Texas Water Development Board

Contract No. 2006-483-031

Final Report: Implementation and testing of various turbulence models within the UTBEST3D shallow water code

PI: Clint Dawson (The University of Texas at Austin)
E-mail: clint@ices.utexas.edu
Voice: (512) 475-8625

Team Members:

Clint Dawson, Professor

Jennifer Proft, Research Scientist

Vadym Aizinger, Visiting Research Scientist

Contents

List of Figures

List of Tables

1 Executive Summary

The University of Texas Bay and Estuary 3D (UTBEST3D) simulator solves the shallow water equations using a recently developed discontinuous Galerkin (DG) finite element approach. The model uses a z-grid with general prismatic elements and has the capability to follow a sloped (piecewise linear) bottom bed. The method is based on the use of discontinuous, piecewise polynomial approximating functions for each primary variable, defined over each element. The advantages of the DG method include the ability to model flows at multiple scales, including resolution of long wave and advection-dominated flows, local (elementwise) mass conservation, and the ability to easily adapt the mesh and polynomial order locally. The primary focus of this project was the incorporation of various turbulence closure models into the code and the preliminary testing of these models. Turbulence models which were implemented include two algebraic (zeroth order) models, a one equation model, and various two equation models. Our preliminary findings indicate that for a simple channel flow problem, model results were found to be in good agreement with results presented in the literature for a similar test case. Results using different orders of approximating spaces, from piecewise constant to piecewise quadratic, indicate that piecewise quadratic approximations of kinetic energy are superior at resolving the boundary layer at the sea bed. Tests for a more difficult test problem in the Bight of Abaco demonstrate that most of the turbulence models which were implemented give similar results, including algebraic, one equation and two equation models.

2 Project Description

2.1 Introduction

In this report, we describe a discontinuous Galerkin (DG) based finite element model for the three-dimensional shallow water equations, UTBEST3D (University of Texas Bay and Estuary Simulator), which has been developed at UT Austin by the investigators.

Despite many recent advances in the development of large-scale simulators for modeling circulation in oceanic to continental shelf, coastal and estuarine environments, the search is still on for methods which are locally mass conservative, can handle very general types of elements, and are stable and accurate under highly varying flow regimes. Recently developed algorithms such as the DG method are of great interest within the surface water modeling community. DG methods are promising because of their flexibility with regard to geometrically complex elements, use of shock-capturing numerical fluxes, adaptivity in polynomial order, ability to handle nonconforming grids, and local conservation properties; see [8] for a historical overview of DG methods. In [2, 7], we investigated DG and related finite volume methods for the solution of the two-dimensional shallow water equations. Viscosity (second-order derivative) terms are handled in this method through the so-called local discontinuous Galerkin (LDG) framework [10], which employs a mixed formulation. Application of the methodology to three-dimensional shallow water models was first described in [15]. The 3D formulation is not a straightforward extension of the two-dimensional algorithm. In particular, it uses a special form of the continuity equation for the free surface elevation and requires postprocessing the elevation solution to smooth the computational domain. While the numerical results given in [15] are promising, further work is needed to ascertain the accuracy, stability and efficiency of this methodology, and to enhance the capabilities of the model to handle baroclinic and turbulent flows. During this project year, we have included and tested algebraic, one and two equation turbulence models in barotropic mode, and have begun testing these models under baroclinic conditions.

The rest of this report is organized as follows. In Sections 2.2-2.4, we discuss the model equations, boundary conditions, and our assumptions about the time-varying computational domain. In Sections 2.5-2.7, we formulate the semi-discrete DG finite element method used to approximate the flow equations. In Sections 2.8-2.13, details of the formulation are discussed, including the implementation of the Riemann solvers used to compute the fluxes on each element face, and how boundary conditions are implemented in the method. In Sections 2.14 and 2.15, we discuss the species and turbulent transport equations which are implemented in the model and how these equations are solved. In Section 2.16, the approximating spaces and time-stepping methods are discussed. Finally, in Section 2.17, numerical results comparing the various turbulence models are presented.

2.2 Model and assumptions

For $\mathbf{a}, \mathbf{b} \in \mathbb{R}^d, \mathbf{c} \in \mathbb{R}^e$, we denote by $\mathbf{ac}$ the tensor-product of $\mathbf{a}$ and $\mathbf{c}$ and by $\mathbf{a} \cdot \mathbf{b}$ the dot-product of $\mathbf{a}$ and $\mathbf{b}$.

Let $\Omega(t) \subset \mathbb{R}^3$ be our time-dependent domain. We assume the top boundary of the domain $\partial\Omega_{top}(t)$ is the only moving boundary. The bottom $\partial\Omega_{bot}$ and lateral $\partial\Omega_D(t)$ boundaries are assumed to be fixed (though the height of the lateral boundaries can vary with time according to the movements of the free surface). We also require the lateral boundaries to be strictly vertical (see Figure 1). The last requirement is only needed to assure that the horizontal cross-section of the domain $\Omega(t)$ (denoted by Ω_{xy}) doesn't change with time.

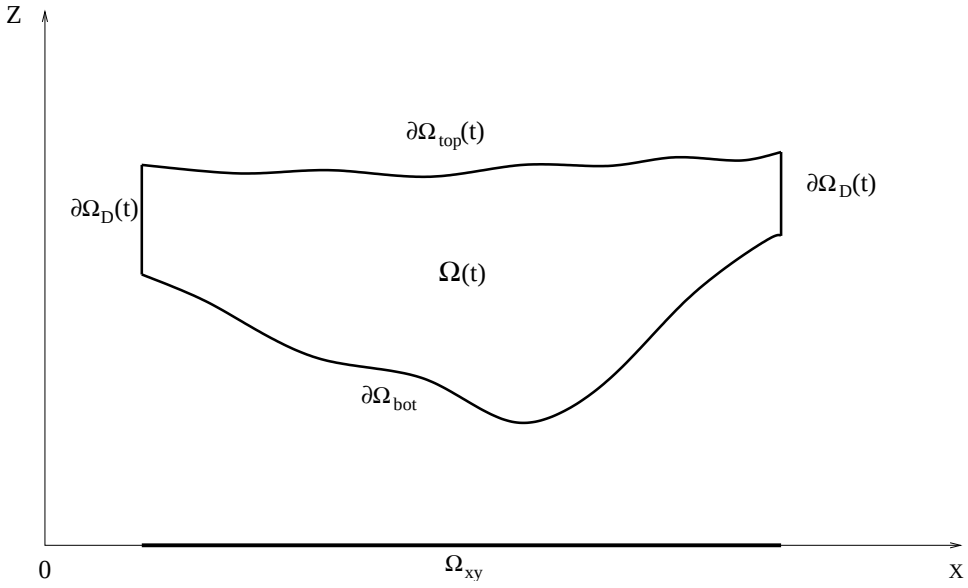


Figure 1: Vertical cross-section of the computational domain $\Omega(t)$.

Keeping in line with the specific anisotropy of $\Omega(t)$ we construct our 3D finite element mesh by extending a 2D triangular mesh of Ω_{xy} in the vertical direction, thus producing a 3D mesh of $\Omega(t)$ that consists of one or more layers of prismatic elements. In order to better reproduce the bathymetry and the free surface elevation of the computational domain we do not require top and bottom faces of prisms to be parallel to the xy-plane, although the lateral faces are required to be strictly vertical.

We introduce the following sets of element and face indices:

- I_e - set of element indices for prismatic elements in $\Omega(t)$;
- $I_{e,xy}$ - set of element indices for triangular elements in Ω_{xy};
- I_f - set of face indices for prism faces;
- $I_{int} \subset I_f$ - set of interior face indices;
- $I_{ext} \subset I_f$ - set of exterior face indices;
- $I_{lat} \subset I_{int}$ - set of interior lateral face indices;
- $I_{horiz} \subset I_{int}$ - set of interior horizontal face indices;
- $I_D \subset I_{ext}$ - set of exterior lateral face indices;
- $I_{top} \subset I_{ext}$ - set of indices for exterior faces on the top boundary,
- $I_{bot} \subset I_{ext}$ - set of indices for exterior faces on the bottom boundary.

For a point $(x, y) \in \Omega_{xy}$ we denote by $z_b(x, y)$ the value of the z-coordinate at the bottom of the domain and by $\xi_s(t, x, y)$ at the top. A key feature of our 3D LDG model is the fact that all primary variables, including the free surface elevation, are discretized using discontinuous polynomial spaces. As a result, computed values of the free surface elevation may have jumps across inter-element boundaries. If our finite element grids were to follow exactly the computed free surface elevation field this would cause the elements in the surface layer to have mismatching lateral faces (staircase boundary). We avoid this difficulty by employing a globally continuous free surface approximation that is obtained from the computed values of the free surface elevation ξ with the help of a smoothing algorithm (see Figure 2). Thus H is the computed height of the water column, and H_s is the the postprocessed height.

It must be noted here that solely the computational mesh is modified by the smoothing algorithm whereas the computed (discontinuous) approximations to all unknowns, including the free surface elevation, are left unchanged. This approach preserves the local conservation property of the LDG method and is essential for our algorithm's stability.

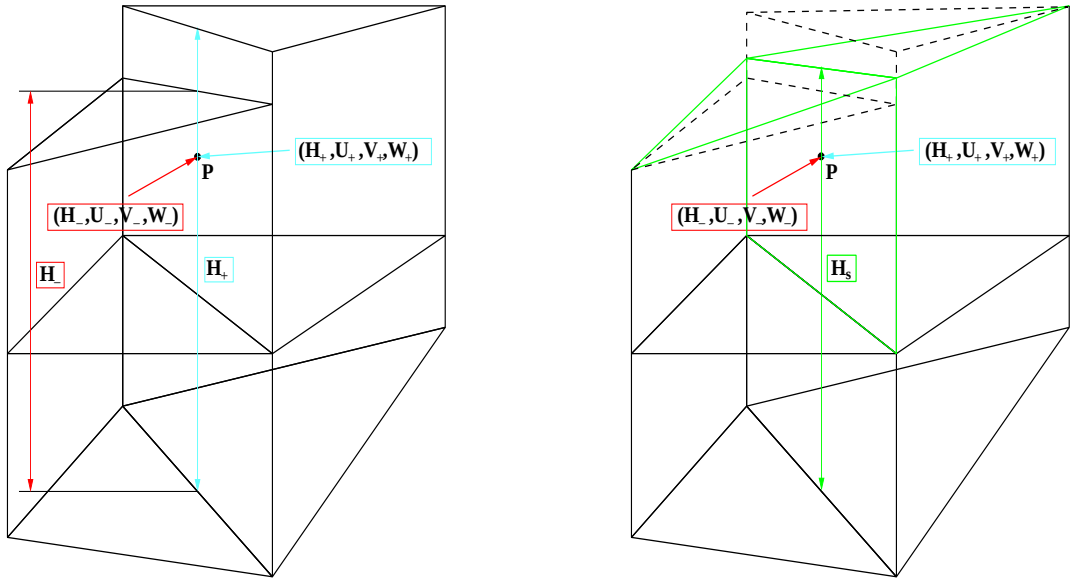
2.3 System of 3D shallow water equations

The momentum equations in conservative form (assuming constant density) are given by [24]

$$\partial_t \mathbf{u}_{xy} \; + \; \nabla \cdot (\mathbf{u}_{xy}\mathbf{u} \; - \; \mathcal{D}\nabla \mathbf{u}_{xy}) \; + \; g\nabla_{xy}\xi \; - \; f_c \mathbf{k} \times \mathbf{u}_{xy} \; = \; \mathbf{F}, \tag{1}$$

where the wind stress, the atmospheric pressure gradient, and the tidal potential are combined into a body force term $\mathbf{F}$, $\nabla_{xy} = (\partial_x, \partial_y)$, ξ is the value of the z coordinate at the free surface, $\mathbf{u} = (u, v, w)$ is the velocity vector, $\mathbf{u}_{xy} = (u, v)$ is the vector of horizontal velocity components, f_c is the Coriolis coefficient, $\mathbf{k} = (0, 0, 1)$ is a unit vertical vector, g is acceleration due to gravity, and $\mathcal{D}$ is the tensor of eddy viscosity coefficients defined as follows:

$$\mathcal{D} = \begin{pmatrix} D_u & 0 \\ 0 & D_v \end{pmatrix}, \tag{2}$$



a) Before surface mesh smoothing b) After surface mesh smoothing

Figure 2: Illustration of mesh smoothing.

with D_u, D_v 3×3 symmetric positive-definite matrices, and $\mathcal{D}\nabla \mathbf{u}_{xy} = \begin{pmatrix} D_u \nabla u \\ D_v \nabla v \end{pmatrix}$. In particular,

$$D_u \;=\; D_v \;=\; \begin{pmatrix} A_x & 0 & 0 \\ 0 & A_y & 0 \\ 0 & 0 & \nu_t \end{pmatrix},$$

where A_x, A_y are the horizontal and ν_t is the vertical eddy viscosity coefficient.

The continuity equation is

$$\nabla \cdot \mathbf{u} \;=\; 0. \tag{3}$$

2.4 Boundary conditions

The following boundary conditions are specified for the system:

- At the bottom boundary $\partial\Omega_{bot}$, we have no normal flow

$$\mathbf{u}(z_b) \cdot \mathbf{n} = 0 \tag{4}$$

and the quadratic slip condition for the horizontal velocity components

$$\nu_t \frac{\partial u}{\partial n}(z_b) = C_f \sqrt{u^2(z_b) + v^2(z_b)}\, u(z_b), \tag{5}$$

$$\nu_t \frac{\partial v}{\partial n}(z_b) = C_f \sqrt{u^2(z_b) + v^2(z_b)}\, v(z_b), \tag{6}$$

where $\mathbf{n} = (n_x, n_y, n_z)$ is an exterior unit normal to the boundary.

- The free surface boundary conditions have the form

$$\partial_t \xi \;+\; u(\xi)\; \partial_x \xi \;+\; v(\xi)\; \partial_y \xi \;-\; w(\xi) \;=\; 0, \tag{7}$$

$$\nabla u(\xi) \cdot \mathbf{n} \;=\; \nabla v(\xi) \cdot \mathbf{n} \;=\; 0. \tag{8}$$

On the lateral boundaries, we consider several common types of boundary conditions:

- Land boundary: No normal flow
$$u_{\mathbf{n}} = \mathbf{u} \cdot \mathbf{n} = 0, \tag{9}$$
and zero shear stress
$$\nabla u_{\boldsymbol{\tau}} \cdot \mathbf{n} \ = \ 0, \tag{10}$$
where $\boldsymbol{\tau}$ and $\mathbf{n}$ denote a unit tangential and a unit exterior normal vectors to the boundary, correspondingly.

- Open sea boundary: Zero normal derivative of the horizontal velocity components
$$\nabla u \cdot \mathbf{n} \ = \ \nabla v \cdot \mathbf{n} \ = \ 0, \tag{11}$$
and prescribed surface elevation $\xi_{os}(x, y, t)$
$$\xi = \xi_{os}(x, y, t). \tag{12}$$

- River boundary: Prescribed velocities
$$\mathbf{u} = \mathbf{u}_r, \tag{13}$$
and prescribed surface elevation
$$\xi = \xi_r. \tag{14}$$

- Radiation boundary: Zero normal derivative of the horizontal velocity components
$$\nabla u \cdot \mathbf{n} \ = \ \nabla v \cdot \mathbf{n} \ = \ 0. \tag{15}$$

Analytically, the free surface elevation can be computed from (7). However, a computationally more robust method [24] is obtained by integrating continuity equation (3) over the total height of the water column. Taking into account boundary conditions (4) – (7) at the bottom and top boundaries we arrive at a 2D equation for the free surface elevation commonly called the primitive continuity equation (PCE),
$$\partial_t \xi \ + \ \partial_x \int_{z_b}^{\xi} u dz \ + \ \partial_y \int_{z_b}^{\xi} v dz \ = \ 0. \tag{16}$$

2.5 LDG discretization

Let us denote $h = \xi - z_b$. Then we can rewrite the mass and momentum conservation equations (16), (1) in the following compact form:
$$\partial_t h \ + \ \nabla_{xy} \cdot \mathbf{C}_h(\mathbf{c}) \ = \ 0, \tag{17}$$
$$\partial_t \mathbf{u}_{xy} \ + \ \nabla \cdot (\mathcal{C}_{\mathbf{u}}(\mathbf{c}) \ - \ \mathcal{D} \nabla \mathbf{u}_{xy}) \ = \ \mathbf{M}(\mathbf{c}), \tag{18}$$
where $\mathbf{c} = (h, u, v, w)$ is the vector of state variables,
$$\mathbf{C}_h(\mathbf{c}) = \begin{pmatrix} \int_{z_b}^{\xi} u \ dz \\ \int_{z_b}^{\xi} v \ dz \end{pmatrix}, \quad \mathcal{C}_{\mathbf{u}}(\mathbf{c}) = \begin{pmatrix} \mathbf{C}_u(\mathbf{c}) \\ \mathbf{C}_v(\mathbf{c}) \end{pmatrix} \ = \ \begin{pmatrix} u^2 + gh & uv & uw \\ uv & v^2 + gh & vw \end{pmatrix},$$
$$\mathbf{M}(\mathbf{c}) \ = \ \begin{pmatrix} F_x - g\partial_x z_b + f_c v \\ F_y - g\partial_y z_b - f_c u \end{pmatrix}.$$

2.6 Weak formulation

First, let us introduce an auxiliary variable ϱ and rewrite the second-order momentum equations (18) in mixed form

$$\partial_t \mathbf{u}_{xy} + \nabla \cdot (\mathcal{C}_{\mathbf{u}}(\mathbf{c}) + \sqrt{\mathcal{D}}\varrho) = \mathbf{M}(\mathbf{c}), \tag{19}$$

$$\varrho = -\sqrt{\mathcal{D}}\nabla \mathbf{u}_{xy}. \tag{20}$$

Let $\mathcal{T}_{\Delta x}$ be a partition of the domain $\Omega(t) \subset \mathbf{R}^d,\ d = 3$ into prisms with strictly vertical lateral (side) faces, and let $\Omega_e(t) \in \mathcal{T}_{\Delta x}$. To obtain a weak form of the momentum equations we multiply (19), (20) by arbitrary smooth test functions $\boldsymbol{\phi}$ and Ψ, integrate them on each element $\Omega_e(t) \in \mathcal{T}_{\Delta x}$, and integrate by parts obtaining

$$\begin{aligned}
(\partial_t \mathbf{u}_{xy}, \boldsymbol{\phi})_{\Omega_e(t)} \ &+ \ \left\langle (\mathcal{C}_{\mathbf{u}}(\mathbf{c}) + \sqrt{\mathcal{D}}\varrho) \cdot \mathbf{n}_e, \boldsymbol{\phi} \right\rangle_{\partial\Omega_e(t)} \\
&- \ \left((\mathcal{C}_{\mathbf{u}}(\mathbf{c}) + \sqrt{\mathcal{D}}\varrho) \cdot \nabla, \boldsymbol{\phi} \right)_{\Omega_e(t)} \ = \ (\mathbf{M}(\mathbf{c}), \boldsymbol{\phi})_{\Omega_e(t)}\,, \\
(\varrho,\, \Psi)_{\Omega_e(t)} \ &= \ -\left\langle \mathbf{u}_{xy}\ (\sqrt{\mathcal{D}}\ \mathbf{n}_e),\, \Psi \right\rangle_{\partial\Omega_e(t)} \ + \ \left(\mathbf{u}_{xy}\ (\sqrt{\mathcal{D}}\ \nabla),\, \Psi \right)_{\Omega_e(t)},
\end{aligned}$$

where $\mathbf{n}_e$ is a unit exterior normal to $\partial\Omega_e(t)$. This weak formulation is well defined for $\mathbf{u}_{xy}(t,x,y,z) \in H^1(0,T;\ V^{d-1})$; $\boldsymbol{\phi}(x,y,z) \in V^{d-1}$; $\varrho(t,x,y,z) \in V^{d-1\times d},\ \forall t \in [0,T]$; and $\Psi(x,y,z) \in V^{d-1\times d}$, where

$$V \ = \ L^2(\Omega(t)) \ \cap \ \{u : u_{|_{\Omega_e(t)}} \in H^1(\Omega_e(t)),\ \forall \Omega_e(t) \in \mathcal{T}_{\Delta x}\}. \tag{21}$$

Fixing the direction of the unit normal $\mathbf{n}$ on the interior faces we can sum over all elements $\Omega_e(t) \in \mathcal{T}_{\Delta x}$ and obtain a weak form of the momentum equations

$$\begin{aligned}
&\sum_{e\in I_e} (\partial_t \mathbf{u}_{xy}, \boldsymbol{\phi})_{\Omega_e(t)} \ + \ \sum_{i\in I_{int}} \left\langle (\mathcal{C}_{\mathbf{u}}(\mathbf{c}) + \sqrt{\mathcal{D}}\varrho) \cdot \mathbf{n}, [\boldsymbol{\phi}] \right\rangle_{\gamma_i(t)} \\
&\quad + \ \sum_{i\in I_{ext}} \left\langle (\mathcal{C}_{\mathbf{u}}(\mathbf{c}) + \sqrt{\mathcal{D}}\varrho) \cdot \mathbf{n}, \boldsymbol{\phi} \right\rangle_{\gamma_i(t)} - \sum_{e\in I_e} \left((\mathcal{C}_{\mathbf{u}}(\mathbf{c}) + \sqrt{\mathcal{D}}\varrho) \cdot \nabla, \boldsymbol{\phi} \right)_{\Omega_e(t)} \\
&\quad = \ \sum_{e\in I_e} (\mathbf{M}(\mathbf{c}), \boldsymbol{\phi})_{\Omega_e(t)}\,,
\end{aligned} \tag{22}$$

$$\begin{aligned}
\sum_{e\in I_e} (\varrho,\, \Psi)_{\Omega_e(t)} \ &= \ -\sum_{i\in I_{int}} \left\langle \mathbf{u}_{xy}\ (\sqrt{\mathcal{D}}\ \mathbf{n}), [\,\Psi] \right\rangle_{\gamma_i(t)} \\
&- \ \sum_{i\in I_{ext}} \left\langle \mathbf{u}_{xy}\ (\sqrt{\mathcal{D}}\ \mathbf{n}),\, \Psi \right\rangle_{\gamma_i(t)} \ + \ \sum_{e\in I_e} \left(\mathbf{u}_{xy}\ (\sqrt{\mathcal{D}}\ \nabla),\, \Psi \right)_{\Omega_e(t)}.
\end{aligned} \tag{23}$$

Discretization of the primitive continuity equation is done in a similar way. Let us denote by Π the orthogonal projection operator from the xyz-space onto the xy-plane ($\Pi(x,y,z) = (x,y)$), and let $\Omega_{e,xy} = \Pi\Omega_e(t)$. Since the free surface is the only moving boundary of $\Omega(t)$, $\Omega_{e,xy}$ are not time-dependent. We multiply (17) by an arbitrary smooth test function $\delta = \delta(x,y)$, integrate it over $\Omega_{e,xy}$, and integrate by parts. Then the mass balance in the water column corresponding to $\Omega_{e,xy}$ can be expressed as

$$(\partial_t h, \delta)_{\Omega_{e,xy}} \ + \ \langle \mathbf{C}_h(\mathbf{c}) \cdot \mathbf{n}, \delta \rangle_{\partial\Omega_{e,xy}} \ - \ (\mathbf{C}_h(\mathbf{c}) \cdot \nabla_{xy}, \delta)_{\Omega_{e,xy}} \ = \ 0.$$

Recalling that $\mathbf{C}_h = \left(\int_{z_b}^{\xi} u dz, \int_{z_b}^{\xi} v dz \right)$ we can rewrite the equation above in a special 2D/3D form

$$(\partial_t h, \delta)_{\Omega_{e,xy}} + \sum_{\Pi\Omega_e(t)=\Omega_{e,xy}} \langle \mathbf{u}_{xy} \cdot \mathbf{n}_{xy}, \delta \rangle_{\partial\Omega_{e,lat}(t)} - \sum_{\Pi\Omega_e(t)=\Omega_{e,xy}} (\mathbf{u}_{xy} \cdot \nabla_{xy}, \delta)_{\Omega_e(t)} = 0,$$

where $\mathbf{n}_{xy} = (n_x, n_y)$, $\partial\Omega_{e,lat}(t)$ denotes the lateral boundary faces of prism $\Omega_e(t)$, and the summation is over the set of 3D elements in the water column corresponding to $\Omega_{e,xy}$. Note that the expression above is well defined for any $\delta(x,y) \in \mathcal{H} \stackrel{\text{def}}{=} L^2(\Omega_{xy}) \cap \{h : h_{|_{\Pi\Omega_e(t)}} \in H^1(\Pi\Omega_e(t)), \forall \Omega_e(t) \in \mathcal{T}_{\Delta x}\}$ and $h(t,x,y) \in H^1(0,T;\mathcal{H})$. Summing over all elements $\Omega_e(t) \in \mathcal{T}_{\Delta x}$ we obtain a weak form of the PCE

$$\sum_{e\in I_{e,xy}} (\partial_t h, \delta)_{\Omega_{e,xy}} \; + \sum_{i\in I_{lat}} \langle \mathbf{u}_{xy}\cdot\mathbf{n}_{xy}, [\delta]\rangle_{\gamma_i(t)} + \sum_{i\in I_D} \langle \mathbf{u}_{xy}\cdot\mathbf{n}_{xy}, \delta\rangle_{\gamma_i(t)} \; - \sum_{e\in I_e} (\mathbf{u}_{xy}\cdot\nabla_{xy}, \delta)_{\Omega_e(t)} \; = \; 0. \tag{24}$$

To discretize the continuity equation we multiply (3) by an arbitrary smooth test function σ, integrate it over $\Omega_e(t)$, and integrate by parts obtaining

$$\langle \mathbf{u}\cdot\mathbf{n}, \sigma\rangle_{\partial\Omega_e(t)} \; - \; (\mathbf{u}\cdot\nabla, \sigma)_{\Omega_e(t)} \; = \; 0.$$

Summing over all elements $\Omega_e(t) \in \mathcal{T}_{\Delta x}$ we get a weak form of the continuity equation

$$\sum_{i\in I_{int}} \langle \mathbf{u}\cdot\mathbf{n}, [\sigma]\rangle_{\gamma_i(t)} \; + \sum_{i\in I_{ext}} \langle \mathbf{u}\cdot\mathbf{n}, \sigma\rangle_{\gamma_i(t)} \; - \sum_{e\in I_e} (\mathbf{u}\cdot\nabla, \sigma)_{\Omega_e(t)} \; = \; 0. \tag{25}$$

2.7 Semi-discrete formulation

Next, we seek to approximate

$$(h(t,\cdot), \mathbf{u}_{xy}(t,\cdot), w(t,\cdot), \mathcal{Q}(t,\cdot)),$$

a solution to problem (22) – (25), with a function

$$(H(t,\cdot), \mathbf{U}_{xy}(t,\cdot), W(t,\cdot), \mathcal{Q}(t,\cdot)) \in \mathcal{H}_\Delta \times U_\Delta \times W_\Delta \times Z_\Delta,$$

where $\mathcal{H}_\Delta \subset \mathcal{H}$, $U_\Delta \subset V^{d-1}$, $W_\Delta \subset V$, and $Z_\Delta \subset V^{d-1\times d}$ are some finite-dimensional subspaces. For this purpose, we can use the weak formulation with one important modification. Since the approximation spaces utilized in the DG methods do not guarantee continuity across the inter-element boundaries, all integrands in the integrals over interior faces have to be replaced by suitably chosen numerical fluxes that preserve consistency and stability of the method. A semi-discrete finite element solution $(H(t,\cdot), \mathbf{U}_{xy}(t,\cdot), W(t,\cdot), \mathcal{Q}(t,\cdot))$ is obtained by requiring that for any $t \in [0,T]$, for all $\Omega_e(t) \in \mathcal{T}_{\Delta x}$, and for all $(\delta, \boldsymbol{\phi}, \Psi, \omega) \in \mathcal{H}_\Delta \times U_\Delta \times W_\Delta \times Z_\Delta$ the following holds:

$$\sum_{e\in I_{e,xy}} (\partial_t H, \delta)_{\Omega_{e,xy}} \; + \sum_{i\in I_{lat}} \left\langle \frac{\hat{C}_{h,\mathbf{n}}(\mathbf{C}_-, \mathbf{C}_+)}{H_s}, [\delta] \right\rangle_{\gamma_i(t)} + \sum_{i\in I_D} \left\langle \frac{\hat{C}_{h,\mathbf{n}}(\mathbf{C}_-, \mathbf{C}_+)}{H_s}, \delta \right\rangle_{\gamma_i(t)} - \sum_{e\in I_e} \left(\frac{\mathbf{U}_{xy} H}{H_s}\cdot\nabla_{xy}, \delta \right)_{\Omega_e(t)} = \; 0, \tag{26}$$

$$\sum_{e\in I_e} (\partial_t \mathbf{U}_{xy}, \boldsymbol{\phi})_{\Omega_e(t)} \; + \sum_{i\in I_{int}} \left\langle \hat{\mathbf{C}}_{\mathbf{u},\mathbf{n}}(\mathbf{C}_-, \mathbf{C}_+) + \sqrt{\mathcal{D}}\hat{\mathcal{Q}}\cdot\mathbf{n}, [\boldsymbol{\phi}] \right\rangle_{\gamma_i(t)} + \sum_{i\in I_{ext}} \left\langle \mathbf{C}_{\mathbf{u},\mathbf{n}}(\mathbf{C}_-, \mathbf{C}_+) + \sqrt{\mathcal{D}}\mathcal{Q}\cdot\mathbf{n}, \boldsymbol{\phi} \right\rangle_{\gamma_i(t)} - \sum_{e\in I_e} \left((\mathcal{C}_{\mathbf{u}}(\mathbf{C}) + \sqrt{\mathcal{D}}\mathcal{Q})\cdot\nabla, \boldsymbol{\phi} \right)_{\Omega_e(t)} = \sum_{e\in I_e} (\mathbf{M}(\mathbf{C}), \boldsymbol{\phi})_{\Omega_e(t)}\,, \tag{27}$$

$$\sum_{e\in I_e}(\mathcal{Q},\Psi)_{\Omega_e(t)} \;=\; -\sum_{i\in I_{int}}\Big\langle \hat{\mathbf{U}}_{xy}\,(\sqrt{\mathcal{D}}\,\mathbf{n}),[\Psi]\Big\rangle_{\gamma_i(t)}$$
$$-\sum_{i\in I_{ext}}\Big\langle \mathbf{U}_{xy}\,(\sqrt{\mathcal{D}}\,\mathbf{n}),\Psi\Big\rangle_{\gamma_i(t)} + \sum_{e\in I_e}\Big(\mathbf{U}_{xy}\,(\sqrt{\mathcal{D}}\,\nabla),\Psi\Big)_{\Omega_e(t)}, \tag{28}$$

$$\sum_{i\in I_{int}}\Big\langle \hat{\mathbf{U}}\cdot\mathbf{n},[\sigma]\Big\rangle_{\gamma_i(t)} + \sum_{i\in I_{ext}}\Big\langle \hat{\mathbf{U}}\cdot\mathbf{n},\sigma\Big\rangle_{\gamma_i(t)} - \sum_{e\in I_e}(\mathbf{U}\cdot\nabla,\sigma)_{\Omega_e(t)} \;=\; 0, \tag{29}$$

where $(\hat{C}_{h,\mathbf{n}}(\mathbf{C}_-,\mathbf{C}_+),\hat{\mathbf{C}}_{\mathbf{u},\mathbf{n}}(\mathbf{C}_-,\mathbf{C}_+))$ is an approximation to the nonlinear boundary flux $(\mathbf{U}_{xy}H\cdot\mathbf{n}_{xy},\mathcal{C}_{\mathbf{u}}\cdot\mathbf{n})$ that depends on the values of the state variables $\mathbf{C}_-,\mathbf{C}_+$ on both sides of the discontinuity. The stability of the method depends to a large degree on this approximation satisfying certain entropy conditions. We will address this issue in some detail in Section 2.8. The restrictions on the linear boundary fluxes $\hat{\mathbf{U}}_{xy},\hat{\mathcal{Q}}$ are much less severe. They can be set equal to arithmetic averages of the values of the corresponding variables on both sides of the discontinuity or some other consistent numerical flux.

Remark 1: The continuity equation is, unlike the mass and momentum conservation equations, not time-dependent, its main role being computation of the vertical velocity component W to maintain a divergence-free velocity field. Regarding (29) and the kinematic boundary condition at the bottom (4) as an initial value problem for W, we can compute W element-by-element in each water column starting at the bottom and using the solution from the element below as an initial condition.

The choice of the boundary flux $\hat{\mathbf{U}}$ in (29) also merits a special mention. On the interior lateral faces it should be set equal to $\frac{\hat{C}_{h,\mathbf{n}}(\mathbf{C}_-,\mathbf{C}_+)}{H_s}$, exactly as it is in the discrete form of the primitive continuity equation (26), in order to preserve the local mass conservation properties of our numerical scheme. On the interior horizontal faces $\hat{\mathbf{U}}$ can be approximated by the average or upwind values of the corresponding variable.

2.8 Riemann solvers for the 3D problem

In this section, we will show how to utilize the boundary flux formulation shown in (26) in a Riemann solver. Since we managed to transform all boundary integrals into a 3D form, we end up with a well-posed Riemann problem that can be solved to produce a numerical boundary flux on the lateral faces satisfying the entropy condition (for a discussion of different entropy conditions see, e.g., [19]).

Let $\mathbf{P}$ be a point on Γ, where Γ is an interior lateral boundary face in the 3D mesh (see Figure 2). Let $\mathbf{n}=(n_x,n_y,n_z)$ be a unit normal to Γ at $\mathbf{P}$. Note that n_z is equal to 0 since all lateral faces are strictly vertical. We denote by $\mathbf{c}=(h,u,v)$ the vector of state variables. Note that we don't include the vertical velocity component in $\mathbf{c}$ because w only enters the normal boundary flux when multiplied by n_z. Then we define the left and right states $\mathbf{c}_L,\mathbf{c}_R$ at $\mathbf{P}$ as $\mathbf{c}_L=\lim_{\varepsilon\to 0-}\mathbf{c}(\mathbf{P}+\varepsilon\mathbf{n}),\mathbf{c}_R=\lim_{\varepsilon\to 0+}\mathbf{c}(\mathbf{P}+\varepsilon\mathbf{n})$. Our task is to compute an entropy solution $\hat{\mathbf{C}}_{\mathbf{n}}(\mathbf{c}_L,\mathbf{c}_R)$ to the Riemann problem for the nonlinear boundary flux $\mathbf{C}_{\mathbf{n}}=(\mathbf{C}_h\cdot\mathbf{n},\mathbf{C}_u\cdot\mathbf{n},\mathbf{C}_v\cdot\mathbf{n})$ at $\mathbf{P}$ (see (26), (27)).

2.9 Riemann solver of Roe

In this solver, an approximation to the normal boundary flux is given by

$$\hat{\mathbf{C}}_{\mathbf{n}}(\mathbf{c}_L,\mathbf{c}_R)=\mathbf{C}_{\mathbf{n}}(\mathbf{c}_L)+\sum_{i=1}^{3}\alpha_i\hat{\lambda}_i^{-}\hat{\mathbf{r}}_i, \tag{30}$$

where $\hat{\lambda}_i$ are the eigenvalues and $\hat{\mathbf{r}}_i$ the corresponding eigenvectors of matrix $\hat{R}_{\mathbf{n}}(\mathbf{c}_L, \mathbf{c}_R)$ defined below, $x^- = min\{0, x\}$, and α_i are calculated from

$$\sum_{i=1}^{3} \alpha_i \hat{\mathbf{r}}_i = \mathbf{c}_R - \mathbf{c}_L. \tag{31}$$

The matrix $\hat{R}_{\mathbf{n}}(\mathbf{c}_L, \mathbf{c}_R)$ has to satisfy the following three conditions [21]:

(i) $\hat{R}_{\mathbf{n}}(\mathbf{c}_L, \mathbf{c}_R)(\mathbf{c}_R - \mathbf{c}_L) = \mathbf{C}_{\mathbf{n}}(\mathbf{c}_R) - \mathbf{C}_{\mathbf{n}}(\mathbf{c}_L)$;

(ii) $\hat{R}_{\mathbf{n}}(\mathbf{c}_L, \mathbf{c}_R)$ is diagonalizable with real eigenvalues;

(iii) $\hat{R}_{\mathbf{n}}(\mathbf{c}_L, \mathbf{c}_R) \to C'_{\mathbf{n}}(\mathbf{c})$ smoothly as $\mathbf{c}_L, \mathbf{c}_R \to \mathbf{c}$, where

$$C'_{\mathbf{n}}(\mathbf{c}) = \begin{pmatrix} un_x + vn_y & hn_x & hn_y \\ gn_x & 2un_x + vn_y & un_y \\ gn_y & vn_x & un_x + 2vn_y \end{pmatrix}. \tag{32}$$

We claim that setting $\hat{R}_{\mathbf{n}}(\mathbf{c}_L, \mathbf{c}_R)$ equal to $C'_{\mathbf{n}}(\overline{\mathbf{c}})$, where $\overline{\mathbf{c}} = \frac{1}{2}(\mathbf{c}_L + \mathbf{c}_R)$, satisfies conditions on $\hat{R}_{\mathbf{n}}$. Indeed, we obtain the following eigenvalues and eigenvectors for $C'_{\mathbf{n}}(\mathbf{c})$:

$$\begin{aligned} \lambda_1(\mathbf{c}) &= \tfrac{3}{2}u_{\mathbf{n}} - \tfrac{1}{2}a, \quad \mathbf{r}_1(\mathbf{c}) = \begin{pmatrix} h \\ u - \frac{n_x}{2}(u_{\mathbf{n}} + a) \\ v - \frac{n_y}{2}(u_{\mathbf{n}} + a) \end{pmatrix}; \\ \lambda_2(\mathbf{c}) &= \tfrac{3}{2}u_{\mathbf{n}}, \qquad \mathbf{r}_2(\mathbf{c}) = \begin{pmatrix} 0 \\ -n_y \\ n_x \end{pmatrix}; \\ \lambda_3(\mathbf{c}) &= \tfrac{3}{2}u_{\mathbf{n}} + \tfrac{1}{2}a, \quad \mathbf{r}_3(\mathbf{c}) = \begin{pmatrix} h \\ u - \frac{n_x}{2}(u_{\mathbf{n}} - a) \\ v - \frac{n_y}{2}(u_{\mathbf{n}} - a) \end{pmatrix}; \end{aligned} \tag{33}$$

where $u_{\mathbf{n}} = un_x + vn_y$ and $a = \sqrt{u_{\mathbf{n}}^2 + 4gh}$. Therefore, condition (ii) is satisfied. Clearly, $C'_{\mathbf{n}}(\overline{\mathbf{c}}) \to C'_{\mathbf{n}}(\mathbf{c})$ smoothly as $\mathbf{c}_L, \mathbf{c}_R \to \mathbf{c}$. Finally, the first condition can be verified by simply substituting the appropriate values in (i).

2.10 Entropy fix for Roe's solver

The Roe's solver described in the previous section is quite adequate for most problems involving shocks, but it might experience difficulties with certain types of rarefaction waves (sonic rarefactions). There are several ways to modify Roe's algorithm that can fix this problem. Here, we give an approach presented in [18].

Let λ_p be the pth eigenvalue in (33). Then we define

$$\mathbf{c}_{p,L} \;=\; \mathbf{c}_L \;+\; \sum_{i=1}^{p-1} \alpha_i \hat{\mathbf{r}}_i, \qquad \mathbf{c}_{p,R} \;=\; \mathbf{c}_L \;+\; \sum_{i=1}^{p} \alpha_i \hat{\mathbf{r}}_i, \tag{34}$$

and, similarly to (30), the normal boundary flux is computed as

$$\hat{\mathbf{C}}_{\mathbf{n}}(\mathbf{c}_L, \mathbf{c}_R) = \mathbf{C}_{\mathbf{n}}(\mathbf{c}_L) + \sum_{i=1}^{3} \alpha_i \tilde{\lambda}_i \hat{\mathbf{r}}_i, \tag{35}$$

where

$$\tilde{\lambda}_i \;=\; \lambda_i^-(\mathbf{c}_{p,L})\;\frac{\lambda_i^+(\mathbf{c}_{p,R})-\lambda_i(\overline{\mathbf{c}})}{\lambda_i^+(\mathbf{c}_{p,R})-\lambda_i^-(\mathbf{c}_{p,L})}, \tag{36}$$

with $x^- = min\{0,x\}$ and $x^+ = max\{0,x\}$.

2.11 Lax-Friedrichs solver

The simplest Riemann solver supported in our 3D simulator is the Lax-Friedrichs scheme. In this method, the normal boundary flux is approximated by

$$\hat{\mathbf{C}}_{\mathbf{n}}(\mathbf{c}_L,\mathbf{c}_R) \;=\; \frac{1}{2}(\mathbf{C}_{\mathbf{n}}(\mathbf{c}_L)+\mathbf{C}_{\mathbf{n}}(\mathbf{c}_R)) \;+\; \frac{1}{2}|\hat{\lambda}_{max}|(\mathbf{c}_L-\mathbf{c}_R), \tag{37}$$

where $\hat{\lambda}_{max}$ is the largest (in absolute value) eigenvalue of $C'_{\mathbf{n}}(\overline{\mathbf{c}})$ (see (33)).

2.12 HLL solver

In the HLL Riemann solver, proposed by Harten, Lax, and van Leer [17], an approximation to the nonlinear boundary flux is computed as

$$\hat{\mathbf{C}}_{\mathbf{n}}(\mathbf{c}_L,\mathbf{c}_R) \;=\; \begin{cases} \mathbf{C}_{\mathbf{n}}(\mathbf{c}_L) & \text{if } s_L \geq 0, \\ \frac{s_R\mathbf{C}_{\mathbf{n}}(\mathbf{c}_L)\;-\;s_L\mathbf{C}_{\mathbf{n}}(\mathbf{c}_R)\;+\;s_Ls_R(\mathbf{c}_R-\mathbf{c}_L)}{s_R-s_L} & \text{if } s_L \leq 0 \leq s_R, \\ \mathbf{C}_{\mathbf{n}}(\mathbf{c}_R) & \text{if } s_R \leq 0, \end{cases} \tag{38}$$

where we choose (see [23]) $s_L \;=\; min\{u_{\mathbf{n},L}-a_L, u^*_{\mathbf{n}}-a^*\}$, $s_R \;=\; max\{u_{\mathbf{n},R}+a_R, u^*_{\mathbf{n}}+a^*\}$ with $u_{\mathbf{n}} = un_x+vn_y$, $a=\sqrt{gh}$, and

$$u^* = \frac{1}{2}(u_{\mathbf{n},L}+u_{\mathbf{n},R})+a_L-a_R, \quad a^* = \frac{1}{2}(a_L+a_R)+\frac{1}{4}(u_{\mathbf{n},L}-u_{\mathbf{n},R}).$$

2.13 Treatment of boundary conditions

In this section, we will discuss specific implementation issues concerning the boundary conditions at the top and bottom domain boundaries (4), (6), (8) and the lateral boundary.

- Bottom: On the bottom boundary, we compute the value of ϱ from (6) and the value of W from (4), the latter using the computed value of $\mathbf{U}_{xy}$ at the sea bed.
- Free surface: At the free surface boundary, we set ϱ equal to 0 and take the velocity values from the interior.

The boundary conditions on the lateral boundaries are handled as follows.

- Land boundary: On the land boundary, we use zero boundary conditions for ϱ and set $\hat{\mathbf{U}}_{xy} = \mathbf{U}_{xy}-(\mathbf{U}_{xy}\cdot\mathbf{n})\mathbf{U}_{xy}$, where $\mathbf{U}_{xy}$ is the value of the LDG solution from the interior of the domain. In the Riemann solver, we define the reflected velocity vector $\mathbf{U}^R_{xy} = \mathbf{U}_{xy}-2(\mathbf{U}_{xy}\cdot\mathbf{n})\mathbf{U}_{xy}$ and solve a Riemann problem for the boundary flux $\hat{\mathbf{C}}_{\mathbf{n}}((H,\mathbf{U}_{xy}),(H,\mathbf{U}^R_{xy}))$.
- Open sea boundary: On the open sea boundary, we take zero boundary conditions for ϱ and set $\hat{\mathbf{U}}_{xy} = \mathbf{U}_{xy}$. In the Riemann solver, we compute the boundary flux $\hat{\mathbf{C}}_{\mathbf{n}}((H,\mathbf{U}_{xy}),(\xi_{os}-z_b,\mathbf{U}_{xy}))$

- River boundary: Here, we set $\hat{\mathbf{u}} = \mathbf{u}_r$, and in the Riemann solver compute the boundary flux $\hat{\mathbf{C}}_{\mathbf{n}}((H, \mathbf{U}_{xy}), (\xi_r - z_b, \mathbf{u}_r))$ The values of ϱ are taken from the interior.
- Radiation boundary: We set ϱ equal to 0 and take values from the interior for all other variables.

2.14 Species transport

Species transport equations for salinity and temperature are included in the model. Transport is described by advection-diffusion equations of the form

$$r_t + \nabla \cdot (\mathbf{u}r) - \nabla \cdot (K_r \nabla r) = f, \quad \Omega(t) \times (0, T),$$

where $r = S$ for salinity or $r = T$ for temperature, and $K_r = \begin{pmatrix} \tilde{A}_x & 0 & 0 \\ 0 & \tilde{A}_y & 0 \\ 0 & 0 & \nu_r \end{pmatrix}$ is a specified diffusion tensor. These equations must be supplemented with initial and boundary conditions. The LDG method is also applied to the solution of these equations. The handling of the diffusion terms is similar to what is described above for $\mathbf{u}_{xy}$, therefore to simplify the discussion we assume $K_r = 0$.

Denote by R a discontinuous approximation to the species concentration r. Multiplying (39) by a discontinuous test function κ, integrating by parts, and approximating $\mathbf{u}$ by $\mathbf{U}$, we obtain the semi-discrete method

$$\begin{aligned} &\sum_{e \in I_e} (\partial_t R, \kappa)_{\Omega_e(t)} \; + \sum_{i \in I_{int}} \left\langle R^{\uparrow} \; \hat{\mathbf{U}} \cdot \mathbf{n}, [\kappa] \right\rangle_{\gamma_i(t)} \\ &+ \sum_{i \in I_{ext}} \left\langle R^{\uparrow} \; \hat{\mathbf{U}} \cdot \mathbf{n}, \kappa \right\rangle_{\gamma_i(t)} - \sum_{e \in I_e} (R \; \mathbf{U} \cdot \nabla, \kappa)_{\Omega_e(t)} \; = \; 0. \end{aligned} \tag{39}$$

Here $\hat{\mathbf{U}}$ is computed using the same method as in (29), and $R^{\uparrow}$ is the upwind value of R, determined by the sign of $\hat{\mathbf{U}} \cdot \mathbf{n}$.

2.15 Turbulence

The issue of realistic modeling of the turbulent eddy viscosity and diffusivity terms is an active research topic. The vertical eddy visosity coefficient is a particularly important parameter if one aims to achieve good vertical resolution of the computational domain.

UTBEST3D provides vertical eddy viscosity models of various levels of computational and conceptual complexity. In order of increasing complexity these include a constant eddy viscosity coefficient, an algebraic (zeroth order) model, as well as one and two equation models.

- The simplest model amounts to explicitly specifiying diagonal entries to the tensor of eddy viscosity coefficients in (1).
- Two algebraic models implemented in UTBEST3D are due to Davies [14] and give good results at a reasonable computational cost in cases where accurate vertical resolution of flow is not important.

  In the first model the eddy viscosity and diffusivity coefficients are set equal to $C_t \frac{(\bar{u}^2 + \bar{v}^2)}{\omega_a}$, where $\bar{u}$ and $\bar{v}$ are depth averaged horizontal velocity components, $C_t = 2 \times 10^{-5}$ is a dimensionless coefficient, and ω_a a typical long wave frequency set to $10^{-4} s^{-1}$.

  Model two is very similar to model one, except that the eddy viscosity is assumed to be proportional to $H\sqrt{\bar{u}^2 + \bar{v}^2}$.

- The first order vertical eddy viscosity closure model solves a transport eqaution for the turbulent kinetic energy in addition to the mass, momentum, and species transport equations.

$$k_t + \nabla \cdot (\mathbf{u}k) - \frac{\partial}{\partial z}(\nu_k \frac{\partial}{\partial z}k) = \nu_t \left(\left(\frac{\partial u}{\partial z} \right)^2 + \left(\frac{\partial v}{\partial x} \right)^2 \right) + \nu_r \frac{g}{\rho_0} \frac{\partial \rho}{\partial z} - \epsilon, \tag{40}$$

  where ν_k is the vertical diffusion coefficient for k and $\epsilon = (C_\mu^0)^3 k^{\frac{3}{2}} l^{-1}$ is the dissipation rate of the turbulent kinetic energy. The turbulent mixing length, $l(z)$, is computed in this model algebraically and is set equal to $\kappa(z - z_b)\sqrt{\xi - z}$ (see Delft3D-Flow manual [16]). $C_\mu^0 = \sqrt{0.3}$ is a calibration constant, $\kappa = 0.4$ is the von Karman constant. Once k is computed one can obtain vertical eddy viscosity and diffusivity coefficients by $\nu_k = \nu_r = \nu_t = C_\mu k^{\frac{1}{2}} l$, where C_μ is a stability function that accounts for stratification effects and is set to 0.5774 for barotropic flow simulations.

  At the free surface and the sea bed Neumann boundary conditions are employed for k: $\nu_k \frac{\partial k}{\partial n} = 0$.

  (40) is discretized similarly to the species transport equation (39).

- The second order closure model implemented in UTBEST3D is based on the generic turbulence length scale model proposed by Warner et al [25]. The advantage of this formulation is the ability to switch between several two equation models, including $k-\epsilon$ and Mellor-Yamada, by changing a few constant parameters. In addition to the transport equation for k this model includes a second transport equation for a derived quantity ψ.

$$\begin{aligned} &\psi_t + \nabla \cdot (\mathbf{u}\psi) - \frac{\partial}{\partial z}(\nu_\psi \frac{\partial}{\partial z}\psi) \\ &\quad = \frac{\psi}{k} \left(C_1 \nu_t \left(\left(\frac{\partial u}{\partial z} \right)^2 + \left(\frac{\partial v}{\partial x} \right)^2 \right) + C_3 \nu_r \frac{g}{\rho_0} \frac{\partial \rho}{\partial z} - C_2 \epsilon F_{wall} \right), \end{aligned} \tag{41}$$

  where $\psi = (C_\mu^0)^p k^m l^n$ and C_3 is equal C_3^- for stably stratified flow and C_3^+ otherwise. Depending on the choice of p, m, and n we obtain different closure schemes. Values of the parameters for four popular two equation models are shown in Table 1. Discretization of (41) is also done similarly to (39).

This model uses Neumann boundary conditions for ψ at the free surface $\nu_\psi \frac{\partial \psi}{\partial n} = 0$ (in the case of no wind forcing at the free surface) and at the sea bed $\nu_\psi \frac{\partial \psi}{\partial n} = -n\nu_\psi (C_\mu^0)^p k^m \kappa l^{n-1}$. Here we assume $l = \kappa \frac{\Delta z_b}{2}$, where Δz_b is the height of the bottom-most element.

2.16 Approximating spaces and time-stepping

The approximations $H \in \mathcal{H}_\Delta$, $\mathbf{U}_{xy} \in U_\Delta$, $W \in W_\Delta$ and $\mathcal{Q} \in Z_\Delta$ are constructed from basis functions consisting of complete polynomials in (x, y, z) defined on each element. Currently the maximum degree polynomial one can choose is two, thus constant, linear and quadratic approximations are allowed. The polynomial degree of the spaces U_Δ and Z_Δ are assumed equal. However, if desired one can choose different orders of polynomials for H and W. For example, one might use piecewise linears for W and H and piecewise constants for $\mathbf{U}_{xy}$. In addition, any transported quantities are approximated using similar discontinuous piecewise polynomials of maximum degree two.

	Mellor-Yamada [20]	$k-\epsilon$ [6]	$k-\omega$ [27]	generic [22]
p	0	3	-1	2
m	1	1.5	0.5	2
n	1	-1	-1	2/3
ν_k	$\frac{\nu_t}{2.44}$	ν_t	$\frac{\nu_t}{2}$	$\frac{\nu_t}{0.8}$
ν_ψ	$\frac{\nu_t}{2.44}$	$\frac{\nu_t}{1.3}$	$\frac{\nu_t}{2}$	$\frac{\nu_t}{1.07}$
C_1	0.9	1.44	0.555	1
C_2	0.5	1.92	0.833	1.22
C_3^+	1	1	1	1
C_3^-	2.53	-0.52	-0.58	0.1
k_{min}	5e-6	7.6e-6	7.6e-6	7.6e-6
ψ_{min}	1e-8	1e-12	1e-12	1e-12
F_{wall}	$1+1.33\left(\frac{l}{z-z_b}\right)^2+0.25\left(\frac{l}{\xi_s-z}\right)^2$	1	1	1

Table 1: Generic turbulence closure model parameters.

After spatial discretization, one arrives at a system of ODEs

$$\mathbf{y}'(t) \;=\; \mathbf{L}_h(\mathbf{y}(t),\, t), \tag{42}$$

where $\mathbf{y}$ represents all degrees of freedom associated with the time-dependent state variables, and $\mathbf{L}_h$ stands for the LDG space discretization operator. The time-stepping method we use is based on an explicit TVD Runge-Kutta method given below, with the order of the method matching the highest order of the spatial discretization. For example, if the highest order is linear, then we integrate in time using a second-order Runge-Kutta method. At each stage of the Runge-Kutta method, the solution process consists of four steps:

1. Solve (29) for the vertical velocity component from the previous time stage. In order to preserve the local mass conservation properties of our LDG scheme we must compute W with the same boundary flux as in the discrete version of the primitive continuity equation (26). Thus, this step requires solution of the Riemann problem on interior lateral faces, the results of which can be then stored for use in 3.

2. Compute the values of auxiliary variable $\mathcal{Q}$ using (28). Since $\mathcal{Q}$ is discontinuous, the computation of $\mathcal{Q}$ is completely local to an element, and only involves the solution of element-wise systems of equations.

3. Compute species transport and turbulent closure scheme unknowns.

4. Compute $\mathbf{C}$ from (26), (27) using solutions to the Riemann problem on interior lateral faces obtained in step 1. The linear term $\hat{\mathcal{Q}}$ can be taken equal to the average of values of $\mathcal{Q}$ on both sides of the discontinuity.

5. Update (when desired) the position of the free surface and perform surface mesh smoothing. Update the geometry of prisms and faces in the surface layer.

2.16.1 The TVD Runge-Kutta time-stepping method

To solve the system of ODEs (42), the traditional Runge-Kutta methods offer a wide variety of explicit and implicit schemes of various order (see, e.g., [5]). For most problems with smooth

solutions, these methods can be utilized in the time-stepping routine without any reservations. However, for problems with discontinuities or very steep gradients, numerical solutions obtained using traditional Runge-Kutta schemes may suffer from spurious oscillations. This was the principal motivation behind the total variation diminishing (TVD) Runge-Kutta methods introduced by Cockburn and Shu in [9] – [13]. These schemes – denoted $RK\Lambda\Pi P$ – can capture discontinuities without or with essentially dampened oscillations.

The main idea of a $RK\Lambda\Pi P$ method is, first, to reformulate the explicit Runge-Kutta scheme in some suitable form and then to perform, where needed, a limiting procedure on the degrees of freedom corresponding to the higher-order (linear, quadratic, etc) basis functions.

The explicit Runge-Kutta scheme used in $RK\Lambda\Pi P$ methods can be written as follows:

$$\begin{aligned}
\mathbf{y}^{(0)} &= \mathbf{y}^{n-1}, \\
\mathbf{y}^{(i)} &= \sum_{l=0}^{i-1} \Big[\alpha_{il}\, \mathbf{y}^{(l)} + \beta_{il}\, \Delta t\, \mathbf{L}_h(\mathbf{y}^{(l)}, t^{n-1} + \delta_l \Delta t) \Big], \; i = 1, \ldots, s, \\
\mathbf{y}^{n} &= \mathbf{y}^{(s)},
\end{aligned} \tag{43}$$

where s is the number of stages.

- The first order scheme is simply the forward Euler method.
- In the second order scheme (s=2) the coefficients are:

$$\begin{aligned}
&\alpha_{10} = \beta_{10} = 1, \;\; \alpha_{20} = \alpha_{21} = \beta_{21} = \frac{1}{2}, \;\; \beta_{20} = 0, \\
&\delta_0 = 0, \;\; \delta_1 = 1.
\end{aligned} \tag{44}$$

- The coefficients for the third order scheme (s=3) are as follows:

$$\begin{aligned}
&\alpha_{10} = \beta_{10} = 1, \;\; \alpha_{20} = \frac{3}{4}, \;\; \alpha_{21} = \beta_{21} = \frac{1}{4}, \;\; \beta_{20} = 0, \\
&\alpha_{30} = \frac{1}{3}, \;\; \beta_{30} = \alpha_{31} = \beta_{31} = 0, \;\; \alpha_{32} = \beta_{32} = \frac{2}{3}, \\
&\delta_0 = 0, \;\; \delta_1 = 1, \;\; \delta_2 = \frac{1}{2}.
\end{aligned} \tag{45}$$

The second component of the $RK\Lambda\Pi P$ methods is the local projection operator $\Lambda\Pi$ whose purpose is to control the magnitude of the higher-order degrees of freedom. Examples of $\Lambda\Pi$ operators for 1D are given in [11] and for standard 2D element shapes in [12]. With the local projection operator, the $RK\Lambda\Pi P$ method is defined as

$$\begin{aligned}
\mathbf{y}^{(0)} &= \mathbf{y}^{n-1}, \\
\mathbf{y}^{(i)} &= \Lambda\Pi \left(\sum_{l=0}^{i-1} \Big[\alpha_{il}\, \mathbf{y}^{(l)} + \beta_{il}\, \Delta t\, \mathbf{L}_h(\mathbf{y}^{(l)}, t^{n-1} + \delta_l \Delta t) \Big] \right), \; i = 1, \ldots, s, \\
\mathbf{y}^{n} &= \mathbf{y}^{(s)}.
\end{aligned} \tag{46}$$

For a suitable choice of the $\Lambda\Pi$ operator and for Δt satisfying the CFL condition, the scheme above can be shown to be TVD in the 1D [11] and 2D [12] cases. We note that in our implementation, we avoid the use of the local projection operator where possible, since our experience with various limiters shows that they generally interfere with the wave structure, thus we take $\Lambda\Pi = I$.

2.17 Numerical results

In this section we present results of some preliminary tests for the turbulence closure schemes implemented in UTBEST3D. For results of simulations on a range of analytical test cases and realistic flow scenarios that employ algebraic turbulence models, see [1, 15].

2.17.1 Channel flow

In the first test case we simulate steady homogenuous flow in a rectangular channel with a constant slope of 4×10^{-5}. A constant flow rate with average flow velocity of 1 m/s is imposed on the upstream (left) boundary. The channel dimensions (in meters) are: $L \times W \times D = 6,000 \times 2,000 \times 10$. The triangular surface grid consisting of 48 elements is shown in Figure 3. In the vertical the domain was subdivided in up to 40 equidistant layers.

The vertical profiles of x-velocity and kinetic turbulent energy at the midpoint of the ouflow boundary (x,y)=(6000, 0) for all of the one and two equation models we implemented are compared in Figure 4. Here we used 5 layers in the vertical direction and piecewise quadratic approximating spaces for all variables. These results were found to be in good agreement with results presented by Warner et al [25] for a similar test case. The velocity profiles in particular are quite similar for all models.

In Figure 5, we compare the vertical x-velocity and kinetic turbulent energy profiles for different approximation spaces and various vertical grid resolutions, for the Mellor-Yamada turbulence model. These include piecewise constant approximations with 10, 20 and 40 vertical layers; piecewise linears with 10 and 20 layers; and piecewise quadratics with 5 and 10 layers. Although the axial velocity profiles lie rather close together, the quadratic approximation appears to have a definite advantage when resolving the boundary layer at the sea bed, and gives reasonably converged results even with only 5 vertical layers. These results also confirm that piecewise constant approximations can be quite inaccurate, especially on coarser grids, which we have observed in previous studies. Piecewise linears or quadratics generally give much more accurate (and physically meaningful) answers. While quadratic solutions are more expensive to compute on a per-element basis, they may also allow for coarser grids, and thus end up being more efficient. Such efficiency studies will be the subject of future work.

2.17.2 Tidal flow in the Bight of Abaco

In the next test case, we simulate tide-driven flow in the Bight of Abaco, The Bahamas. The domain bathymetry and the finite element mesh consisting of 1696 elements are shown in Figure 6.

The following tidal forcing with time(t) in hours was imposed at the open sea boundary:

$$\begin{aligned} \hat{\xi}(t) &= 0.075\cos(\tfrac{t}{25.82} + 3.40) \\ &+ 0.095\cos(\tfrac{t}{23.94} + 3.60) \\ &+ 0.100\cos(\tfrac{t}{12.66} + 5.93) \\ &+ 0.395\cos(\tfrac{t}{12.42} + 0.00) \\ &+ 0.060\cos(\tfrac{t}{12.00} + 0.75) \quad (meters). \end{aligned} \tag{47}$$

The simulations were cold-started and the tidal forcing was imposed gradually over a period of two days. The Coriolis parameter was set to $3.19 \times 10^{-5} s^{-1}$.

In Figure 7, we compare the free surface elevation at time t=1,000,000 seconds computed using piecewise linear approximating spaces and up to 5 layers in the vertical direction. The actual number of vertical layers at any point in the grid varies due to changes in the bathymetry. The

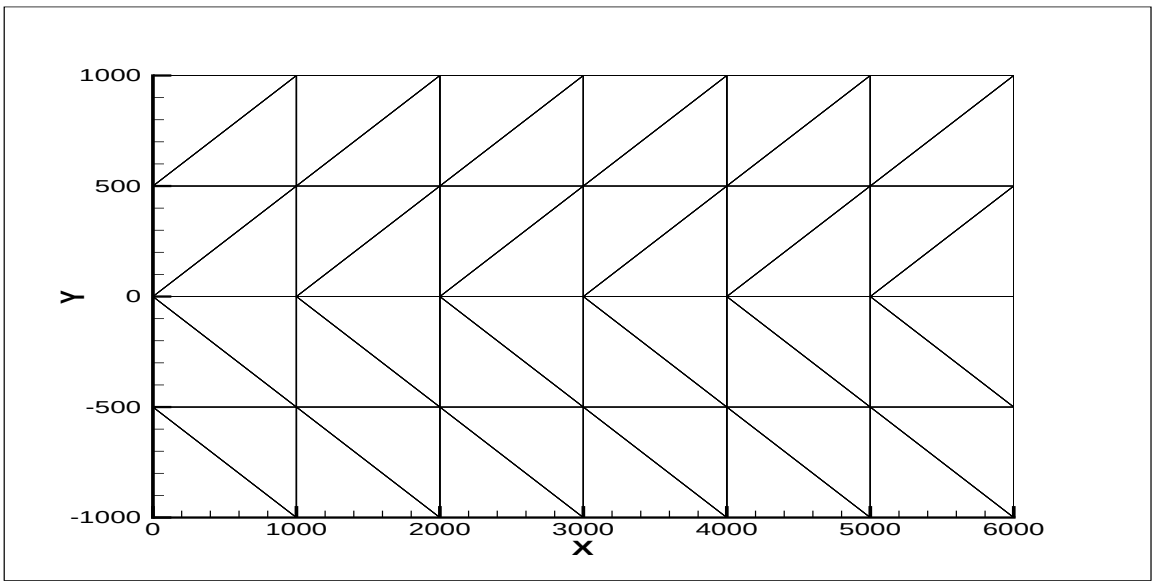
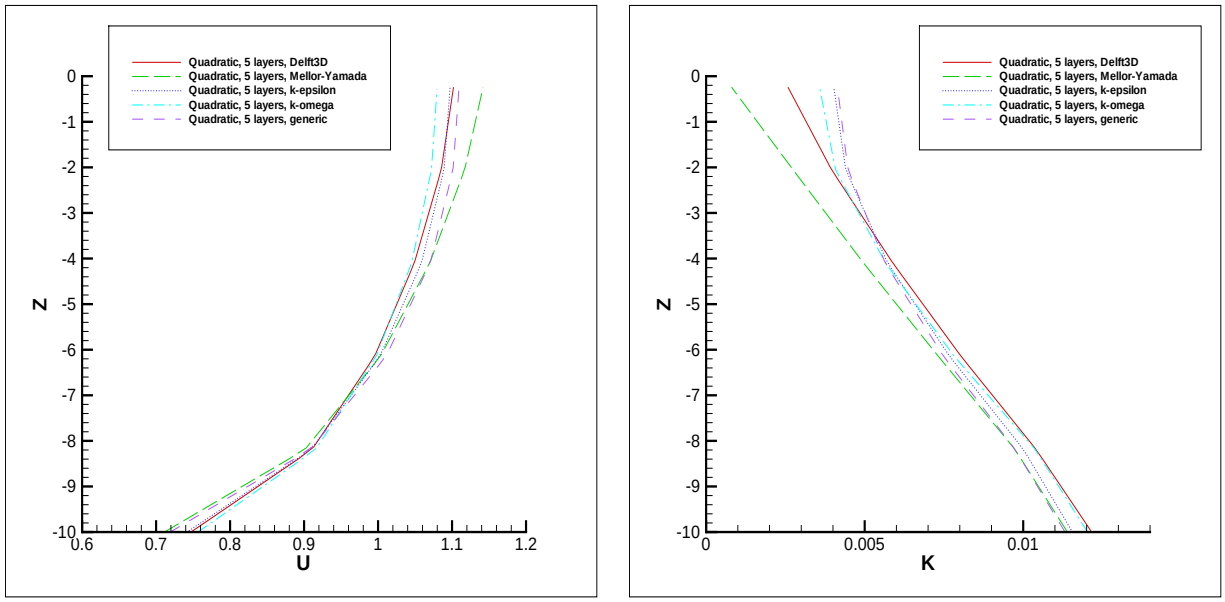


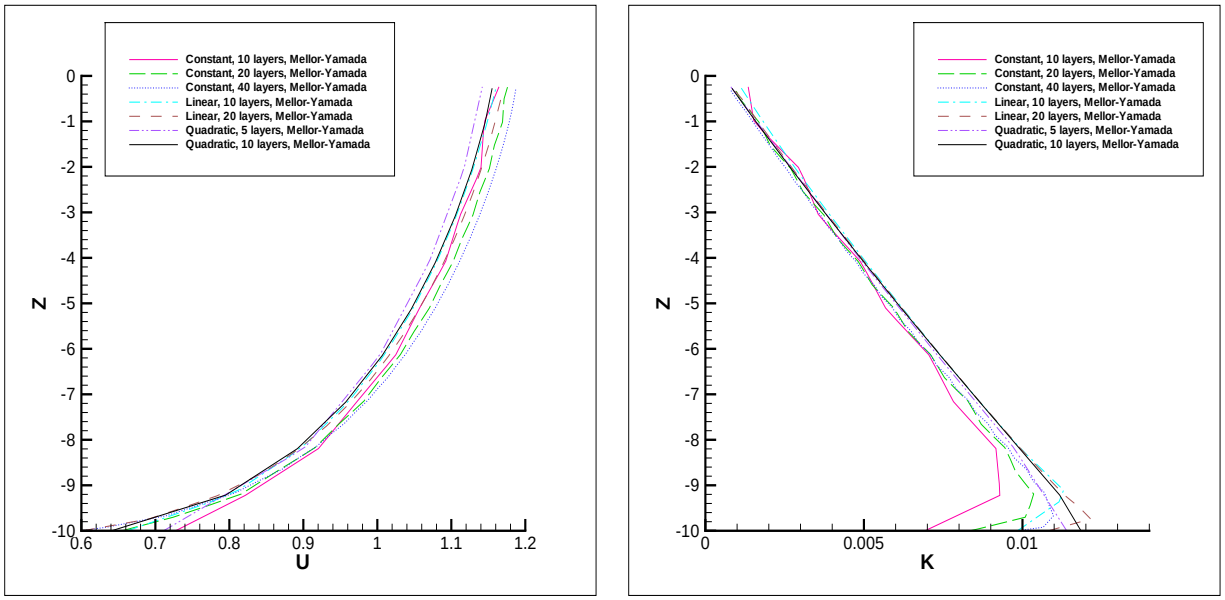
Figure 3: 2D mesh and the computational domain.



a) x-velocity b) kinetic turbulent energy

Figure 4: Comparison of order one (Delft3D), and four order two turbulence closure schemes.

time step for these runs was 5 seconds, except in the $k-\omega$ model, where it was 4 seconds. The results, including those obtained using the algebraic turbulence closure model, are closely matched (the differences are on the order of 1 cm or less). We did however observe numerical instabilities with the Mellor-Yamada model (results not presented) for this problem, perhaps due to the model's sensitivity to variations in the bathymetry. This will be the subject of future investigation.



a) x-velocity b) kinetic turbulent energy

Figure 5: Comparison of Mellor-Yamada scheme results for different approximation spaces and vertical grid resolutions.

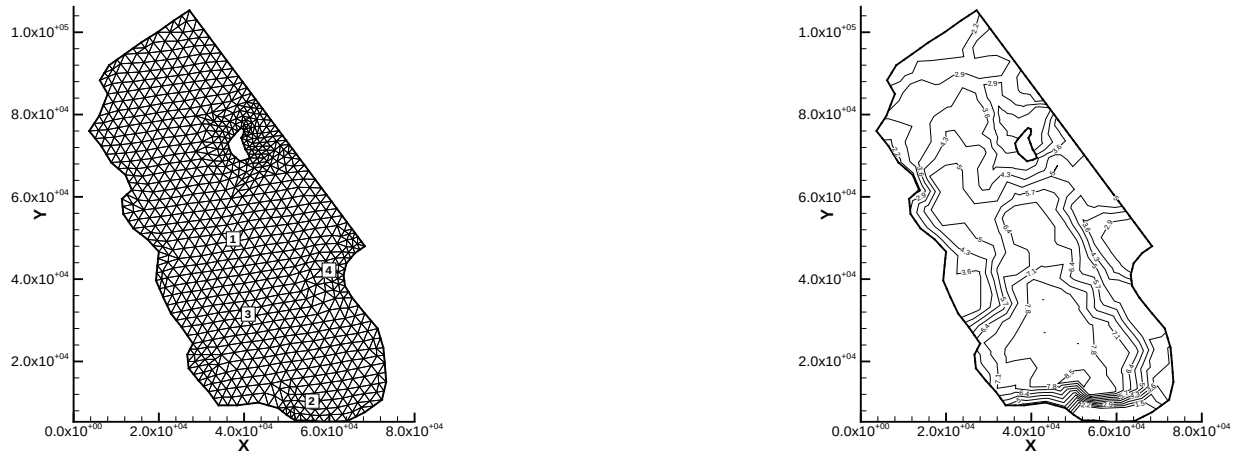


Figure 6: Finite element mesh and domain bathymetry for tidal flows around Bahamas simulation.

3 Conclusions and recommendations

Various algebraic, one and two equation turbulence models have been implemented in the UTBEST3D simulator, developed at the University of Texas at Austin. The model has gone through preliminary testing, with further testing to occur over the next few months.

Specific recommendations include testing this model under baroclinic conditions on problems of particular interest to the Texas Water Development Board. Inclusion of forcings due to wind, and implementation of a wetting and drying algorithm are the next steps in the development. The wetting and drying algorithm will mimic recent work on thin-layer models in a discontinuous

Galerkin-based code for the depth-averaged shallow water equations, developed by our group in collaboration with the group of J. Westerink [4].

Comparisons of UTBEST3D to other models, such as the ELCIRC or SELFE models, are warranted at this point, on problems of interest to TWDB.

4 Bibliography

References

[1] V. Aizinger, *A Discontinuous Galerkin Method for Two- and Three-Dimensional Shallow-Water Equations*, PhD Thesis, University of Texas at Austin, 2004.

[2] V. Aizinger, C. Dawson, *A discontinuous Galerkin method for two-dimensional flow and transport in shallow water*, Advances in Water Resources, 25, pp. 67-84, 2002.

[3] D.N. Arnold, F. Brezzi, B. Cockburn, L.D. Marini, *Unified Analysis of Discontinuous Galerkin Methods for Elliptic Problems*, SIAM J. Num. Anal., 39, pp. 1749-1779, 2002.

[4] S. Bunya, E.J. Kubatko, J.J. Westerink, C. Dawson and S. Yoshimura, *A mass conserving moving boundary method for discontinuous Galerkin solutions to the shallow water equations*, in preparation.

[5] J. C. Butcher, *The numerical analysis of ordinary differential equations: Runge-Kutta and general linear methods*, John Wiley and Sons, New York, 1987.

[6] Burchard, H., Bolding, K., *Comparative analysis of four second-moment turbulence closure models*, Ocean Model., 3, pp 33-50, 2001.

[7] S. Chippada, C. N. Dawson, M. Martinez, M. F. Wheeler, *A Godunov-type finite volume method for the system of shallow water equations*, Comput. Meth. Appl. Mech. Engrg., 151, pp. 105-129, 1998.

[8] B. Cockburn, G. Karniadakis and C.-W. Shu, *The development of discontinuous Galerkin methods*, in Discontinuous Galerkin Methods: Theory, Computation and Applications, B. Cockburn, G. Karniadakis and C.-W. Shu, editors, Lecture Notes in Computational Science and Engineering, volume 11, Part I: Overview, pp.3-50, Springer, 2000.

[9] B. Cockburn, C.-W. Shu, *The Runge-Kutta local projection P^1-discontinuous Galerkin method for scalar conservation laws*, RAIRO Modél. Math. Anal. Numér., 25, pp. 337-361, 1991.

[10] B. Cockburn, C.-W. Shu, *The local discontinuous Galerkin finite element method for convection-diffusion systems*, SIAM J. Numer. Anal., 35, pp. 2440-2463, 1998.

[11] B. Cockburn, C.-W. Shu, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for scalar conservation laws II: General framework*, Math. Comp., 52, pp. 411-435, 1989.

[12] B.Cockburn, S.Hou, C.-W. Shu, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: The multidimensional case*, Math. Comp., 54, pp. 545-581, 1990.

[13] B. Cockburn, C.-W. Shu, *The Runge-Kutta discontinuous Galerkin finite element method for conservation laws V: Multidimensional systems*, J. Comput. Phys., 141, pp. 199-224, 1998.

[14] A. M. Davies, *A three-dimensional model of the Northwest European continental shelf, with application to the M4 tide*, J. Phys. Oceanogr., 16(5), pp. 797-813, 1986.

[15] C.Dawson, V.Aizinger, *A Discontinuous Galerkin Method for Three-Dimensional Shallow Water Equations*, Journal of Scientific Computing, to appear.

[16] *Delft3D-FLOW User Manual*, WL—Delft Hydraulics, 2005.

[17] A. Harten, P. D. Lax, B. van Leer, *On upstream differencing and Godunov-type schemes for hyperbolic conservation laws*, SIAM Rev., 25, pp. 35-61, 1983.

[18] A. Harten, J. M. Hyman, *Self-Adjusting Grid for One-Dimensional Hyperbolic Conservation Laws*, J. Comp. Phys., 50, pp. 235-269, 1983.

[19] R.J. LeVeque, *Numerical Methods for Conservation Laws*, Birkhäuser, Basel, 1992.

[20] Mellor, G.L., Yamada, T., *Development of a turbulence closure model for geophysical fluid problems*, Rev. Geophys. Space Phys., 20, pp. 851-875, 1982.

[21] P.L. Roe, *Approximate Riemann solvers, parameter vectors, and difference schemes,* J. Comput. Phys., 43, pp. 357-372, 1981.

[22] Umlauf, L., Burchard, H., *A generic length-scale equation for geophysical turbulence models*, J. Marine Res., 61, pp 235-265, 2003.

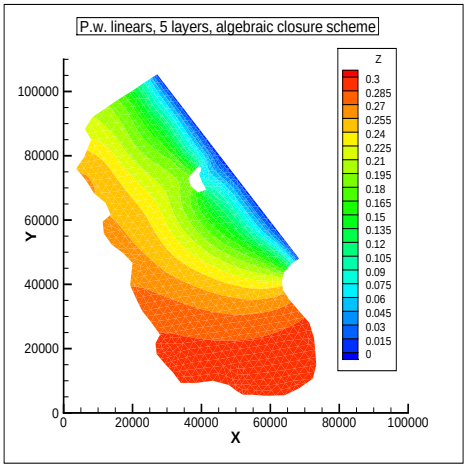
[23] A. Valiani, V. Caleffi, A. Zanni, *Finite Volume scheme for the 2D Shallow Water equations: Application to a flood event in the Toce river*, Universit degli Studi di Ferrara, preprint.

[24] C. B. Vreugdenhil, *Numerical Methods for Shallow-Water Flow*, Kluwer, 1994.

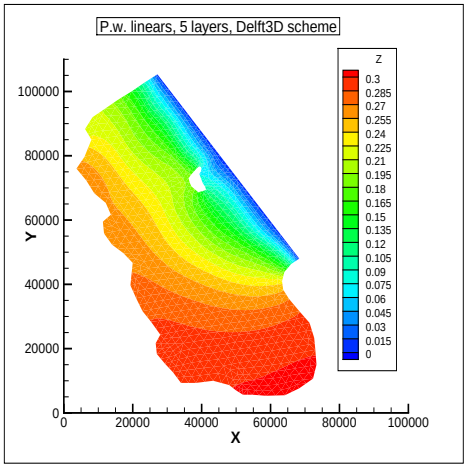
[25] Warner, J.C., Sherwood, C.R. Arango, H.G., Signell, R.P., *performance of four turbulenc closure models implemented using a generic length scale method*, Ocean Modeling, 8, pp. 81-113, 2005.

[26] T. Weiyan, *Shallow Water Hydrodynamics*, Elsevier Oceanography Series, 55, Elsevier, 1992.

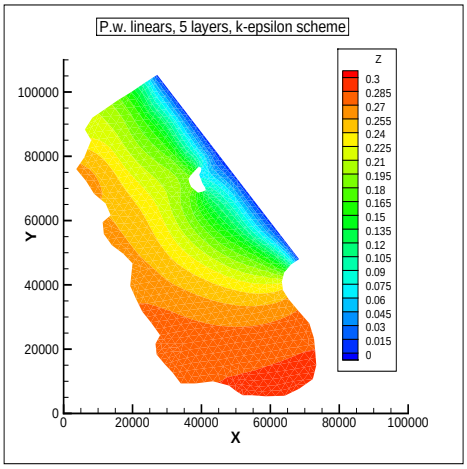
[27] Saffman, P.G., Wilcox, D.C., *Turbulence-model predictions for turbulent boundary layers*, AIAA J. 12(4), pp. 541-546, 1974.



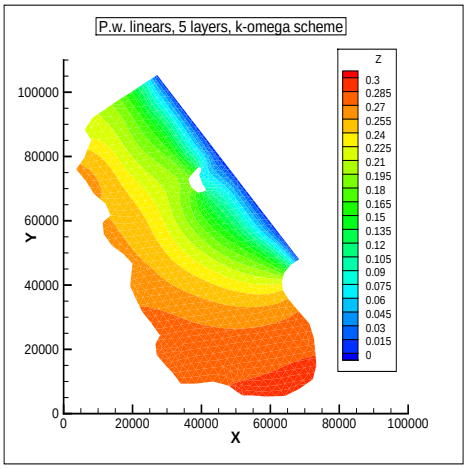
Algebraic (order 0) model



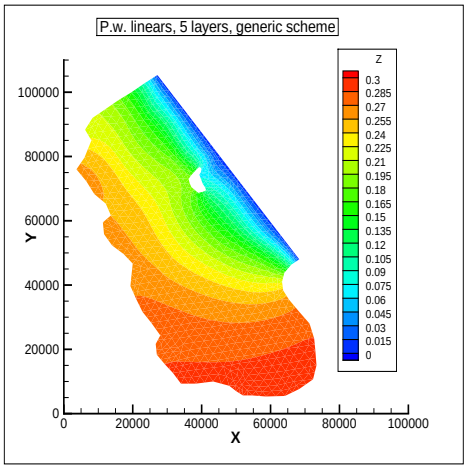
Delft3D (order 1) model



k-ϵ (order 2) model



k-ω (order 2) model



generic (order 2) model

Figure 7: Free surface elevation at time t=1,000,000 seconds for the Bight of Abaco. Lengths are in meters.