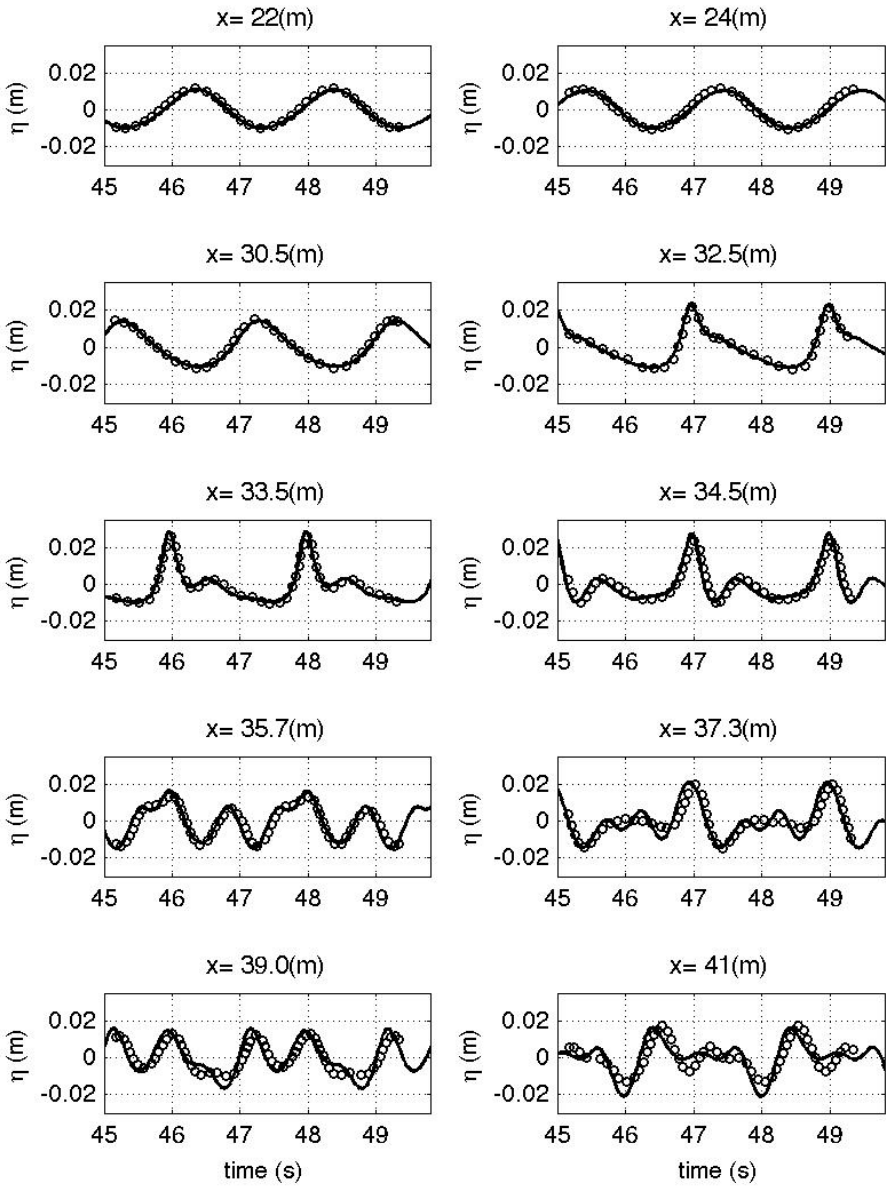

Figure 13: Comparison of model (solid lines) and data (circles) at measurement gauges (Note that the reference of x shifts 20m vs. experiment). Luth et al. Case A.

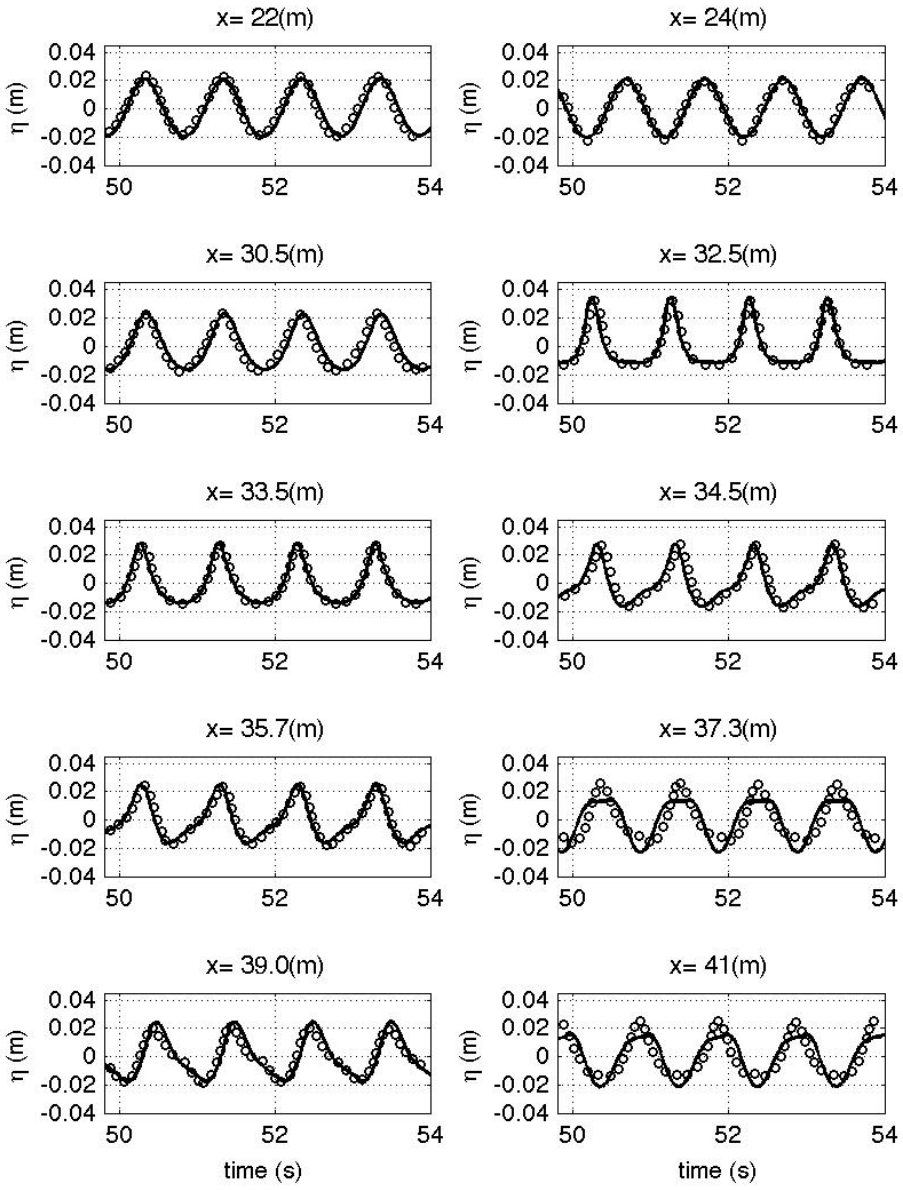

Figure 14: Comparison of model (solid lines) and data (circles) at measurement gauges (Note that the reference of x shifts 20m vs. experiment). Luth et al. Case C.

on the exposed front side, due to trapping of wave fronts propagating around the island circumference. These tests have been used in a number of validation studies for a variety of models, including nonlinear shallow water equations (Liu et al 1995) and Boussinesq equations (Chen et al, 2000). The benchmark test is specified in Section 3.3 of Appendix A of Synolakis et al (2007).

Large-scale laboratory experiments were performed at Coastal Engineering Research Center, Vicksburg, Mississippi, in a 30m-wide, 25m-long, and 60cm-deep wave basin (Figure 15). In the physical model, a 62.5cm-high, 7.2m toe-diameter, and 2.2m crest-diameter circular island with a 1:4 slope was located in the basin (Figure 16). Experiments were conducted at depth of 32cm, with three different solitary waves (H/d=0.045, 0.091, 0.181). Water-surface time histories were measured with 27 wave gages located around the perimeter of the island (Figure 17).

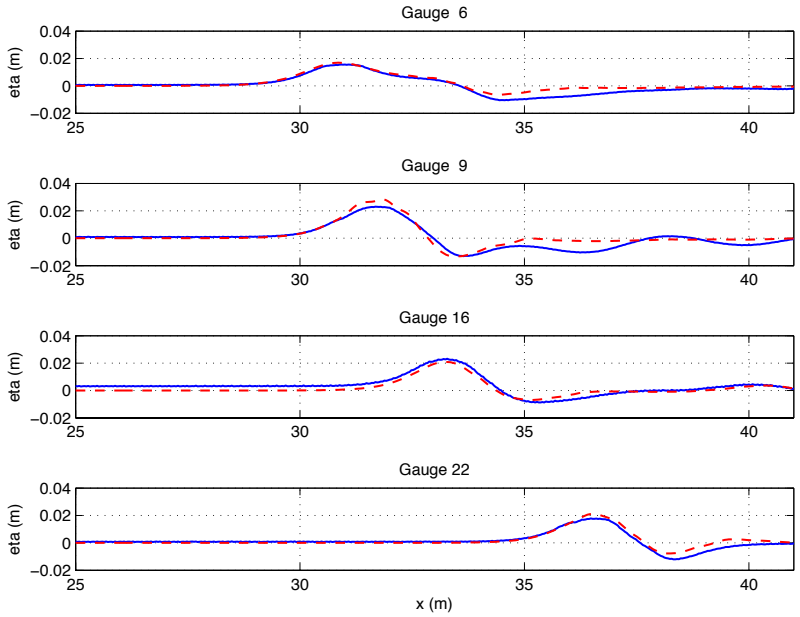

Figure 15: View of conical island(top) and basin(bottom)(from Synolakis et al (2007, Figure A16)).

For this benchmark test, time histories of the surface elevation around the circular island are given at four locations, i.e., in the front of the island at the toe (Gauge 6) and gauges closest to the shoreline with the numbers 9, 16, and 22 located at the 0°, 90°, and 180° radial lines (Figure 17). A grid size of $\Delta x = 0.05m$ is considered for proper numerical simulation of this benchmark. Figures 18-20 shows the comparison between the laboratory data with numerical calculations. Table 1 represents the error of the maximum runup for each gauge for different wave heights.

Input parameters in the tests are listed as below.

! ——————DEPTH————————————————-
DEPTH_TYPE = DATA

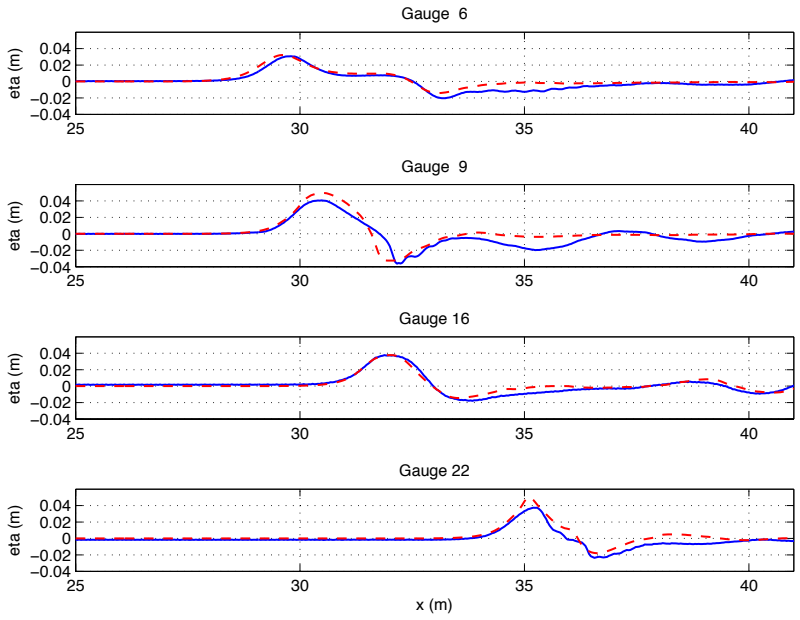

Figure 16: Definition sketch for conical island. All dimensions are in cm (from Synolakis et al (2007, Figure A17)).

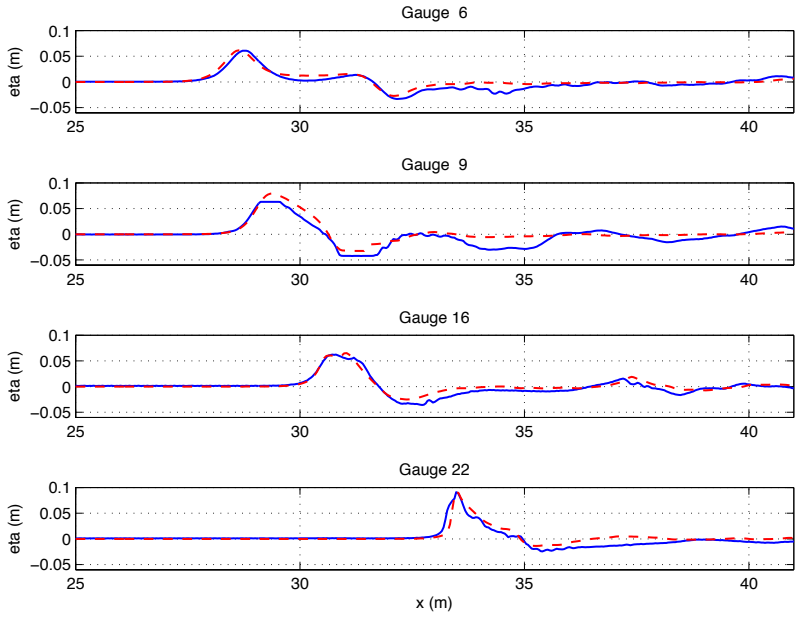

Figure 17: Schematic gauge locations around the conical island(from Synolakis et al (2007, Figure A18)).

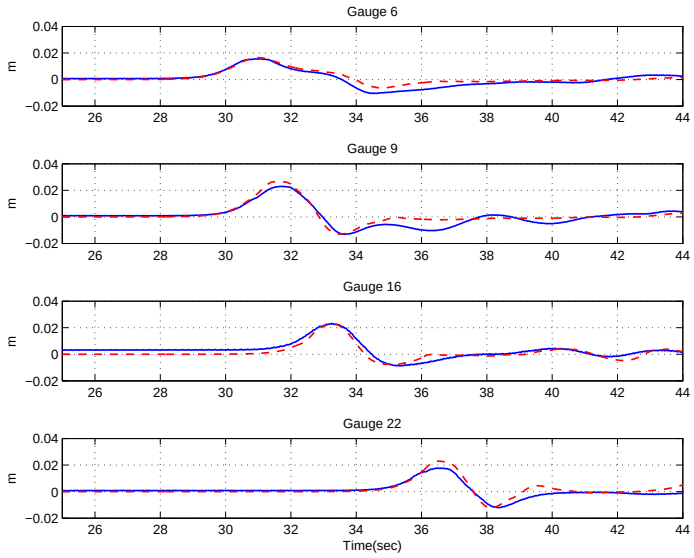

Figure 18: Comparison of computed and measured time series of free surface for $H/d = 0.045$.Solid lines: measured, Dashed lines: Computed.

	Gauge Number			
H/d	6	9	16	22
0.045	6.0	13.2	0.1	18.9
0.091	3.2	16.6	11.6	0.26
0.181	1.6	13.33	13.8	13.3

Table 1: Percent error of predicted maximum runup calculated for each gauge in conical island test.

DEPTH_FILE = ../input/depth.txt

! ————————DIMENSION————————————
Mglob = 702
Nglob = 602

! ———————— TIME————————————————-
TOTAL_TIME = 25.0
PLOT_INTV = 1.0
PLOT_INTV_STATION = 0.05
SCREEN_INTV = 1.0

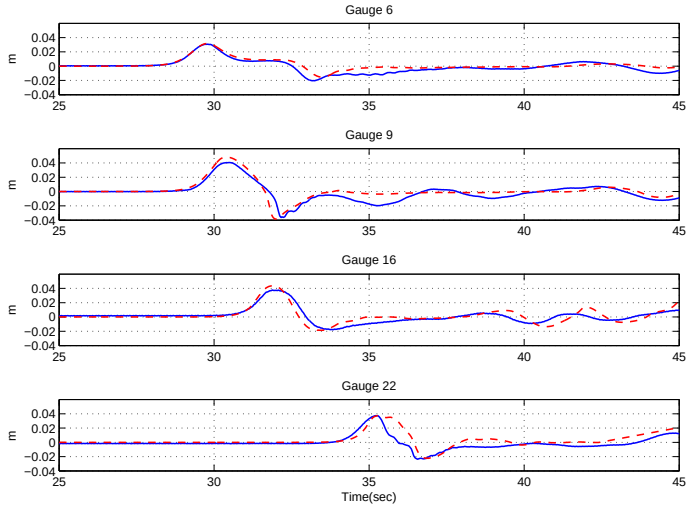

Figure 19: Comparison of computed and measured time series of free surface for $H/d = 0.091$.Solid lines: measured, Dashed lines: Computed.

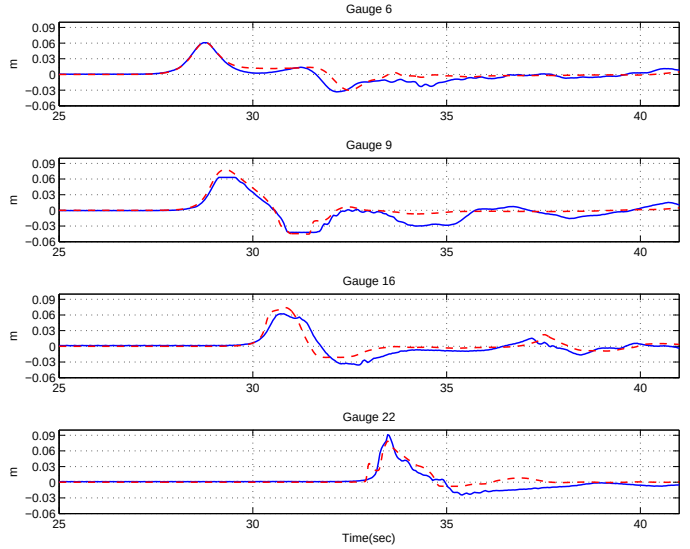

Figure 20: Comparison of computed and measured time series of free surface for $H/d = 0.181$.Solid lines: measured, Dashed lines: Computed.

```
    ! ————————GRID————————————————-
DX = 0.05
DY = 0.05

    ! ————————-WAVEMAKER———————————
WAVEMAKER = INI_SOL
AMP = 0.0144 ! 0.02912 for case B and 0.05792 for case C
DEP = 0.32
LAGTIME = 5.0
XWAVEMAKER = 8.0

    ! ————————- SPONGE LAYER ——————————
DIRECT _SPONGE = F
FRICTION_SPONGE = F
DIFFUSION_SPONGE = F

    ! ————————-PHYSICS———————————
DISPERSION = T
Gamma1 = 1.0
Gamma2 = 1.0
Gamma3 = 1.0
Beta_ref=-0.531
SWE_ETA_DEP = 0.80
Cd = 0.01 ! note: not sensitive, could give a small value like 0.001

    ! ————————-NUMERICS————————————-
Time_Scheme = Runge_Kutta
HIGH_ORDER = FOURTH
CONSTRUCTION = HLLC
CFL = 0.5

    ! ——————WET-DRY————————————-
MinDepth=0.001
MinDepthFrc = 0.001

    ! ————————OUTPUT———————————
NumberStations = 16
STATIONS_FILE = gauges.txt
DEPTH_OUT = T
ETA = T
```

Hmax = T
MASK = T

The results of surface elevation at seven gauges are saved as sta_0001 ... sta_0016 as specified in input.txt. Model/data comparisons are carried out at sta_0001 (gauge 6), sta_0002 (gauge 9), sta_0008 (gauge16), sta_0022 (gauge 22). A post-processing MATLAB script called *plot_comparison.m* can be used for model/data comparisons.

5.7 Solitary wave runup on a shelf with an island (in directory /solitary_runup_2d)

In this test, we performed a simulation of the solitary wave runup measured recently in a large wave basin at Oregon State University's O.H. Hinsdale Wave Research Laboratory. The basin is 48.8 m long, 26.5 m wide, and 2.1 m deep. A complex bathymetry has been constructed, which consists of a 1:30 slope planar beach connected to a triangle shaped shelf and a conical island on the shelf as shown in Figure 21. Solitary waves were generated on the left side by a piston-type wavemaker. Surface elevation and velocity were collected at many locations by wave gauges and ADV's in alongshore and cross-shore arrays (See Swigler and Lynett, 2011 for details). Figure 21 shows wave gauges (circles) and ADV's (triangles) used for model/data comparisons in the present study. Gauge 1 - 9 were located at $(x, y) =$ (7.5, 0.0), (13.0, 0.0), (21.0 0.0), (7.5 5.0), (13.0 5.0), (21.0 5.0), (25.0 0.0), (25.0 5.0) and (25.0, 10.0), respectively. ADV 1 - 3 were located at (13.0 0.0), (21.0, 0.0) and (21.0, -5.0), respectively.

The modeled bathymetry was constructed by combining the measured data of the shelf and the analytical equation of the cone, which was used for the design of the island in the experiment. The computational domain was modified by extending the domain from x = 0.0 m to -5.0 m with a constant water depth of 0.78 m in order to use a solitary wave solution as an initial condition. The width of the computational domain in the y direction is the same as OSU's basin. Grid spacing used in the model is 0.1 m in both directions. A solitary wave solution based on Nwogu's extended Boussinesq equations was used with centroid located at x = 5.0 The wave height is 0.39 m as that used in the laboratory experiment.

Figure 22 shows results of computed water surfaces at t= 6.4 s, 8.4 s and 14.4 s, respectively. The wave front becomes very steep as the wave climbs on the shelf, which was well captured by the model. The wave scattering pattern is clearly seen in the bottom panel of Figure 22. Wave breaking on the shelf was observed in the laboratory experiment and was also seen in the model.

Figure 23 shows time series of modeled surface elevations and measurements at Gauge 1 - 9 (from top to bottom). Good agreement between model and data is found at the gauge in front of the island (Gauge 1, top panel), as the model successfully predicts the solitary wave propagation and its reflection from the shore. The model also captures the collision of edge waves propagating around the two sides of the island, as indicated at the gauge behind the island (Gauge 3). The model predicts the timing of wave collision well but over-predicts the peak of wave runup. The model/data comparisons at Gauges 5, 6, 8, and 9, which are located at the north-side shelf, indicates that the model predicts wave refraction and breaking on the shelf reasonably well.

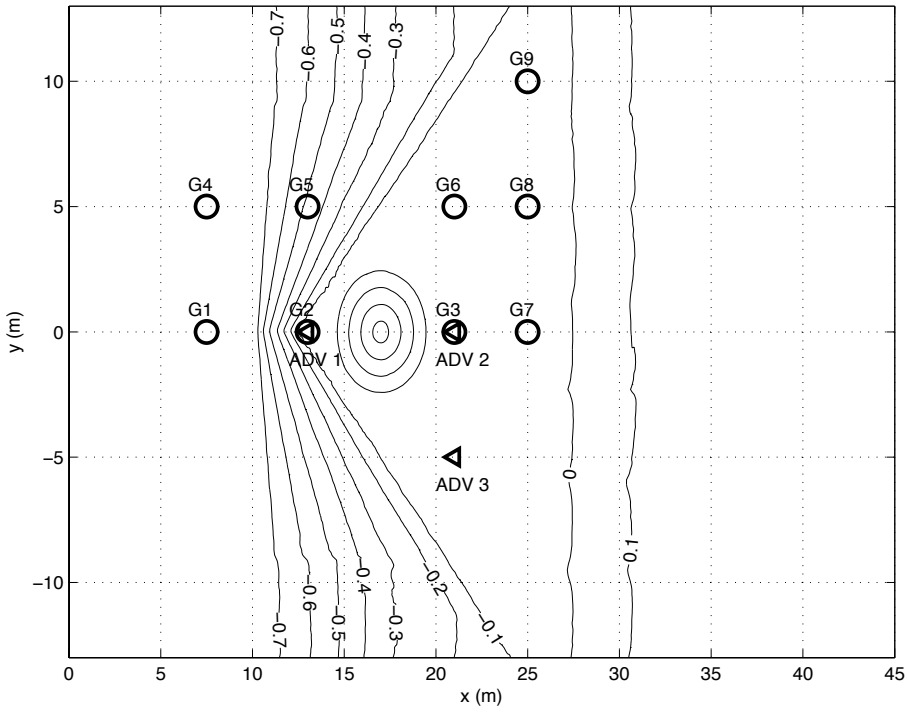

Figure 21: Bathymetry contours (in meters) and measurement locations used in model/data comparisons. Circles: pressure gauges, triangles: ADV.

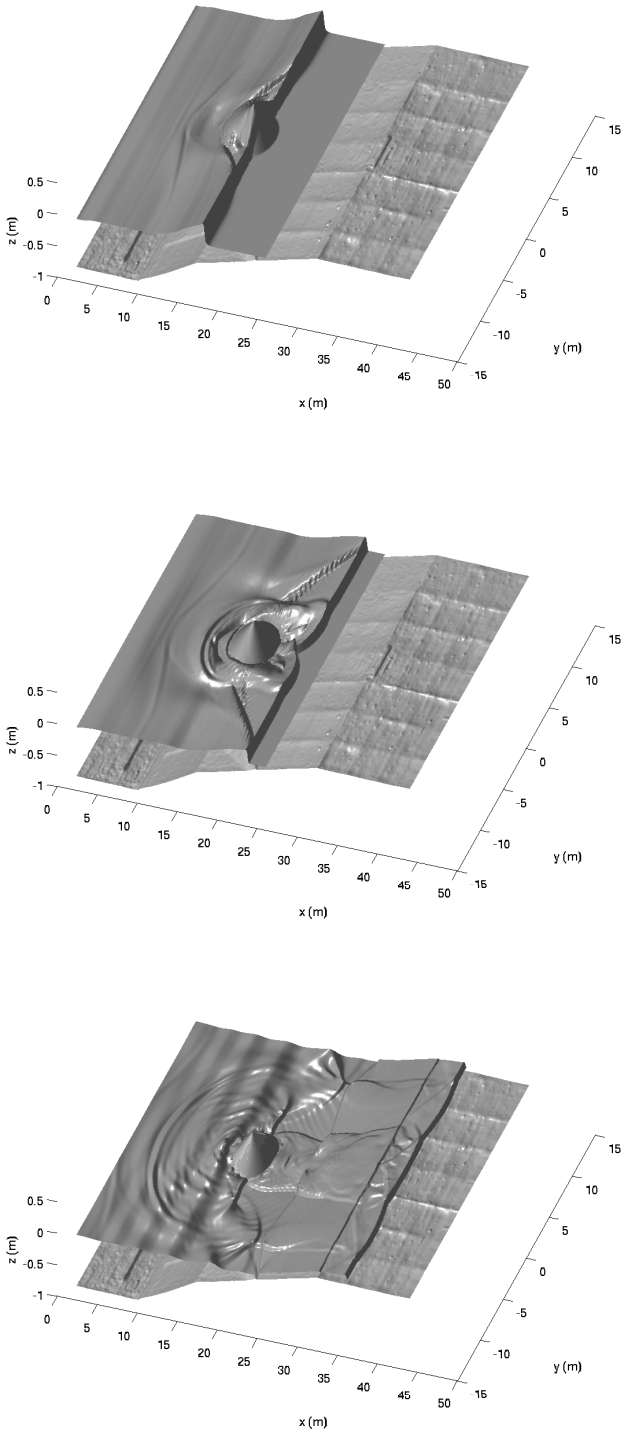

Figure 22: Modeled water surface at (top) t = 6.4 s, (middle) t =8.4 s, (bottom) t =14.4 s.

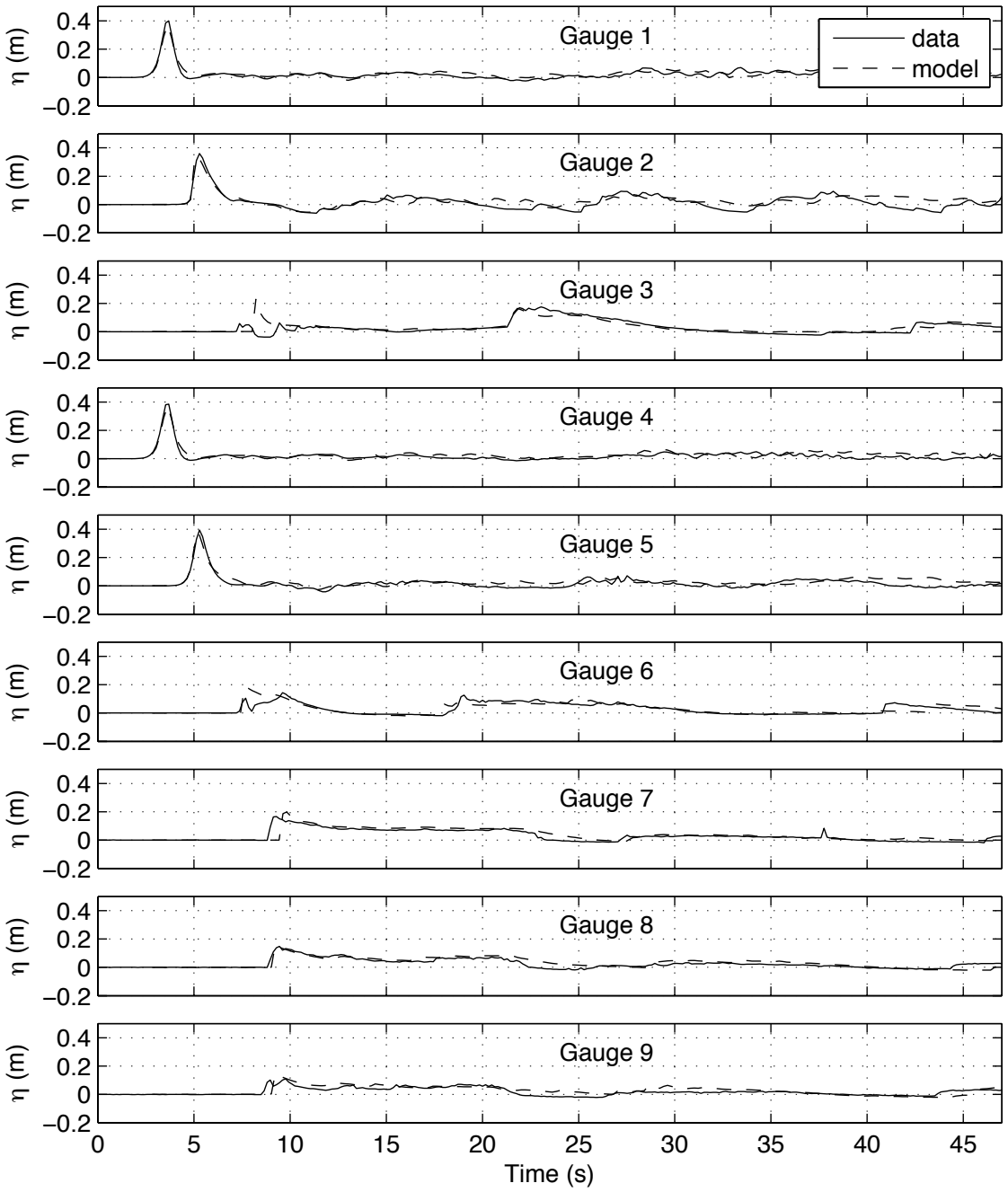

Figure 23: Model/data comparisons of time series of surface elevation at (top) Gauge 1 - Gauge 9. Solid line: model, stars: data.

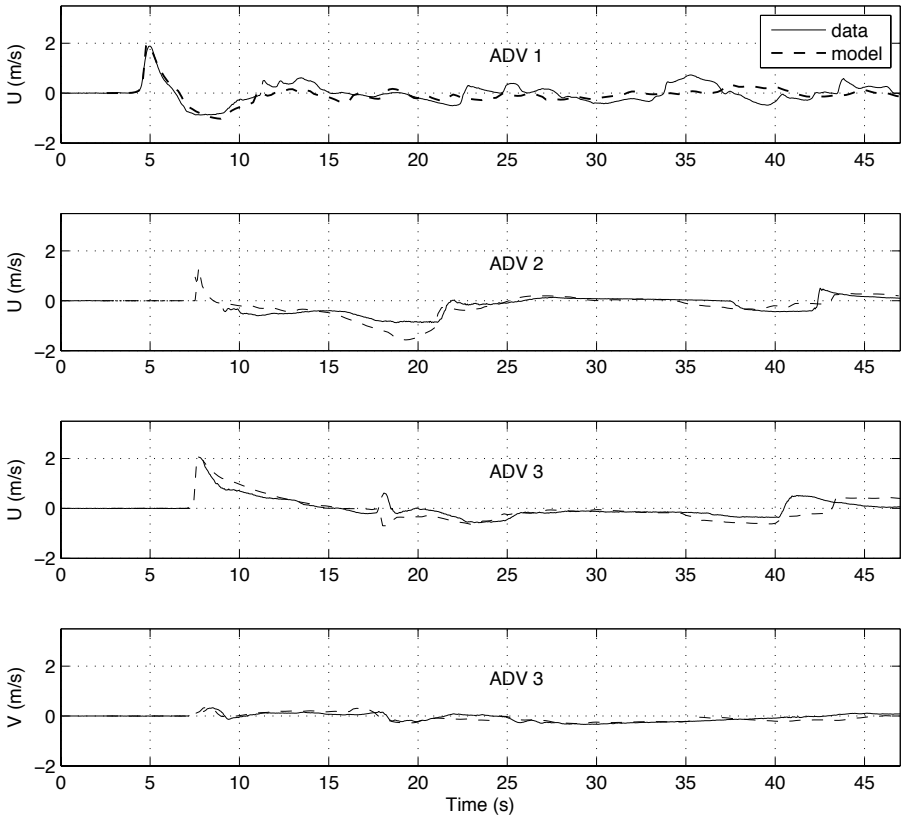

Figure 24: Model/data comparisons of time series of velocity u component at ADV 1 (top panel), ADV 2 (second panel), ADV 3 (third panel), and v component at ADV 3 (bottom panel). Solid line: model, dashed line: data.

Figure 24 shows model/data comparisons of velocity time series of velocity u component at ADV 1 (top panel), ADV 2 (second panel), ADV 3 (third panel), and v component at ADV 3 (bottom panel). The model predicts the peak velocity and the entire trend of velocity variation in time at both locations. An underprediction of the seaward velocity is found at ADV 2. The velocity in the y-direction was not compared at ADV 1 and ADV2 because the measured values were too small.

The input parameters for this test are listed below.

```
    ! ————————————DEPTH————————————————————-
DEPTH_TYPE = DATA
DEPTH_FILE = ../input/depth_wkshop.txt

    ! ————————————DIMENSION————————————————
Mglob = 501
Nglob = 261

    ! ———————— TIME————————————————————-
TOTAL_TIME = 45.0
PLOT_INTV = 0.2
PLOT_INTV_STATION = 0.2
SCREEN_INTV = 0.2

    ! ————————————GRID————————————————————-
DX = 0.1
DY = 0.1

    ! ————————————-WAVEMAKER————————————————
WAVEMAKER = INI_SOL
AMP = 0.39
DEP = 0.78
LAGTIME = 5.0
XWAVEMAKER = 10.0 ! note that means x = 5 m because the domain starts at x = -5

    ! ————————————- SPONGE LAYER ————————————
DIRECT _SPONGE = F
FRICTION_SPONGE = F
DIFFUSION_SPONGE = F

    ! ————————————-PHYSICS————————————————
DISPERSION = T
Gamma1 = 1.0
```

Gamma2 = 1.0
Gamma3 = 1.0
Beta_ref=-0.531
SWE_ETA_DEP = 0.80
Cd = 0.001

! ————————-NUMERICS————————————-
Time_Scheme = Runge_Kutta
HIGH_ORDER = FOURTH
CONSTRUCTION = HLLC
CFL = 0.5

! ——————WET-DRY————————————-
MinDepth=0.001
MinDepthFrc = 0.001

! ———————OUTPUT———————————
DEPTH_OUT = T
U = T
V = T
ETA = T
MASK = T

Several post-processing procedures are performed in order to get model/data comparisons. First, *read_result.m* is used to obtained time series of η at 9 pressure gauges , and u and v at 3 ADV locations. Then, *plot_WG.m* is used for comparisons of surface elevation at the 9 gauges, *plot_ADV_AB.m* for comparisons of u at ADV 1 and 2, and *plot_ADV_C.m* for u and v at ADV 3.

5.8 Nesting case

We provide a 1-D example indicating that a solitary wave of permanent form is undistorted as it moves from a large domain Grid_A to a smaller scale grid Grid_B. In Grid_A, the computational domain is 6000m long with 10 m constant water depth. Grid size is 4 m. An initial solitary wave with an amplitude of 1m is centered at x=1000.0 m. Grid_B, with a grid size of dx=2m , is nested inside Grid_A. The nesting boundary is at x=2000.0m in Grid_A. The calculation of Grid_A provides a time series of u, v and η through output at gauges at x=2000.0m ($i = 500$, where i is grid number in x direction in Grid_A). The spherical mode is used a demonstration of the nesting case.

There are three sub-directories for the nesting example.

directory: /examples/sph_nesting /

/grid_A/ contains input file, executive and station file for Grid_A run.

/make_nest_file/ contains a Fortran code to generate the nesting input based on output from Grid_A.

/grid_B/ contains input file and executive file for Grid_B calculation.

procedure:

1) run model Grid_A. **NOTE:** the executive file should be compiled with '-DCOUPLING' turned OFF (Makefile in /src/).

2) generate nesting input file using convert.f in /make_nest_file/.

3)run model Grid_B. **NOTE:** the executive file should be compiled with '-DCOUPLING' turned ON (Makefile in /src/).

postprocessing: use plot_ab.m to plot results. Figure 25 shows results from Grid_A (upper pannl) and Grid_B (lower panel).

The input parameters for Grid_A are listed below.

```
    ! ——————————DEPTH———————————————-
DEPTH_TYPE = FLAT
DEPTH_FLAT = 10.0

    ! ——————————— TIME———————————————-
TOTAL_TIME = 400.0
PLOT_INTV = 1.0
PLOT_INTV_STATION = 0.0 ! 0.0 means make station output every time step for nesting data
SCREEN_INTV = 1.0

    ! ——————————GRID———————————————-
StretchGrid = F
Lon_West = 120.0
Lat_South = 0.0
Dphi = 0.000035972
Dtheta = 0.0001

    ! ——————————-WAVEMAKER———————————————
WAVEMAKER = INI_SOL
AMP = 0.1
DEP = 10.0
LAGTIME = 0.0
XWAVEMAKER = 1000.0

    ! ——————————-PHYSICS———————————————
DISPERSION = T
```

```
Gamma1 = 1.0
Gamma2 = 0.0
Gamma3 = 1.0
Beta_ref=-0.531
SWE_ETA_DEP = 0.80
Cd = 0.00

    ! ————————-NUMERICS———————————-
Time_Scheme = Runge_Kutta
HIGH_ORDER = THIRD
CONSTRUCTION = HLLC
CFL = 0.5
FroudeCap = 10.0

    ! ————————OUTPUT———————————
NumberStations = 3
STATIONS_FILE = station.txt
```

The input parameters for Grid_B are listed below.

```
    ! ————————DEPTH———————————————-
DEPTH_TYPE = FLAT
DEPTH_FLAT = 10.0

    ! ———————— TIME———————————————-
TOTAL_TIME = 400.0
PLOT_INTV = 1.0
PLOT_INTV_STATION = 1.00
SCREEN_INTV = 1.0

    ! ————————GRID———————————————-
StretchGrid = F
Lon_West = 120.0
Lat_South = 0.0
Dphi = 0.000017986
Dtheta = 0.0001

    ! ————————-WAVEMAKER———————————
WAVEMAKER = INI_SOL
AMP = 0.1
DEP = 10.0
```

```
LAGTIME = 0.0
XWAVEMAKER = 1000.0

    ! ——————-PHYSICS————————————
DISPERSION = T
Gamma1 = 1.0
Gamma2 = 0.0
Gamma3 = 1.0
Beta_ref=-0.531
SWE_ETA_DEP = 0.80
Cd = 0.00

    ! ——————-NUMERICS————————————-
Time_Scheme = Runge_Kutta
HIGH_ORDER = THIRD
CONSTRUCTION = HLLC
CFL = 0.5
FroudeCap = 10.0

    ! ——————— COUPLING ——————————-
COUPLING_FILE = ../make_nest_file/coupling.txt
```

6 Dealing with numerical instability

One of the goals of the new code development is to improve numerical stability for large-scale and long time simulations. It has been indicated, by both our own tests and users' feedback, that the numerical stability of the TVD version of FUNWAVE has been greatly improved in contrast to the previous version. No more filtering is required in FUNWAVE-TVD. However, numerical instability is a general property of numerical algorithms, especially as a higher-order numerical scheme is applied. Here, we would like to share our experiences in dealing with numerical instabilities.

1. Avoid large bathymetric slope.

   A rapidly varying bathymetry may cause numerical noises due to a large ∇h in the source term. Although the surface gradient term is in a well-balanced form as described in the theory section, an unreasonably large bottom slope may cause numerical instability. Based on our experiences, $|\nabla h| > 1.0$ should be avoided.

2. Use FroudeCap to limit a huge velocity occurring in extremely shallow water

   The model uses a Froude number cap to avoid an unrealistic flow velocity in extremely shallow water specified as $MinDepthFrc$ by a user. The unrealistic velocity usually occurs at

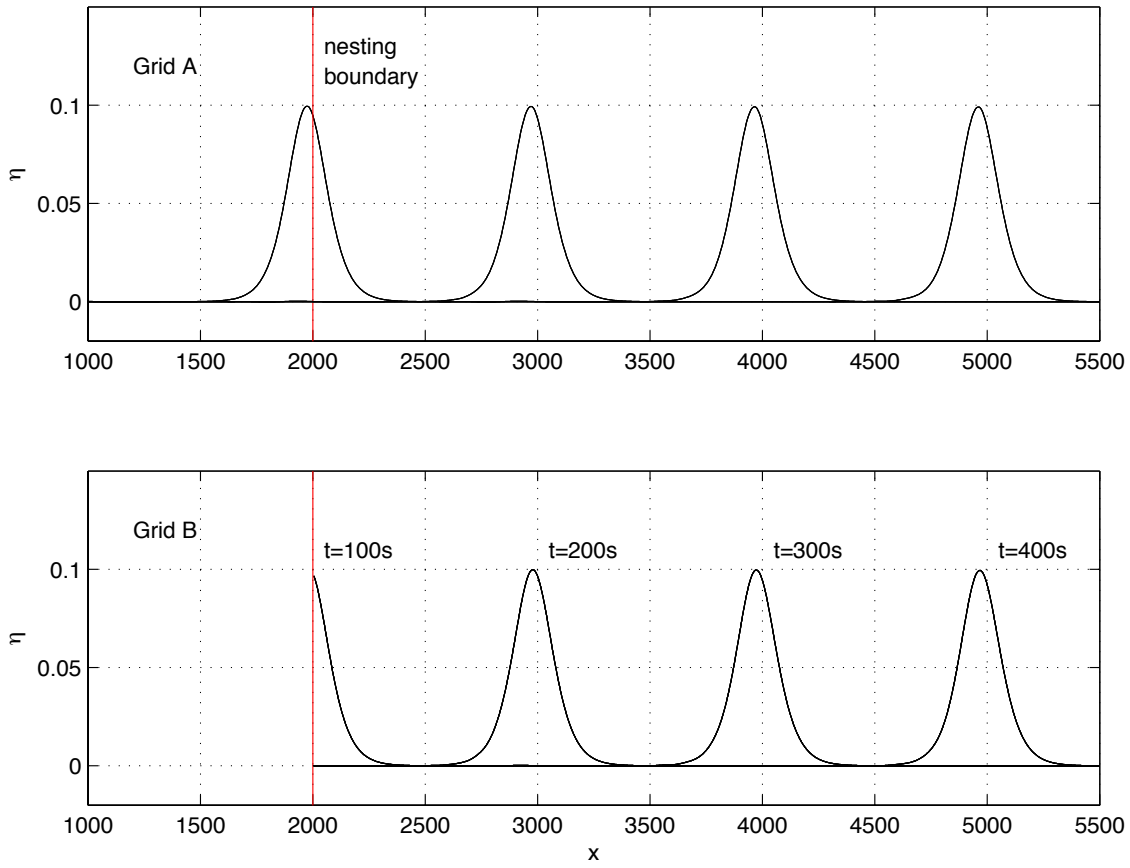

Figure 25: Solitary wave calculated in a larger domain Grid_A (upper) and in a nested smaller domain Grid_B (lower) at t=100s, 200s, 300s, and 400s.

a wetting-drying interface. The default value for FroudeCap is 10.0. Using a smaller value such as FroudeCap = 3.0 or smaller can avoid the problem and also improve the model efficiency in terms of the CFL algorithm. Note: a FroudeCap smaller than 1.5 is not suggested because it may cap velocities at flooding water fronts.

3. Adjust MinDepthFrc

   The default value of the wetting-drying threshold, MinDepthFrc, is 0.001 m. Using a small value may cause a large velocity at a wetting-drying point, resulting in a small dt. It may also cause numerical instability. Use a larger value when a problem occurs.

4. Use a smaller Courant number

   The default value of CFL is 0.5. A smaller value can be used if instability occurs.

5. Use a lower-order spatial scheme

   The MUSCL-TVD schemes up to fourth-order are implemented in the code. Initially we used the scheme introduced by Yamamoto et al. (1998) who applied the Minmod limiter in the fourth-order scheme. Abadie et al. (2012) pointed out that the fourth-order scheme has an instability problem for simulations in deep water. For this reason, the third-order scheme is suggested if an instability problem occurs.

   In Version 3.0, we implemented the fourth-order scheme of Erduran et al. (2005) who used the van-Leer limiter for the third-order part of the fourth-order scheme. We hope that this scheme can improve the model stability.

7 References

Agnon, Y., Madsen, P. A. and Schäffer, H. A., 1999, "A new approach to high-order Boussinesq models", *J. Fluid Mech.*, **399**, 319-333.

Berkhoff, J. C. W., Booy, N. and Radder, A. C., 1982, "Verification of numerical wave propagation models for simple harmonic linear water waves", *Coast. Engrg.*, **6**, 255-279.

Bermùdez, A. and Vàzquez, M. E., 1994, "Upwind methods for hyperbolic conservation laws with source terms", *Comput. Fluids*, **23** (8), 1049-1071.

Briggs, M. J., Synolakis, C. E., and Harkins, G. S., 1994, "Tsunami run- up on a conical island", *Proc., Waves-Physical and Numerical Modelling*, International Association for Hydraulic Research, Delft, The Netherlands, 446-455.

Brown, J., MacMahan, J., Reniers, A. J. H. M. and Thornton, E., 2009, "Surf zone diffusivity on a rip-channeled beach", *J. Geophys. Res.*, **114**, C11015, doi:10.1029/2008JC005158.

Chen, Q., Dalrymple, R. A., Kirby, J. T., Kennedy, A. B. and Haller, M. C., 1999, "Boussinesq modelling of a rip current system", em J. Geophys. Res., **104**, 20,617-20,637.

Chen, Q., Kirby, J. T., Dalrymple, R. A., Kennedy, A. B. and Chawla, A., 2000, "Boussinesq modeling of wave transformation, breaking and runup. II: 2D", *J. Waterway, Port, Coastal and Ocean Engineering*, **126**, 48-56.

Chen, Q., Kirby, J. T., Dalrymple, R. A., Shi, F. and Thornton, E. B., 2003, "Boussinesq modeling of longshore currents", *Journal of Geophysical Research*, **108**(C11), 3362, doi:10.1029/2002JC001308.

Chen, Q., Kaihatu, J. M., and Hwang, P. A., 2004, "Incorporation of Wind Effects Into Boussinesq Wave Models", *J. Waterway, Port, Coastal and Ocean Engineering*, **130**, 312-321.

Chen, Q., 2006, "Fully nonlinear Boussinesq-type equations for waves and currents over porous beds", *Journal of Engineering Mechanics.* **132** (2): 220-230.

Erduran, K. S., Ilic, S., and Kutija, V., 2005, "Hybrid finite-volume finite-difference scheme for the solution of Boussinesq equations", *Int. J. Numer. Meth. Fluid.*, **49**, 1213-1232.

Gobbi, M. F., Kirby, J. T. and Wei, G., 2000, "A fully nonlinear Boussinesq model for surface waves. II. Extension to $O(kh^4)$", *Journal of Fluid Mechanics*, **405**, pp. 181-210.

Goda, Y., 2000, Random seas and design of maritime structures, *Advanced Series on Ocean Engineering*, Volume 15, 2nd Edition, World Scientific, Singapore.

Gottlieb, S., Shu C.-W., and Tadmore, E., 2001, "Strong stability-preserving high-order time discretization methods", *SIAM Review*, **43** (1), 89 - 112.

Hansen, J. B., and Svendsen, I. A., 1979, "Regular waves in shoaling water: Experimental data", Tech. Rep. ISVA Ser., 21, Technical Univ. of Denmark, Denmark.

Kennedy, A. B., Chen, Q., Kirby, J. T. and Dalrymple, R. A., 2000, "Boussinesq modeling of wave transformation, breaking and runup. I: 1D", *J. Waterway, Port, Coastal and Ocean Engineering*, **126**, 39-47.

Kennedy, A. B., Kirby, J. T., Chen, Q. and Dalrymple, R. A., 2001, "Boussinesq-type equations with improved nonlinear performance", *Wave Motion*, **33**, pp. 225-243.

Kim D. H., Cho, Y. S., and Kim, H. J., 2008, "Well-balanced scheme between flux and source terms for computation of shallow-water equations over irregular bathymetry", *Journal of Engineering Mechanics*, **134**, 277-290.

Kim, D. H., Lynett, P. J. and Socolofsky, S. A., 2009, "A depth-integrated model for weakly dispersive, turbulent, and rotational fluid flows", *Ocean Modeling*, **27**, 198-214.

Kim, D. H., 2009, "Turbulent flow and transport modeling by long waves and currents", Ph.D. dissertation, Texas A& M University.

Kirby, J. T., Wei, G., Chen, Q., Kennedy, A. B. and Dalrymple, R. A., 1998, "FUNWAVE 1.0, Fully nonlinear Boussinesq wave model. Documentation and users manual". Report CACR-98-06, Center for Applied Coastal Research, Department of Civil and Environmental Engineering, University of Delaware.

Kirby, J.T., Shi, F., Watts, P., Grilli, S.T., 2004, "Propagation of short, dispersive tsunami waves in ocean basins". EOS Transactions of the AGU 85 (47) Abstract OS21E-02.

Kirby, J. T., Shi, F., Harris, J. C., and Grilli, S. T., 2012, "Sensitivity analysis of trans-oceanic tsunami propagation to dispersive and Coriolis effects", *Ocean Modelling*, under revision.

Liang, Q. and Marche, F., 2009, "Numerical resolution of well-balanced shallow water equations with complex source terms", *Advances in Water Resources*, **32**, 873 - 884.

Long, W. and Kirby, J. T., 2006, "Boussinesq modeling of waves, currents and sediment transport", Research Report No. CACR-06-02, Center for Applied Coastal Research, Dept. of Civil and Environmental Engineering, Univ. of Delaware, Newark.

Lynett, P. J., Wu, T.-R. and Liu, P. L.-F., 2002, "Modeling wave runup with depth-integrated equations", *Coastal Engineering*, **46**, 89-107.

Madsen, P.A., Srensen, O.R., 1992, "A new form of the Boussinesq equations with improved linear dispersion characteristics. Part 2. A slowly-varying bathymetry", *Coastal Engineering* **18** (3-4), 183-204.

Mase, H., and Kirby, J. T., 1992, "Hybrid frequency-domain KdV equation for random wave transformation", *Proc., 23rd Int. Conf. Coast. Eng.*, ASCE, New York, 474-487.

Naik, N. H., Naik, V. K., and Nicoules, M., 1993, "Parallelization of a class of implicit finite difference schemes in computational fluid dynamics", *International Journal of High Speed Computing*, **5**: 1-50.

Ning, D. Z., Zang, J., Liang, Q., Taylor, P. H., and Borthwick, A. G. L., 2008, "Boussinesq cut-cell model for non-linear wave interaction with coastal structures", *International Journal for Numerical Methods in Fluids*, **57** (10), 1459-1483.

Nwogu, O., 1993, "An alternative form of the Boussinesq equations for nearshore wave propagation", *Journal of Waterway, Port, Coastal, and Ocean Engineering*, **119** (6), pp. 618-638.

Nwogu, O. and Demirbilek, Z., 2001, "BOUSS-2D: A Boussinesq wave model for coastal regions and harbors", ERDC/CHL TR-01-25, Coastal and Hydraulics Laboratory, USACOE Engineer Research and Development Center, Vicksburg, MS.

Roeber, V., Cheung, K. F., and Kobayashi, M. H., 2010, "Shock-capturing Boussinesq-type model for nearshore wave processes", *Coastal Engineering*, **57**, 407-423.

Rogers, B. D., Borthwick, A. G. L., and Taylor, P. H., 2003, "Mathematical balancing of flux gradient and source terms prior to using Roe's approximate Riemann solver", *Journal of Computational Physics*, **192**, 422-451.

Shi, F., Kirby, J. T., Harris, J. C., Geiman, J. D., and Grilli, S. T., 2012 a, "A high-order adaptive time-stepping TVD solver for Boussinesq modeling of breaking waves and coastal inundation", *Ocean Modelling*, 43-44, 36-51.

Shi, F., Kirby, J. T., and Tehranirad, B. , 2012 b, Tsunami benchmark results for spherical coordinate, Center for Applied Coastal Research Report, CACR 2012-02, University of Delaware, Newark, Delaware.

Shi, F., Kirby, J. T., Tehranirad, B. and Harris, J. C., 2011, FUNWAVE-TVD, Version 1.0, users' manual and benchmark tests, Center for Applied Coastal Research Report, CACR 2011-04, University of Delaware, Newark, Delaware.

Shi, F., Dalrymple, R. A., Kirby, J. T., Chen, Q. and Kennedy, A., 2001, "A fully nonlinear Boussinesq model in generalized curvilinear coordinates". *Coastal Engineering*, **42**, pp. 337-358.

Shiach, J. B. and Mingham, C. G., 2009, "A temporally second-order accurate Godunov-type scheme for solving the extended Boussinesq equations", *Coastal Engineering*, **56**, 32-45.

Sitanggang, K. I. and Lynett, P., 2005, "Parallel computation of a highly nonlinear Boussinesq equation model through domain decomposition", *Int. J. Num. Meth. Fluids*, **49**, 57-74.

Smagorinsky, J., 1963, "General circulation experiments with the primitive equations. I. The basic experiment", *Mon. Weather Rev*, **91**, 99-165.

Swigler, D. and Lynett, P. ,2011, "Laboratory study of the three-dimensional turbulence and kinematic properties associated with a solitary wave traveling over an alongshore-variable, shallow shelf", in review.

Synolakis, C. E., Bernard, E. N., Titov, V. V., Kânoglu, U. and González, F. I., 2007, "Standards, criteria, and procedures for NOAA evaluation of tsunami numerical models", *NOAA Tech. Memo. OAR PMEL-135*, National Oceanic and Atmospheric Administration.

Tehranirad, B., Shi, F., Kirby, J. T., Harris, J. C. and Grilli, S., 2011, "Tsunami benchmark results for fully nonlinear Boussinesq wave model FUNWAVE-TVD, Version 1.0", Research Report No. CACR-11-02, Center for Applied Coastal Research, University of Delaware.

Ting, F.C.K., Kirby, J.T., 1994, "Observation of undertow and turbulence in a laboratory surf zone". *Coast. Eng.* **24**, 5180.

Tonelli, M. and Petti, M., 2009, "Hybrid finite volume - finite difference scheme for 2DH improved Boussinesq equations", *Coast. Engrng.*, **56**, 609-620.

Tonelli, M. and Petti, M., 2010, "Finite volume scheme for the solution of 2D extended Boussinesq equations in the surf zone", *Ocean Engrng.*, **37**, 567-582.

Toro, E. F., 2009, *Riemann solvers and numerical methods for fluid dynamics: a practical introduction*, Third edition, Springer, New York.

Wei, G., Kirby, J.T., Grilli, S.T., Subramanya, R., 1995, "A fully nonlinear Boussinesq model for surface waves: Part I. Highly nonlinear unsteady waves", *Journal of Fluid Mechanics* **294**, 7192.

Wei, G. and Kirby, J. T., 1995, "A time-dependent numerical code for extended Boussinesq equations", *Journal of Waterway, Port, Coastal and Ocean Engineering*, **120**, pp. 251-261.

Yamamoto, S., Daiguji, H., 1993, "Higher-order-accurate upwind schemes for solving the compressible Euler and NavierStokes equations", *Computers and Fluids*, **22** (2/3), 259270.

Yamamoto, S., Kano, S. and Daiguji, H, 1998, "An efficient CFD approach for simulating unsteady hypersonic shockshock interference flows", *Computers and Fluids*, **27** (56), pp. 571-580.

Zelt, J. A., 1991, "The runup of nonbreaking and breaking solitary waves", *Coastal Engineering*, **15**, pp. 205-246.

Zhen, F., 2004, "On the numerical properties of staggered vs. non-staggered grid schemes for a Boussinesq equation model", MCE Thesis, Department of Civil and Environmental Engineering, University of Delaware.

Zhou, J. G., Causon, D. M., Mingham C. G., and Ingram, D. M., 2001, "The surface gradient method for the treatment of source terms in the shallow-water equations", *Journal of Computational Physics*, **168**, 1-25.

8 Appendix A. Expansions of $(U_1^{'}, V_1^{'})$, $(U_1^{''}, V_1^{''})$, (U_2, V_2), (U_3, V_3) and (U_4, V_4)

Note that there are typos in the expansion expressions in both Shi et al. (2012) and the previous manual lower than Version 3.0 (Thanks to Young-Kwang Choi for pointing out the typos and checking the corresponding code).

The expanded forms of $(U_1^{'}, V_1^{'})$, $(U_1^{''}, V_1^{''})$, (U_2, V_2), (U_3, V_3) and (U_4, V_4) can be written as

$$U_1^{'} = \frac{1}{2}(1-\beta)^2 h^2 (u_{xx} + v_{xy}) - (1-\beta) h[(hu)_{xx} + (hv)_{xy}] \tag{107}$$

$$V_1^{'} = \frac{1}{2}(1-\beta)^2 h^2 (u_{xy} + v_{yy}) - (1-\beta) h[(hu)_{xy} + (hv)_{yy}] \tag{108}$$

$$\begin{aligned} U_1^{''} = & -\Big[\eta\eta_x(u_{tx} + v_{ty}) + \frac{1}{2}\eta^2(u_{txx} + v_{txy}) + \eta_x\Big\{(hu)_{tx} + (hv)_{ty}\Big\} + \eta\Big\{(hu)_{txx} + (hv)_{txy}\Big\}\Big] \\ & -\Big[\beta(1-\beta)h\eta_t - \beta^2\eta\eta_t\Big](u_{xx} + v_{xy}) - \Big[\beta(1-\beta)h\eta - \frac{1}{2}\beta^2\eta^2\Big](u_{txx} + v_{txy}) \\ & +\beta\eta_t\Big[(hu)_{xx} + (hv)_{xy}\Big] + \beta\eta\Big[(hu)_{txx} + (hv)_{txy}\Big] \end{aligned} \tag{109}$$

$$\begin{aligned} V_1^{''} = & -\Big[\eta\eta_y(u_{tx} + v_{ty}) + \frac{1}{2}\eta^2(u_{txy} + v_{tyy}) + \eta_y\Big\{(hu)_{tx} + (hv)_{ty}\Big\} + \eta\Big\{(hu)_{txy} + (hv)_{tyy}\Big\}\Big] \\ & -\Big[\beta(1-\beta)h\eta_t - \beta^2\eta\eta_t\Big](u_{xy} + v_{yy}) - \Big[\beta(1-\beta)h\eta - \frac{1}{2}\beta^2\eta^2\Big](u_{txy} + v_{tyy}) \\ & +\beta\eta_t\Big[(hu)_{xy} + (hv)_{yy}\Big] + \beta\eta\Big[(hu)_{txy} + (hv)_{tyy}\Big] \end{aligned} \tag{110}$$

$$\begin{aligned} U_2 = & \Big[(\beta-1)(h+\eta)\Big\{u[(hu)_x + (hv)_y]_x + v[(hu)_x + (hv)_y]_y\Big\} \\ & +\Big\{\frac{1}{2}(1-\beta)^2h^2 - \beta(1-\beta)h\eta + \frac{1}{2}(\beta^2-1)\eta^2\Big\}\Big\{u(u_x+v_y)_x + v(u_x+v_y)_y\Big\} \\ & +\frac{1}{2}\Big\{(hu)_x + (hv)_y + \eta(u_x+v_y)\Big\}^2\Big]_x \end{aligned} \tag{111}$$

$$\begin{aligned} V_2 = & \Big[(\beta-1)(h+\eta)\Big\{u[(hu)_x + (hv)_y]_x + v[(hu)_x + (hv)_y]_y\Big\} \\ & +\Big\{\frac{1}{2}(1-\beta)^2h^2 - \beta(1-\beta)h\eta + \frac{1}{2}(\beta^2-1)\eta^2\Big\}\Big\{u(u_x+v_y)_x + v(u_x+v_y)_y\Big\} \\ & +\frac{1}{2}\Big\{(hu)_x + (hv)_y + \eta(u_x+v_y)\Big\}^2\Big]_y \end{aligned} \tag{112}$$

$$U_3 = -v\omega_1 - \omega_0\Big[\Big\{(\beta-\frac{1}{2})(h+\eta)\Big\}\Big\{(hu)_x+(hv)_y\Big\}_y + \Big\{(\frac{1}{3}-\beta+\frac{1}{2}\beta^2)h^2+(\frac{1}{6}-\beta+\beta^2)\eta h+(\frac{1}{2}\beta^2-\frac{1}{6})\eta^2\Big\}(u_x+v_y)_y\Big] \quad (113)$$

$$V_3 = u\omega_1 + \omega_0\Big[\Big\{(\beta-\frac{1}{2})(h+\eta)\Big\}\Big\{(hu)_x+(hv)_y\Big\}_x + \Big\{(\frac{1}{3}-\beta+\frac{1}{2}\beta^2)h^2+(\frac{1}{6}-\beta+\beta^2)\eta h+(\frac{1}{2}\beta^2-\frac{1}{6})\eta^2\Big\}(u_x+v_y)_x\Big] \quad (114)$$

$$\omega_0 = v_x - u_y \quad (115)$$

$$\omega_1 = \Big[(\beta-1)h_x+\beta\eta_x\Big]\Big[\Big\{(hu)_x+(hv)_y\Big\}_y + \Big\{(\beta-1)h+\beta\eta\Big\}(u_x+v_y)_y\Big] - \Big[(\beta-1)h_y+\beta\eta_y\Big]\Big[\Big\{(hu)_x+(hv)_y\Big\}_x + \Big\{(\beta-1)h+\beta\eta\Big\}(u_x+v_y)_x\Big] \quad (116)$$

$$U_4 = \Big(\frac{1}{3}-\beta+\frac{1}{2}\beta^2\Big)h^2(u_{xx}+v_{xy}) + \Big(\beta-\frac{1}{2}\Big)h\Big\{(hu)_{xx}+(hv)_{xy}\Big\} + \Big[\Big\{\Big(\frac{1}{6}-\beta+\beta^2\Big)h\eta+\Big(\frac{1}{2}\beta^2-\frac{1}{6}\Big)\eta^2\Big\}(u_{xx}+v_{xy}) + \Big(\beta-\frac{1}{2}\Big)\eta\Big\{(hu)_{xx}+(hv)_{xy}\Big\}\Big] \quad (117)$$

$$V_4 = \Big(\frac{1}{3}-\beta+\frac{1}{2}\beta^2\Big)h^2(u_{xy}+v_{yy}) + \Big(\beta-\frac{1}{2}\Big)h\Big\{(hu)_{xy}+(hv)_{yy}\Big\} + \Big[\Big\{\Big(\frac{1}{6}-\beta+\beta^2\Big)h\eta+\Big(\frac{1}{2}\beta^2-\frac{1}{6}\Big)\eta^2\Big\}(u_{xy}+v_{yy}) + \Big(\beta-\frac{1}{2}\Big)\eta\Big\{(hu)_{xy}+(hv)_{yy}\Big\}\Big] \quad (118)$$

In detail,

$$
\begin{aligned}
U_2 &= (\beta-1)(h+\eta)_x\Big\{u\Big[(hu)_{xx}+(hv)_{yx}\Big]+v\Big[(hu)_{xy}+(hv)_{yy}\Big]\Big\}\\
&\quad+(\beta-1)(h+\eta)\Big\{u\Big[(hu)_{xx}+(hv)_{yx}\Big]+v\Big[(hu)_{xy}+(hv)_{yy}\Big]\Big\}_x\\
&\quad+\Big\{\frac{1}{2}(1-\beta)^2h^2-\beta(1-\beta)h\eta+\frac{1}{2}(\beta^2-1)\eta^2\Big\}_x\Big\{u(u_x+v_y)_x+v(u_x+v_y)_y\Big\}\\
&\quad+\Big\{\frac{1}{2}(1-\beta)^2h^2-\beta(1-\beta)h\eta+\frac{1}{2}(\beta^2-1)\eta^2\Big\}\Big\{u(u_x+v_y)_x+v(u_x+v_y)_y\Big\}_x\\
&\quad+\Big\{(hu)_x+(hv)_y+\eta(u_x+v_y)\Big\}\Big\{(hu)_x+(hv)_y+\eta(u_x+v_y)\Big\}_x\\
&= (\beta-1)(h_x+\eta_x)\Big\{u\Big[(hu)_{xx}+(hv)_{yx}\Big]+v\Big[(hu)_{xy}+(hv)_{yy}\Big]\Big\}\\
&\quad+(\beta-1)(h+\eta)\Big\{u_x\Big[(hu)_{xx}+(hv)_{yx}\Big]+u\Big[(hu)_{xx}+(hv)_{yx}\Big]_x\\
&\qquad\qquad+v_x\Big[(hu)_{xy}+(hv)_{yy}\Big]+v\Big[(hu)_{xy}+(hv)_{yy}\Big]_x\Big\}\\
&\quad+\Big\{(1-\beta)^2hh_x-\beta(1-\beta)(h_x\eta+h\eta_x)+(\beta^2-1)\eta\eta_x\Big\}\Big\{u(u_{xx}+v_{yx})\\
&\qquad\qquad+v(u_{xy}+v_{yy})\Big\}\\
&\quad+\Big\{\frac{1}{2}(1-\beta)^2h^2-\beta(1-\beta)h\eta+\frac{1}{2}(\beta^2-1)\eta^2\Big\}\Big\{u_x(u_{xx}+v_{yx})+u(u_{xx}+v_{yx})_x\\
&\qquad\qquad+v_x(u_{xy}+v_{yy})+v(u_{xy}+v_{yy})_x\Big\}\\
&\quad+\Big\{(hu)_x+(hv)_y+\eta(u_x+v_y)\Big\}\Big\{(hu)_{xx}+(hv)_{yx}+\eta_x(u_x+v_y)+\eta(u_{xx}+v_{yx})\Big\}
\end{aligned}
\tag{119}
$$

$$
\begin{aligned}
V_2 &= (\beta-1)(h+\eta)_y\Big\{u\Big[(hu)_{xx}+(hv)_{yx}\Big]+v\Big[(hu)_{xy}+(hv)_{yy}\Big]\Big\}\\
&\quad+(\beta-1)(h+\eta)\Big\{u\Big[(hu)_{xx}+(hv)_{yx}\Big]+v\Big[(hu)_{xy}+(hv)_{yy}\Big]\Big\}_y\\
&\quad+\Big\{\frac{1}{2}(1-\beta)^2h^2-\beta(1-\beta)h\eta+\frac{1}{2}(\beta^2-1)\eta^2\Big\}_y\Big\{u(u_{xx}+v_{yx})+v(u_{xy}+v_{yy})\Big\}\\
&\quad+\Big\{\frac{1}{2}(1-\beta)^2h^2-\beta(1-\beta)h\eta+\frac{1}{2}(\beta^2-1)\eta^2\Big\}\Big\{u(u_{xx}+v_{yx})+v(u_{xy}+v_{yy})\Big\}_y\\
&\quad+\Big\{(hu)_x+(hv)_y+\eta(u_x+v_y)\Big\}\Big\{(hu)_x+(hv)_y+\eta(u_x+v_y)\Big\}_y\\
&= (\beta-1)(h_y+\eta_y)\Big\{u\Big[(hu)_{xx}+(hv)_{yx}\Big]+v\Big[(hu)_{xy}+(hv)_{yy}\Big]\Big\}\\
&\quad+(\beta-1)(h+\eta)\Big\{u_y\Big[(hu)_{xx}+(hv)_{yx}\Big]+u\Big[(hu)_{xx}+(hv)_{yx}\Big]_y\\
&\qquad\qquad+v_y\Big[(hu)_{xy}+(hv)_{yy}\Big]+v\Big[(hu)_{xy}+(hv)_{yy}\Big]_y\Big\}\\
&\quad+\Big\{(1-\beta)^2hh_y-\beta(1-\beta)(h_y\eta+h\eta_y)+(\beta^2-1)\eta\eta_y\Big\}\Big\{u(u_{xx}+v_{yx})\\
&\qquad\qquad+v(u_{xy}+v_{yy})\Big\}\\
&\quad+\Big\{\frac{1}{2}(1-\beta)^2h^2-\beta(1-\beta)h\eta+\frac{1}{2}(\beta^2-1)\eta^2\Big\}\Big\{u_y(u_{xx}+v_{yx})+u(u_{xx}+v_{yx})_y\\
&\qquad\qquad+v_y(u_{xy}+v_{yy})+v(u_{xy}+v_{yy})_y\Big\}\\
&\quad+\Big\{(hu)_x+(hv)_y+\eta(u_x+v_y)\Big\}\Big\{(hu)_{xy}+(hv)_{yy}+\eta_y(u_x+v_y)+\eta(u_{xy}+v_{yy})\Big\}
\end{aligned}
\tag{120}
$$