Texas Water Development Board
Project Number 0804830789

Final Report: UTBEST3D hydrodynamic model

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

Christopher Mirabito, Graduate Research Assistant

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 discontinuous Galerkin (DG) finite element method defined on unstructured prismatic meshes. The method is based on the use of discontinuous, piecewise polynomial approximating functions for each primary variable, defined over each element. The potential advantages of the DG method over more standard approaches 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 flexibility of the code allows for both lower order and/or higher order polynomials to be used to approximate the solutions, by simply setting a parameter in the input file. The method is also scalable on parallel machines.

During this project year, the first task was developing and testing a new parallel version of the code. This development will enhance our capabilities for future large-scale applications. The simulator was parallelized for small to large scale distributed memory platforms, using domain decomposition and the Message Passing Interface (MPI) library. A preprocessing code was developed which reads the global grid and input, and uses the software package (METIS) to divide the global domain into a specified number of subdomains, which are mapped to processors. Each processor then reads its specific data and computes the solution on its subdomain, passing information to neighboring subdomains as required. The parallel code was tested on the Ranger and Lonestar supercomputers at the Texas Advanced Computing Center (TACC), using data for the Corpus Christi Bay region supplied by TWDB. The tests show excellent scalability for the code up to 512 processors for low order approximations (piecewise constant for each solution component) and up to 1600 processors for higher order approximations (piecewise linear approximations). The results of these tests show that a one year, fully baroclinic simulation of Corpus Christi Bay may be completed in a matter of hours for low order approximations to a matter of a few days for higher order approximations. Future improvements to the efficiency of the code, and the time-stepping algorithm, could reduce these times further.

The second task was to collaborate with TWDB personnel in conducting verification and validation studies of the parallel code using data from Corpus Christi Bay. TWDB provided grid, boundary condition (elevation forcing and river inflows), wind and evaporation data files which were used in these studies. In order to perform these studies, additional features had to be added to the code to read and process the data files. Verification tests of standard tidal flows were performed using both low and higher order approximations, and for barotropic and baroclinic assumptions. These tests revealed that in the open ocean and away from freshwater inflow regions, the code produces similarly accurate results independent of the order of approximation. Near freshwater inflow regions, there were some differences between low and higher order approximations, and between baroclinic and barotropic solutions, which require further investigation. Additional numerical simulations were performed to test the new features added to the code, including 1) reading elevation forcing data (rather than computing it through standard tidal formulas), 2) reading specified evaporation rates, and 3) reading specified wind velocities. Extensive validation studies (comparison to other model output and/or comparison to field data) have not been performed yet; however, we are at the point where these studies may be performed in the near future.

2 Model Description

2.1 Introduction

In this report, we discuss the parallelization and recent applications of the UTBEST3D (University of Texas Bay and Estuary 3D) simulator, which has been developed at UT Austin by the investigators. The UTBEST3D model development was motivated by the fact that, despite many recent advances in the development of 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 higher-order accurate under highly varying flow regimes. Algorithms such as the discontinuous Galerkin method are of great interest within the ocean and coastal modeling communities; at a recent unstructured grid ocean modeling workshop, held in Halifax, Nova Scotia, many of the talks focused on applications of the DG method to three-dimensional ocean models. 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 [5] for a historical overview of DG methods.

In [1, 4], 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 [6], which employs a mixed formulation. Application of the methodology to three-dimensional shallow water models was described in [8, 2] and in a series of TWDB annual reports dating back to 2005. 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.

The code was originally developed as a serial code. During this project, the code was parallelized for distributed memory clusters. The parallelization of the code is based on domain decomposition, whereby the global three-dimensional domain is split into subdomains. Each processor then computes the solution on its assigned subdomain, and shares information with neighboring subdomains using MPI (message passing interface). Because of the local nature of the DG scheme, and the use of explicit time-stepping, the code scales well in parallel.

The parallel code has been tested for parallel efficiency and accuracy, and applied to the simulation of tide-driven flows in Corpus Christi Bay. Data for the bay was supplied by TWDB. Additional features have been incorporated into the code, including time-varying open sea boundary data, wind data, and evaporation data.

The rest of this report is organized as follows. In the next section, we briefly review the modeling assumptions used in the code, describe the flow equations, boundary conditions, and turbulence closure models. We then describe the parallelization of the code, including a description of the domain decomposition algorithm, and present some parallel scaling studies for baroclinic flows in Corpus Christi Bay. Then we discuss applications of the parallel code to barotropic and baroclinic tidal flows in the bay, and to flows with open ocean elevation, wind and evaporation specified through input data files.

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 the 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.

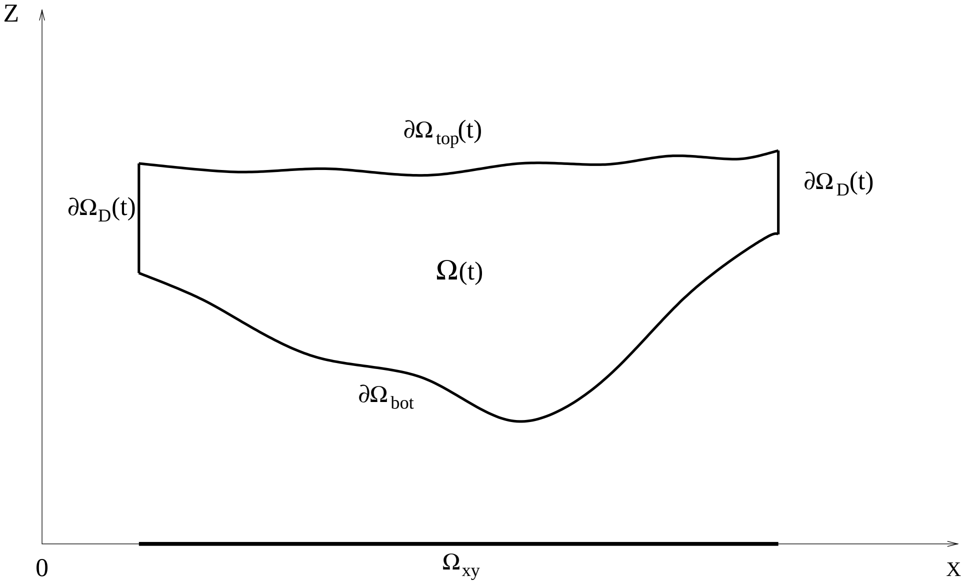
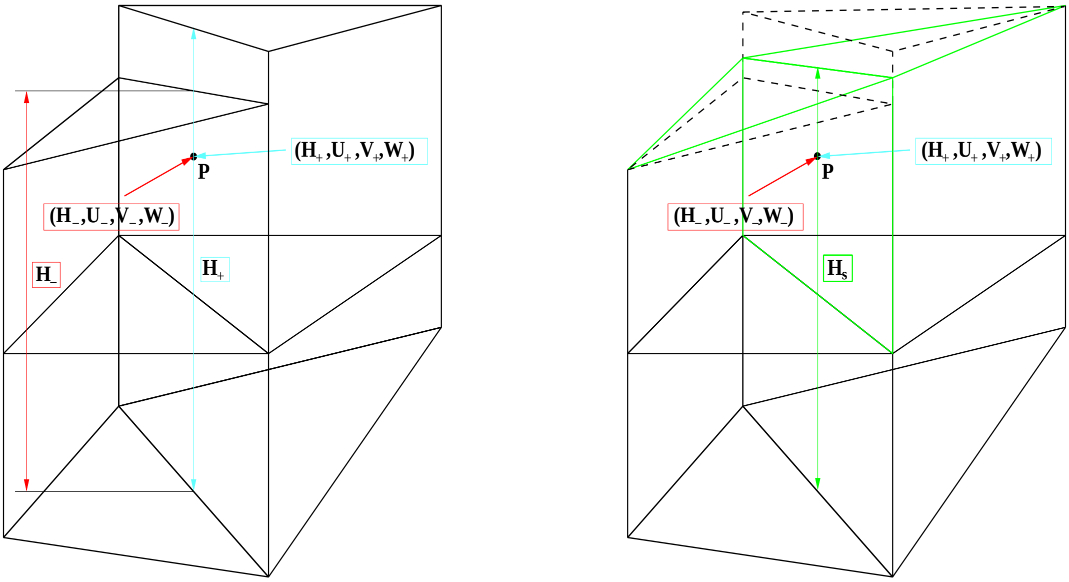


Figure 1: Vertical cross-section of the computational domain $\Omega(t)$.

Keeping in line with the specific anisotropy of $\Omega(t)$ we construct a 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.

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 the 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 the 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 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.



a) Before surface mesh smoothing

b) After surface mesh smoothing

Figure 2: Illustration of mesh smoothing.

2.3 3D baroclinic shallow water equations

The momentum equations in conservative form are given by [14]

$$\partial_t \mathbf{u}_{xy} + \nabla \cdot (\mathbf{u}_{xy}\mathbf{u} - \mathcal{D}\nabla \mathbf{u}_{xy}) + g\nabla_{xy}\xi + \frac{g}{\rho_0}\nabla_{xy}\int_z^{\xi}(\rho(T,S,\xi-\tilde{z})-\rho_0)d\tilde{z} - f_c\mathbf{k}\times\mathbf{u}_{xy} = \mathbf{F}, \quad (1)$$

where ρ_0 is the reference density and $\rho(T,S,p)$ is the density computed from the equation of state. The wind stress, atmospheric pressure gradient, and 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:

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

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 equation of state used in UTBEST3D is due to Klinger [11] and is given by

$$\rho(T,S,p) = C(p)\ \beta(p)S - \alpha(T,p)T - \gamma(T,p)(35-S)T, \quad (3)$$

where

$$
\begin{aligned}
C &= 999.83 \;+\; 5.053p \;-\; 0.048p^2, &(4)\\
\beta &= 0.808 \;-\; 0.0085p, &(5)\\
\alpha &= 0.0708(1 \;+\; 0.351p \;+\; 0.068(1-0.0683p)T), &(6)\\
\gamma &= 0.003(1 \;-\; 0.059p \;-\; 0.012(1-0.064p)T). &(7)
\end{aligned}
$$

p is the height of the water column above the point expressed in kilometers, T is the temperature in degrees Celsius, and S is the salinity in psu.

The continuity equation is

$$\nabla \cdot \mathbf{u} \;=\; 0. \tag{8}$$

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

  $$\mathbf{u}(z_b) \cdot \mathbf{n} = 0 \tag{9}$$

  and the quadratic slip condition for the horizontal velocity components

  $$
  \begin{aligned}
  \nu_t \frac{\partial u}{\partial z}(z_b) &= C_f \sqrt{u^2(z_b) + v^2(z_b)}\,u(z_b), &(10)\\
  \nu_t \frac{\partial v}{\partial z}(z_b) &= C_f \sqrt{u^2(z_b) + v^2(z_b)}\,v(z_b), &(11)
  \end{aligned}
  $$

  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

  $$\partial_t \xi \;+\; u(\xi)\,\partial_x \xi \;+\; v(\xi)\,\partial_y \xi \;-\; w(\xi) \;=\; 0, \tag{12}$$

  and

  $$\nabla u(\xi) \cdot \mathbf{n} \;=\; \nabla v(\xi) \cdot \mathbf{n} \;=\; 0 \tag{13}$$

  in the case of no wind. In the presence of wind forcing however, the last equation is replaced by

  $$\nu_t \frac{\partial \mathbf{u}}{\partial z}(\xi) = \tau_{\mathbf{s}}, \tag{14}$$

  where $\tau_{\mathbf{s}}$ is the surface stress which can be specified directly or computed from the wind velocity at 10m above the water surface, $\mathbf{U}_{10}$, by

  $$\tau_{\mathbf{s}} \;=\; \frac{\rho_a}{\rho_0} C_s |\mathbf{U}_{10}| \mathbf{U}_{10} \tag{15}$$

  with $C_s = 10^{-3}(A_{W1} + A_{W2}|\mathbf{U}_{10}|)$ for $U_{low} \leq |\mathbf{U}_{10}| \leq U_{high}$ and C_s held constant at the extremal values outside of this interval. Similarly to [17] we set $A_{W1} = 0.1$, $A_{W2} = 0.063$, $U_{low} = 6\ m/s$, $U_{high} = 50\ m/s$.

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

- Land boundary: No normal flow

  $$u_{\mathbf{n}} = \mathbf{u} \cdot \mathbf{n} = 0, \tag{16}$$

  and zero shear stress

  $$\nabla u_{\boldsymbol{\tau}} \cdot \mathbf{n} \;=\; 0, \tag{17}$$

  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

  $$\nabla u \cdot \mathbf{n} \;=\; \nabla v \cdot \mathbf{n} \;=\; 0, \tag{18}$$

  and prescribed surface elevation $\xi_{os}(x, y, t)$

  $$\xi = \xi_{os}(x, y, t). \tag{19}$$

- River boundary: Prescribed velocities

  $$\mathbf{u} = \mathbf{u}_r, \tag{20}$$

  and prescribed surface elevation

  $$\xi = \xi_r. \tag{21}$$

- Radiation boundary: Zero normal derivative of the horizontal velocity components

  $$\nabla u \cdot \mathbf{n} \;=\; \nabla v \cdot \mathbf{n} \;=\; 0. \tag{22}$$

Analytically, the free surface elevation can be computed from (12). However, a computationally more robust method [14] is obtained by integrating continuity equation (8) over the total height of the water column. Taking into account boundary conditions (9) – (12) at the bottom and top boundaries we arrive at a 2D equation for the free surface elevation commonly called the primitive continuity equation (PCE),

$$\partial_t \xi \;+\; \partial_x \int_{z_b}^{\xi} u dz \;+\; \partial_y \int_{z_b}^{\xi} v dz \;=\; 0. \tag{23}$$

2.5 Species transport

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

$$r_t + \nabla \cdot (\mathbf{u}r) - \nabla \cdot (K_r \nabla r) = f, \quad \Omega(t) \times (0, T), \tag{24}$$

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 DG method is also applied to the solution of these equations.

2.6 Turbulence

UTBEST3D provides vertical eddy viscosity models of various levels of computational and conceptual complexity. In order of increasing complexity those 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 specifying diagonal entries to the tensors of eddy viscosity/diffusivity coefficients for all variables in (1) and (24).
- Two algebraic models implemented in UTBEST3D are due to Davies [7] and give good results at a reasonable computational cost in cases where accurate vertical resolution of flow is not important.

  In the first algebraic 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 equation for the turbulent kinetic energy in addition to the mass, momentum, and species transport equations.

  $$k_t + \nabla \cdot (\mathbf{u}k) - \frac{\partial}{\partial z}(\nu_k \frac{\partial}{\partial z}k) = \nu_t \left(\left(\frac{\partial u}{\partial z}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 \right) + \nu_r \frac{g}{\rho_0}\frac{\partial \rho}{\partial z} - \epsilon, \qquad (25)$$

  where ν_k is the vertical diffusivity 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 is computed algebraically in this model and is set equal to $l(z) \ = \ \kappa\ (z - z_b)\ \sqrt{\xi - z}\ F_l(Ri)$ (see Delft3D-Flow manual [9]). $C_\mu^0 = \sqrt{0.3}$ is a calibration constant, $\kappa = 0.4$ is the von Karman constant, and $F_l(Ri)$ is the damping function accounting for stratification effects. F_l depends on the gradient Richardson number

  $$Ri \ = \ \frac{-\frac{g}{\rho_0}\frac{\partial \rho}{\partial z}}{\left(\frac{\partial u}{\partial z}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2} \qquad (26)$$

  and is of the form:

  $$F_l(Ri) \ = \ \begin{cases} e^{-2.3Ri}, & Ri \geq 0, \\ (1 - 14Ri)^{0.25}, & Ri < 0. \end{cases} \qquad (27)$$

  Once k is computed one can obtain the vertical eddy viscosity and diffusivity coefficients by taking $\nu_t = C_\mu^0 k^{\frac{1}{2}} l$ and $\nu_r = \nu_k = \frac{\nu_t}{0.7}$ correspondingly. Neumann type boundary conditions for k are used at the free surface and the sea bed $\nu_k \frac{\partial k}{\partial n} = 0$.
- The second order closure model implemented in UTBEST3D is based on the generic turbulence length scale model proposed by Warner et al [15]. The main 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 derived quantity ψ

  $$\psi_t + \nabla \cdot (\mathbf{u}\psi) - \frac{\partial}{\partial z}(\nu_\psi \frac{\partial}{\partial z}\psi) = \frac{\psi}{k}\left(C_1 \nu_t \left(\left(\frac{\partial u}{\partial z}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 \right) + C_3 \nu_r \frac{g}{\rho_0}\frac{\partial \rho}{\partial z} - C_2 \epsilon F_{wall} \right), \qquad (28)$$

where $\psi = (C_\mu^0)^p k^m l^n$ and C_3 is equal to C_3^- for stably stratified flow and C_3^+ otherwise. Depending on the choice of p, m, and n we obtain different closure schemes. The turbulent mixing length is computed using k and ψ. The eddy viscosity and diffusivity coefficients are obtained from $\nu_t = \sqrt{2} S_m k^{\frac{1}{2}} l$ and $\nu_r = \sqrt{2} S_h k^{\frac{1}{2}} l$, where S_m and S_h are stability functions given by:

$$S_h = \frac{0.4939}{1 - 30.19 G_h}, \qquad S_m = \frac{0.392 + 17.07 S_h G_h}{1 - 6.127 G_h}, \tag{29}$$

where $G_h = G_{h_u} - \frac{(G_{h_u} - G_{h_c})^2}{G_{h_u} + G_{h0} - 2G_{h_c}}$ and $G_{h_u} = \min(G_{h0}, \max(-0.28, \frac{g}{\rho_0} \frac{\partial \rho}{\partial z} \frac{l^2}{2k}))$ with $G_{h0} = 0.0233, G_{h_c} = 0.02$.

To improve stability properties of the two-equation model we employ Neumann boundary conditions for ψ at the free surface $\nu_\psi \frac{\partial \psi}{\partial z} = -n \nu_\psi (C_\mu^0)^p k^m \kappa l_s^{n-1}$ and at the sea bed $\nu_\psi \frac{\partial \psi}{\partial z} = n \nu_\psi (C_\mu^0)^p k^m \kappa l_b^{n-1}$, where the turbulent mixing length is derived from the law of the wall: $l_s = l_{|\xi_s} = z_1 e^{\frac{\kappa |u|}{\tau_s}}$ and $l_b = l_{|z_b} = z_0 \frac{\kappa}{\sqrt{Cf}}$ with C_f being as in (11) and z_1 the surface roughness coefficient.

Values of the parameters for four popular two equation models are shown in Table 1. Discretization of (28) is also done similarly to (24).

	Mellor-Yamada [12]	$k - \epsilon$ [3]	$k - \omega$ [16]	generic [13]
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}	7.6e-6	7.6e-6	7.6e-6	7.6e-6
ψ_{min}	1e-8	1e-8	1e-8	1e-8
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.

3 Description of the serial code

The serial version of UTBEST3D is based on a discontinuous Galerkin discretizaton; that is, each unknown variable is approximated as a discontinuous, piecewise polynomial, the degree of which is chosen by the user. At each time step, the solution is advanced in time using explicit Runge-Kutta methods. For piecewise constant spatial approximations, a forward Euler method is used for temporal integration; for piecewise linear approximations, a second order Runge-Kutta method is used, etc. The algorithm is described in the user's manual and has been described in detail in previous TWDB reports.

The input to the code consists of four input files, which are named fort.14, fort.15, fort.17 and utbest_config.inp. The fort.14 and fort.15 files are modeled after input files used for ADCIRC, ELCIRC and SELFE. The full description of these files is given in the user's manual, but briefly:

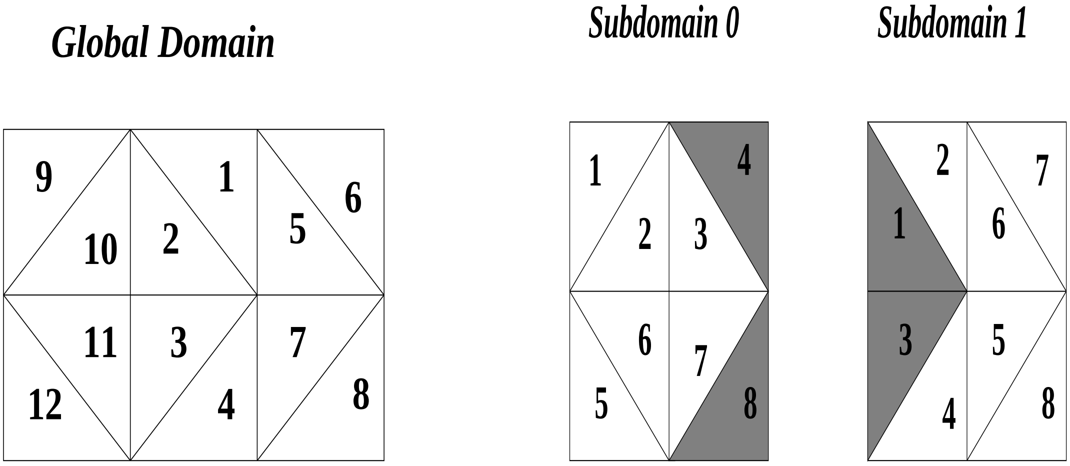
- fort.14–contains finite element mesh (nodes and elements) information for the 2D grid, node numbers and coordinates, element-to-node connectivity
- fort.15–contains 2D run parameters, time step information, output specifications
- fort.17–contains finite element edge information needed for the DG method; edge numbering, nodes connected to an edge, elements on each side of the edge, types of boundary conditions specified on the boundary edges
- utbest_config.inp–contains additional run parameters needed for the 3D code, locations of input/output files, turbulence options, number of vertical layers, and order of approximation used in the DG method.

4 Parallelization of the model

The model has been parallelized using domain decomposition and MPI. The 2D projection Ω_{xy} of the 3D domain Ω is partitioned into overlapping subdomains using the METIS library [10]. Each subdomain has a one-to-one correspondence with a compute processor. A partitioning code was written which reads in the two-dimensional mesh information contained in fort.14 and fort.17 and passes this information to METIS, which uses an algorithm to divide the global grid into local grids for each subdomain. The partitioning code creates new input files fort_np*xxx*.14 and fort_np*xxx*.17, where *xxx* is the total number of subdomains. The local grid information for each processor is contained in these files. On each subdomain, the elements/nodes are renumbered from one to the number of elements/nodes contained in the subdomain. The preprocessor creates additional files fort_np*xxx*.18 and fort_np*xxx*.offset. The first file contains information about the local-to-global element numbering and the message passing information from one subdomain to its neighbors, while the second file is a binary file used in MPI–I/O routines in UTBEST3D.

The basic ideas are captured in Figure 3. Here we see a global grid consisting of 12 elements which has been divided into two overlapping subdomains with 8 elements each. On each subdomain, the elements have been relabeled 1-8. Local element 1 on subdomain 0, for example, is global element 9. The global elements 1-4 are in the overlap region between the two subdomains, and thus are shared by both subdomains. Each subdomain has “resident” elements and “ghost” elements. The ghost elements in each subdomain are shaded. Elements 4 and 8 on subdomain 0 are ghost elements, labeled 2 and 4 on subdomain 1, while elements 1-3 and 5-7 are resident elements. On subdomain 1, elements 1 and 3 are ghost elements, labeled elements 3 and 7 on subdomain 0, while resident elements are 2 and 4-8. The resident elements are those elements where the DG solution is computed correctly. To do so however, the DG method requires information from the nearest neighboring elements, which may include ghost elements. The information for the ghost elements must be obtained from the domain where this element resides. Thus, for example, the solution in element 2 on subdomain 1 must be passed to subdomain 0, while the solution in element 3 on subdomain 0 must be passed to subdomain 1. All of the message passing information computed in the mesh partitioning code and captured in fort_npxxx.18.

When running UTBEST3D in parallel, each processor reads its grid information from fort_npxxx.14 and fort_npxxx.17 and its message passing information from fort_npxxx.18. The information contained in fort.15 is global information which is common to each processor. At each stage in the Runge-Kutta scheme, MPI is used to pass element solution information among the subdomains. Since the code is fully explicit, no global systems of equations must be solved and no global information must be broadcast to all of the processors; furthermore, the DG scheme is very local and only requires nearest neighbor communications. Therefore, the parallel speed-up of the code is quite good, as we show in the next section.



a) Global mesh consisting of 12 elements

b) Paritition of the mesh into 2 subdomains

Figure 3: Illustration of domain decomposition

5 Parallel performance

The code has been implemented and tested on both the Lonestar and Ranger supercomputers at the Texas Advanced Computing Center (TACC). Here we focus on parallel testing which occurred on Ranger. Ranger is a Sun Constellation Linux Cluster with 3936 nodes. Each node contains four AMD Opteron Quad-Core 64-bit processors, with 32 GB of memory. Therefore, each compute node contains 16 cores or CPUs, for a total of 62976 CPUs.

The primary applicaton focus of this project was a study of the Texas coast centered around Corpus Christi Bay, south to Baffin Bay and north to Aransas and Copano Bays. A two-dimensional triangular surface grid for this region was provided by TWDB, consisting of 18862 elements and 10877 nodes. An aerial view of the domain and the top layer of the finite element mesh is shown in Figure 4. The decomposition of the domain into 128 subdomains, using the preprocessor described in the previous section, is shown in Figure 5.

The code has been tested for parallel performance under a variety of scenarios, here we focus on one scenario. We consider baroclinic flow with a $k - \epsilon$ turbulence closure model. Thus, we are solving for variables u, v, w, ξ, S, T, k and ϵ. A standard tidal forcing is imposed on the open ocean boundary. We consider piecewise constant approximations for all variables, and piecewise linear approximations for all variables except for salinity, which we approximate as piecewise constant. A stable time step is chosen for each case, which for piecewise constants was 1 second, and for piecewise linears was .5 seconds, and we ran a one day simulation. We assume no wind and no evaporation. The domain is discretized vertically into a maximum of five layers; however, due to the shallow nature of the bay, the majority of the domain has one or two vertical layers. The 3D mesh contains a total of 24379 prismatic elements.

In Table 2, we show the wall clock time vs. number of CPUs for the piecewise constant case.

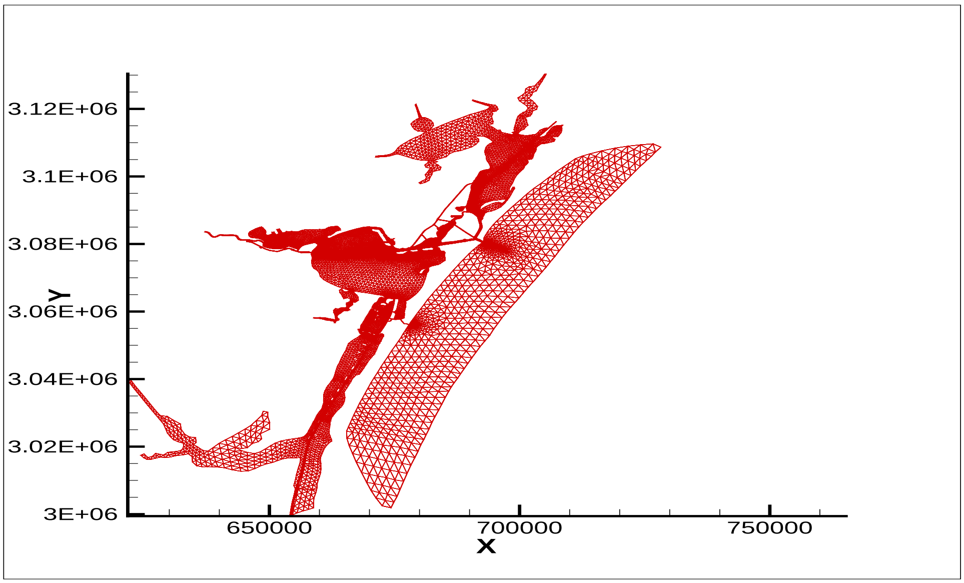


Figure 4: Aerial view of Baffin/Corpus Christi/Aransas/Copano Bays and top layer of the finite element mesh

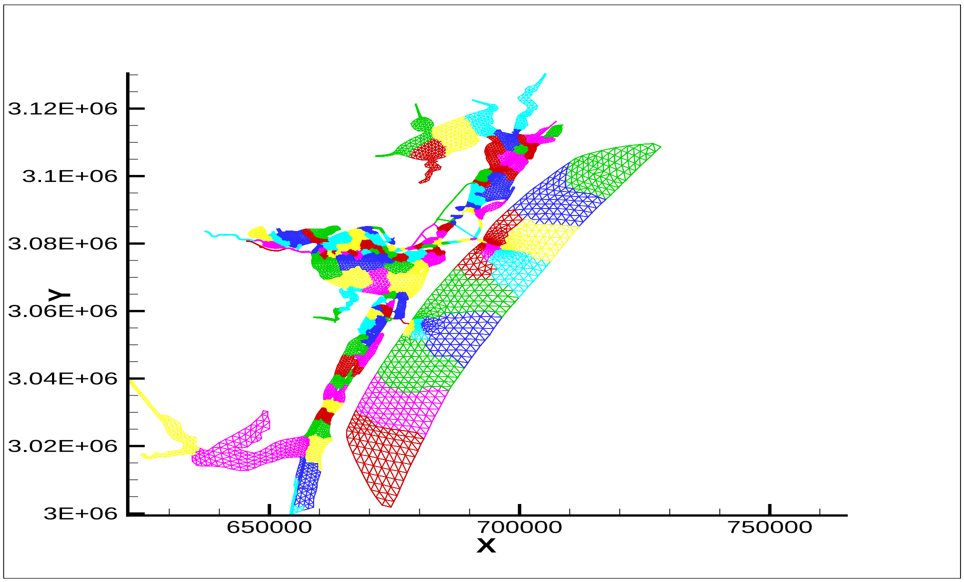


Figure 5: Aerial view of domain decomposition of Baffin/Corpus Christi/Aransas/Copano Bays into 128 subdomains

The third column, labeled efficiency measure, is defined on each row of the table by

$$E = \frac{\mathrm{WCT}(2 * x)}{2 * \mathrm{WCT}(x)}$$

where $\mathrm{WCT}(x)$ is the wall clock time for x processors. Given that the mesh is fixed and we are doubling the number of processors from one run to the next, the theoretically optimal value for E is 1.0. Note that, in going from 16 to 32 processors on Ranger, we are going from intra-node communications to inter-node communications, since Ranger has 16 cores per node. This might explain the slighty discrepancy in efficiency from 16 to 32 CPUs. Overall, the efficiency is excellent up to 512 CPUs, where it begins to tail off. Above 512 CPUs the efficiency drops off drastically because message passing begins to dominate the computational work assigned to each CPU. We also note the efficiency is actually better than 1.0 ("super-efficiency") when going from 64 to 128 processors. This is probably due to better utilization of cache on each node as the memory per node is decreased. We have seen this behavior with other parallel codes on Ranger. The outcome of this test is that we can perform one day of simulation of baroclinic flows in the Corpus Christi Bay region on Ranger in approximately 2 minutes. Extrapolating we should be able to perform a 1 year baroclinic simulation in 12-13 hours with piecewise constant approximations.

CPUs	Wall clock time in seconds	Efficiency measure E
16	3354	–
32	1997	.84
64	1019	1.02
128	399	1.27
256	200	1.00
512	131	.76

Table 2: Parallel scaling for piecewise constant approximations

In order to increase the accuracy of the DG method, we can raise the order of the approximating space by using piecewise linear approximations on each element. This introduces three additional parameters (x, y and z slopes) for each variable on each element, thus the number of degrees of freedom is significantly increased over piecewise constants, by roughly a factor of $3 * NC * NE$ where NC is the number of solution components (not counting elevation) and NE is the number of elements in the three-dimensional mesh (the cost of computing the water elevation is also increased, but this variable is a function only of x and y). Furthermore, the numerical integrations over each element must be more accurate and hence more expensive, and we also use a 2-stage Runge-Kutta method rather than an 1-stage Euler method. On a serial machine, the additional cost of computing a piecewise linear solution over a piecewise constant solution is directly proportional to the increase in the number of degrees of freedom. This is not necessarily the case on a parallel machine. Since there is now more work per subdomain, the parallel performance may actually improve and offset some of the expense. In Table 3, we examine the wall clock time vs. number of CPUs for the piecewise linear case. Because of the increased amount of computational effort, we chose for our base run 256 CPUs. Increasing the order of approximation also requires a reduced time step for stability; therefore we perform a 1 day simulation with a .5 second time step. As seen in Table 3, we see excellent parallel scaling for piecewise linear approximations for 256, 512 and 1024 processors. At 2048 processors the performance starts to degrade; this is not surprising, since at this point for the given mesh there are fewer than 10 elements per processor. Further testing revealed that the "optimal" number of processors for this case is 1600, which gives almost the exact same CPU time as 2048 processors. In summary, for piecewise linear approximations, we are able to do 1 day of simulation in about 16 minutes on Ranger, or about 8 times longer than with piecewise constants. Thus a one year simulation will take about 4 days of compute time on Ranger.

CPUs	Wall clock time in seconds	Efficiency measure E
256	6158	–
512	2923	1.05
1024	1549	.94
2048	985	.79

Table 3: Parallel scaling for piecewise linear approximations

6 Verification Studies

We performed a number of simulations on the region shown in Figure 4, to test the accuracy and stability of the code and to test new features to the code. As remarked above, a finite element mesh for the region was obtained from TWDB. Additional input files and run parameters were given to us by TWDB. The additional data included open ocean boundary elevations, wind data and evaporation rates over a 12 month period. We added features to UTBEST3D to read and process these data files for use in the code. In order to thoroughly debug and test the model for this domain, we took as a base case a tidal flow scenario with elevation specified at the open ocean boundary using standard formulas for tidal forcings, and ignoring wind, evaporation. Below we describe the results of these studies.

6.1 Base case: function specified tidal forcing

The first test case was a 1 month simulation of tidal flow, assuming no wind and no evaporation. The following tidal forcing with time (t) in hours was imposed at the open sea boundary (shown in Figure 6),

$$\begin{aligned} \hat{\xi}(t) &= 0.075\cos(\tfrac{t}{25.82} + 3.40) \\ &+ 0.095\cos(\tfrac{t}{23.94} + 3.60) \\ &+ 0.100\cos(\tfrac{t}{12.66} + 5.93) \\ &+ 0.395\cos(\tfrac{t}{12.42} + 0.00) \\ &+ 0.060\cos(\tfrac{t}{12.00} + 0.75) \quad (meters). \end{aligned} \tag{30}$$

The tide was ramped up over a 2 day period.

In addition to the tidal forcing, a number of river and power plant inflows/outflows were included in the model. For the purposes of these tests, the flow rates were assumed constant over the 1 month simulation period. The rivers, flow rates, temperature and salinity boundary conditions are given in Table 4. The approximate locations of the rivers and power plants are labeled in Figure 6.

We simulated both barotropic and baroclinic cases. For this test problem, temperature is initially constant at 20 C and because the temperature is also 20 C all of the inflow locations, the temperature solution should stay constant for all time. Salinity is initially 35 ppt over the whole domain; however, salinity is assumed to be 0 ppt at the inflow points. Therefore, there may be some baroclinic effects near these locations. Away from these points, salinity remains essentially constant, and baroclinic effects should be negligible. For the baroclinic case, the $k - \epsilon$ turbulence closure model was used to compute the vertically eddy viscosity. For the barotropic case, the first algebraic model described in Section 2.6 was used for vertical eddy viscosity.

We also compared piecewise constant ($P0$) and piecewise linear ($P1$) approximations of the primary variables. In the latter case, we kept the salinity approximation as a piecewise constant,

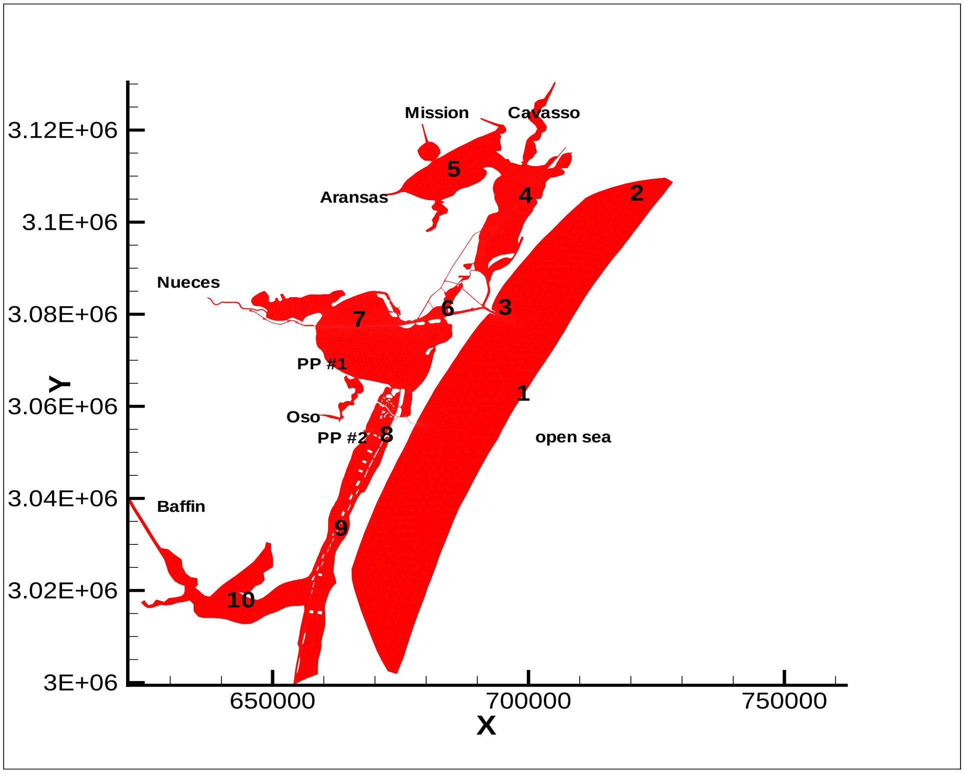


Figure 6: Aerial view of the domain showing the open sea boundary, the various rivers and power plants, and the locations 1-10 where solution quantities were measured

River/power plant	Flow rate m/s	Temperature (c)	Salinity (ppt)
Cavasso	$-7.8 * 10^{-5}$	20	0
Mission	$-3.37 * 10^{-4}$	20	0
Aransas	$-1.82 * 10^{-4}$	20	0
Nueces	$-4.85 * 10^{-3}$	20	0
Power plant 1 outflow	$-4.43 * 10^{-2}$	20	35
Power plant 1 inflow	$-4.63 * 10^{-2}$	20	0
Oso	$-5.68 * 10^{-4}$	20	0
Power plant 2 outflow	$-1.12 * 10^{-1}$	20	35
Power plant 2 inflow	$1.05 * 10^{-1}$	20	0
Baffin	$-1.86 * 10^{-4}$	20	0

Table 4: Inflow/outflow points, flow rates, temperature and salinity for Corpus Christi Bay

to avoid spurious overshoot/undershoot caused by the steep salinity gradients. For the P^0 case, a time step of $\Delta t = 1$ second was used, while for the P^1 case, $\Delta t = .5$ seconds. The model and run parameters for the four test cases considered are summarized in Table 5.

During the course of each simulation, we measured elevation and depth-averaged horizontal velocities at a number of locations throughout the domain. We present solutions at 10 of those locations, labeled 1-10 in Figure 6 and described in more detail in Table 6. In generating these

Run	Model	Approximation	Time step	Turbulence model
1	barotropic	$P0$	1 sec	Algebraic 1
2	barotropic	$P1$	.5 sec	Algebraic 1
3	baroclinic	$P0$	1 sec	$k--\epsilon$
4	baroclinic	$P1$	.5 sec	$k--\epsilon$

Table 5: Basic description of runs 1-4

Location	Description	Approximate lat/long (degrees)
1	open sea	27.5545, -96.6907
2	open sea	27.9993, -96.5533
3	Port Aransas	27.8148, -97.0477
4	Aransas Bay	28.0623, -96.9571
5	Copano Bay	28.1107, -97.1054
6	East Corpus Christi Bay	27.8148, -97.1356
7	West Corpus Christi Bay	27.8294, -97.3196
8	Between Corpus Christi and Baffin Bay	27.6130, -97.2547
9	Between Corpus Christi and Baffin Bay	27.3815, -97.3663
10	Baffin Bay	27.2620, -97.5449

Table 6: Locations, description and approximate latitute/longtitude in degrees

figures, we printed solutions at intervals of 200 time steps, the plotting package linearly interpolates between data points.

In Figures 7 and 8, we plot elevation solutions at these 10 locations for runs 1-4 and for the time interval 25 to 30 days. Note that the vertical scale varies in each plot, depending on the magnitude of the solution in each location. At locations 1, 2 and 3, all four solutions are virtually identical. These locations are in the open ocean, near where the tidal boundary condition is specified. Therefore, the elevation amplitude is larger in this region (about .6 meters) than it is in the interior coastal region. Furthermore, baroclinic effects are negligible in this region, so we would not expect to see differences between barotropic and baroclinic solutions. As we move into the bays and interior waterways, we observe reasonable agreement between the elevations at location 6, with the $P0$ barotropic and baroclinic solutions exhibiting higher amplitudes than the $P1$ solutions, on the order of 20%-30% higher. The maximum amplitudes here are around .15 meters. We also observe reasonable agreement at locations 4 and 7, where the amplitudes are around .03 meters, and .06 meters, respectively. The differences in solutions are more pronounced at the other locations. At location 5, we see differences on the order of 50% between $P0$ and $P1$ solutions, here the elevation amplitude is on the order of .03 meters. $P0$ barotropic and baroclinic solutions are virtually identical. There are some differences between linear barotropic and baroclinic solutions, however, location 5 is near a freshwater inflow region so some differences may be expected. The largest differences in elevation amplitudes are seen at locations 8, 9 and 10 in the bottom half of the domain. The amplitudes in these regions are on the order of .03-.05 meters, however, we see large differences between $P0$ and $P1$ baroclinic solutions at these locations. The $P0$ solutions are generally more damped at these locations, which is to be expected. The behavior at location 10 is especially curious, as the $P0$ barotropic and baroclinic, and $P1$ barotropic solutions are very close, while the $P1$ baroclinic solution has much larger amplitude. The reasons for these differences will require further investigation.

Next, in Figures 9 and 10 we compare solutions for the u (x) component of velocity. Again note that the vertical scale varies from plot to plot. As with elevations, all four solutions agree at locations 1 and 2. The largest magnitude of u, around .4 m/s, occurs at locations 3 and 6, where we have flow in a narrow channel between the open ocean and Corpus Christi Bay. Agreement is good at location 3. At location 6, we see some differences in magnitude between $P0$ and $P1$ solutions, with the $P0$ solutions being damped by about 50%. Again, baroclinic effects are not noticeable at these locations, in the sense that we don't see differences between baroclinic and barotropic solutions. Similar behavior is observed at locations 4 and 8. At locations 5, 9 and 10, we observe differences in magnitude between barotropic and baroclinic approximations. in particular the $P1$ baroclinic solution has larger magnitude, and by comparison the $P0$ solutions are damped. The behavior at location 7 is curious, as the $P0$ barotropic and baroclinic solution are quite different from each other and quite different from the $P1$ solutions. This location is very near the ship channel and near the Nueces River inflow, and may be an area where the additional resolution given by linear approximations is needed.

Finally, in Figures 11 and 12 we compare solutions for the v (y) component of velocity. These solutions compare well at locations 1-6. At locations 8-10, the comparisons are reasonable, with the $P0$ solution again more damped compared with the $P1$ solutions and some differences between baroclinic and barotropic. The outlier is again location 7, where the constant baroclinic solution is quite different from the other solutions. However, v is an order of magnitude smaller at this location than at the other locations.

To summarize these results, the baroclinic and barotropic $P0$ and $P1$ elevation and velocity solutions are essentially identical at locations in the open ocean. As we move inland, good qualitative agreement is observed at most locations between the different $P0$ and $P1$ solutions; phase errors are generally not observed but the $P0$ solution is often damped, most likely due to the fact that this approximation is lower order and hence more numerically diffusive. There are some locations, particularly near inflow boundaries, where the differences are larger and the causes need to be further investigated. One possible remedy would be to refine the mesh in these areas.

6.2 Data specified tidal forcing

We added features to the code to allow for open ocean elevation data, wind velocities and evaporation rates to be read in from specified input files. We obtained data files from TWDB. The open sea elevation data was given in 60 second intervals and was assumed to be constant spatially over the boundary. The data for the full 30 days is plotted in Figure 13. For the purposes of testing this feature we ran only a barotropic case, with $P0$ approximations and time step of 1 second. The elevation data was interpolated into 1 second intervals for input to the code.

The elevation and u and v components of velocity, for days 25-30 are plotted in Figures 14-19 at the ten locations listed above. We note that the elevation solutions at locations 1, 2 and 3, which are closest to the open ocean, mimic the elevation forcing data closely. This would indicate that the code is incorporating the data correctly. The velocity is wiggly near the open ocean boundary, also mimicking the roughness in the elevation data; however, it tends to smooth out in the interior of the domain.

6.3 Tides, wind and evaporation

Finally, we give results of simulations with tidal forcing, evaporation and wind. The evaporation and wind data are plotted in Figure 13 for the 30 day period simulated. The evaporation data is from August of 1987. In the figures below, instead of plotting solutions from days 25 to 30 we plot the complete time history of the solutions from 0 to 30 days. We also consider solutions only at locations 1, 3, 4, 6, 7 and 9 in Figure 6. In Figure 20, we plot elevation solutions for tidal forcing only (also given above), tides and wind (no evaporation), and tides, wind and evaporation, at each location. In Figure 21 we plot the x component of velocity. The y component shows similar behavior.

In these figues, we observe that wind forcing has a minimal effect on the water surface elevation over the domain. The tidal signal is slightly depressed at location 4 in Aransas bay, but is virtually indistinguishable from the elvation only solution at all other locations. The axes are different on each plot. The X-velocity solution shows a noticable shift at location 1, and somewhat lesser wind effects at stations 4, 7 and 9. We also note that evaporation has very little effect on the solutions. We remark that these tests are very preliminary and further testing needs to be performed.

7 Conclusions

In this report, we have described recent developments and applications of the simulator UTBEST3D.

The code has been parallelized for small clusters to petascale parallel platforms using domain decomposition and MPI. Studies on the Ranger machine at TACC show that the code exhibits excellent scaled speed-up. The use of parallel computation greatly enhances the capability of the code to perform high resolution, baroclinic simulations in reasonable time.

New features have also been added to the code, including options to read open ocean elevation data, wind velocities, and evaporation rates. Other options will be added during the coming year. The code has been applied to barotropic and baroclinic simulations of the Corpus Christi Bay region, extending from Baffin Bay to Aransas and Copano Bays. Grid, elevation, wind and evaporation data were obtained for these studies from TWDB. For baroclinic and barotropic tide-only simulations, using low order or higher order solutions in the DG discretization gives very similar solutions in the open ocean and near-ocean parts of the domain. In the deeper recesses of the bays and especially near inflow regions, differences are observed in the solutions, and these differences require further investigation. They are most likely due to grid resolution and the inherent numerical diffusion in low order approximations. Barotropic simulations with tidal and wind forcing show that winds have a strong dampening effect on elevations and velocities. Simulations which also included evaporation did not exhibit any significant differences with simulations with no evaporation.

8 Bibliography

References

[1] 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.

[2] V. Aizinger, C. Dawson, *The local discontinuous Galerkin method for three-dimensional shallow water flow*, Comp. Meth. Appl. Mech. Eng., 196, pp. 734–746, 2007.

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

[4] 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.

[5] 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.

[6] 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.

[7] 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.

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

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

[10] G. Karypis and V. Kumar, *A fast and high quality multilevel scheme for partitioning irregular graphs*, SIAM J. Sci. Comp., 20, pp. 359–392, 1999.

[11] http://mason.gmu.edu/ bklinger/seawater.pdf.

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

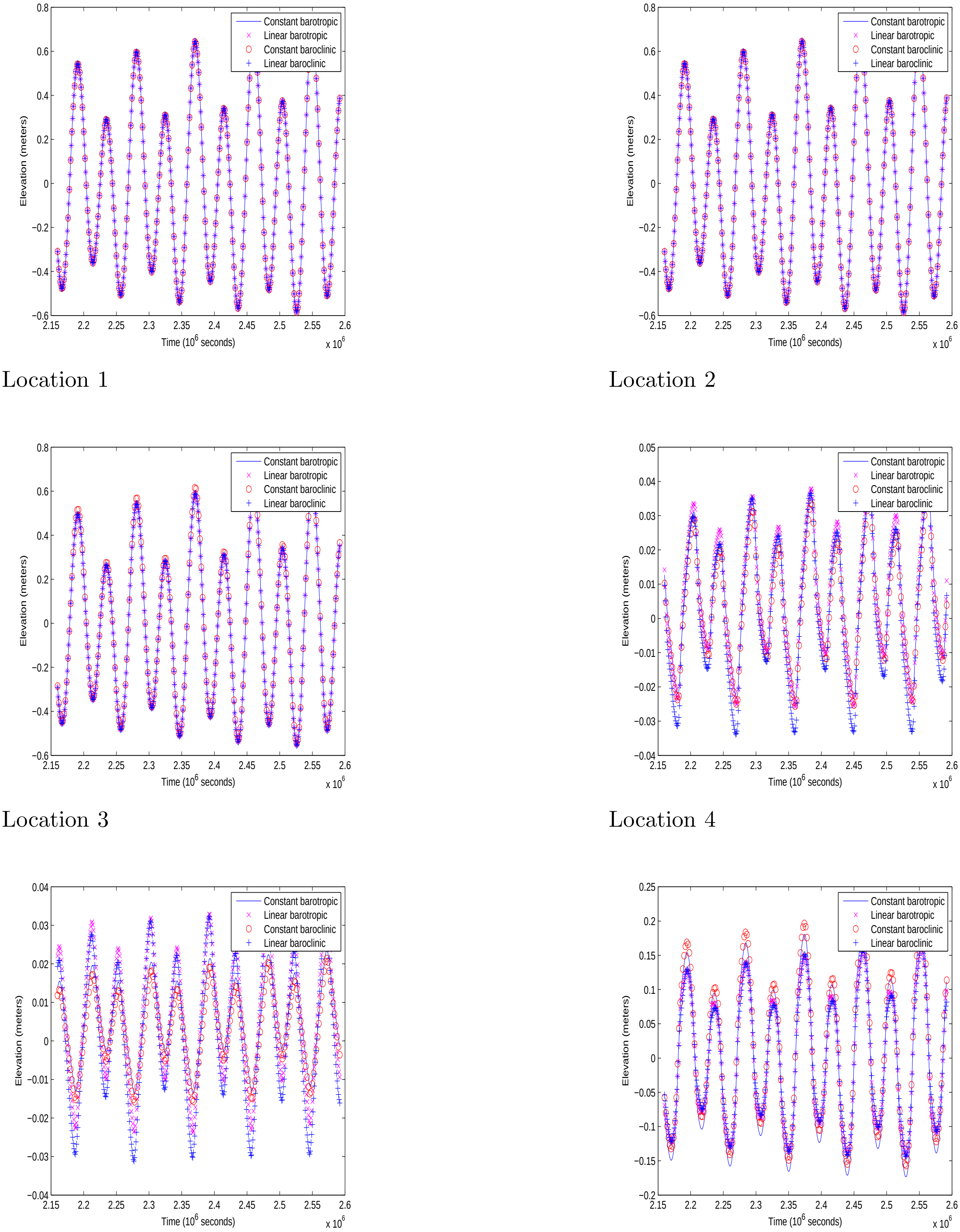
[13] Umlauf, L., Burchard, H., *A generic length-scale equation for geophysical turbulence models*, J. Marine Res., 61, pp 235-265, 2003.

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

[15] 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.

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

[17] Zhang, Y.-L., Baptista, A.M. and Myers, E.P. *A cross-scale model for 3D baroclinic circulation in estuary-plume-shelf systems: I. Formulation and skill assessment*, Cont. Shelf Res. 24, pp. 2187-2214, 2004.



Location 1

Location 2

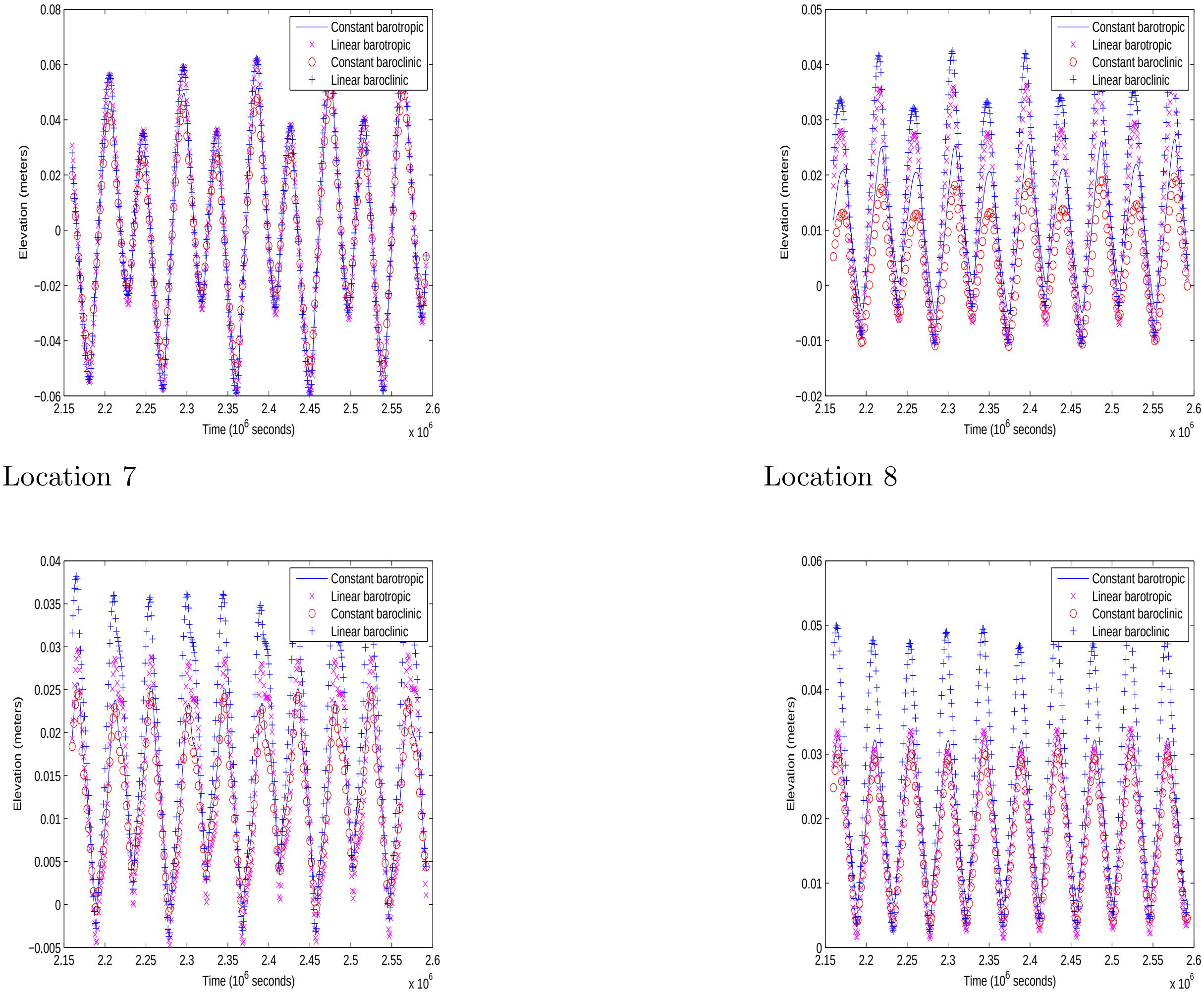
Location 3

Location 4

Location 5

Location 6

Figure 7: Elevation solutions at locations 1-6. Tidal forcing given by (30). Comparison of P^0 and P^1 barotropic and baroclinic solutions for days 25-30.



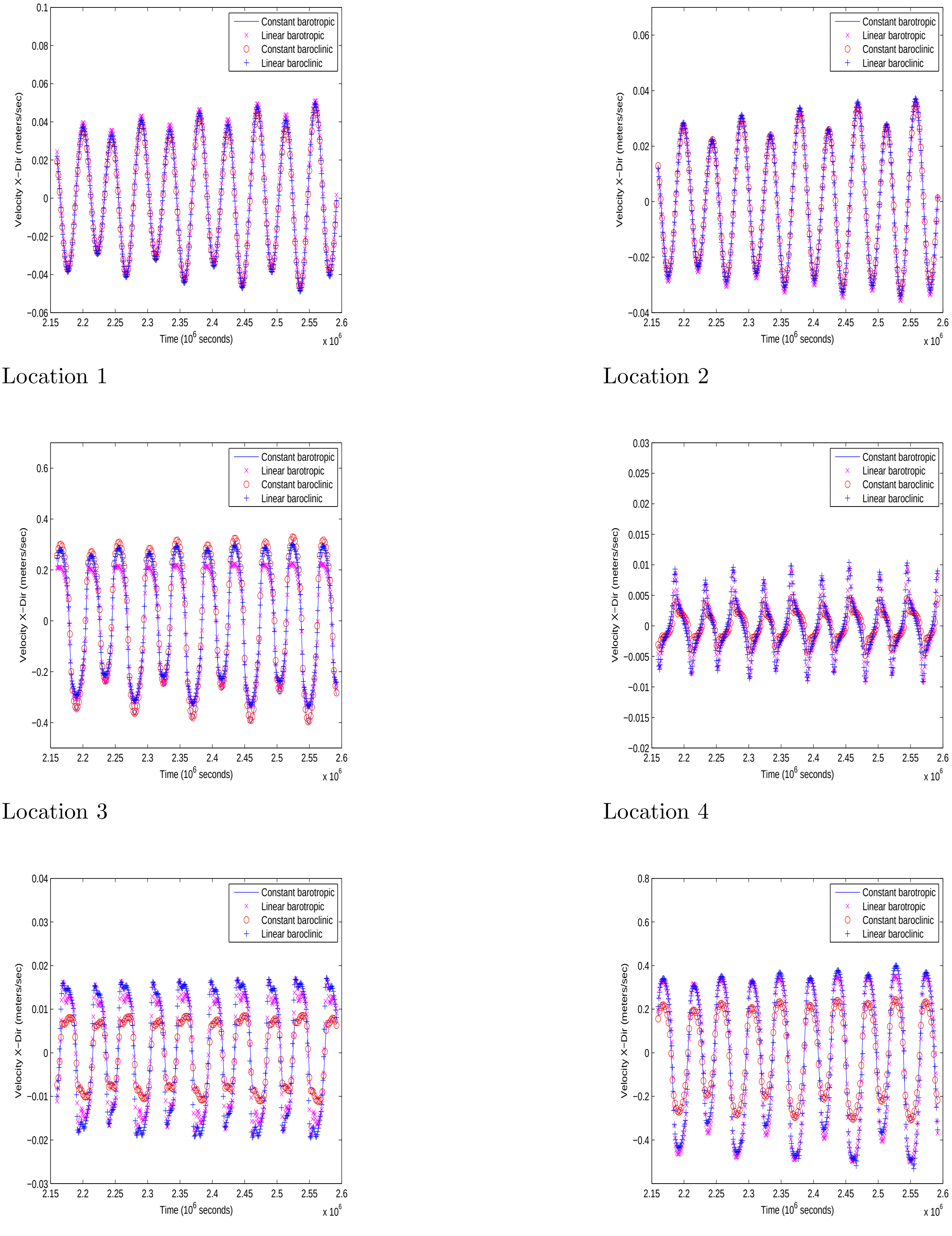
Location 7

Location 8

Location 9

Location 10

Figure 8: Elevation solutions at locations 7-10. Tidal forcing given by (30). Comparison of $P0$ and $P1$ barotropic and baroclinic solutions for days 25-30.



Location 1

Location 2

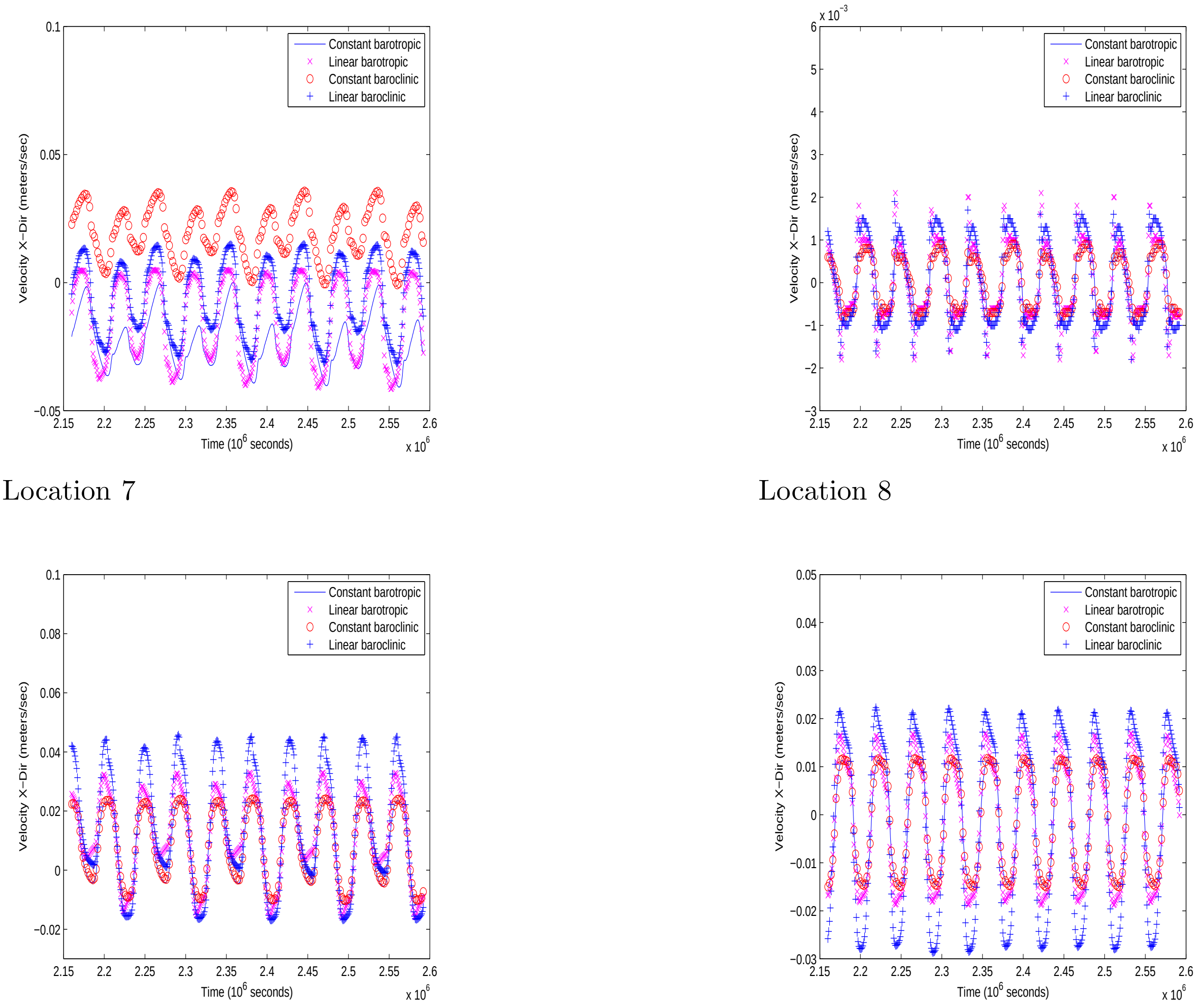
Location 3

Location 4

Location 5

Location 6

Figure 9: Velocity: u component solutions at locations 1-6. Tidal forcing given by (30). Comparison of P^0 and P^1 barotropic and baroclinic solutions for days 25-30.



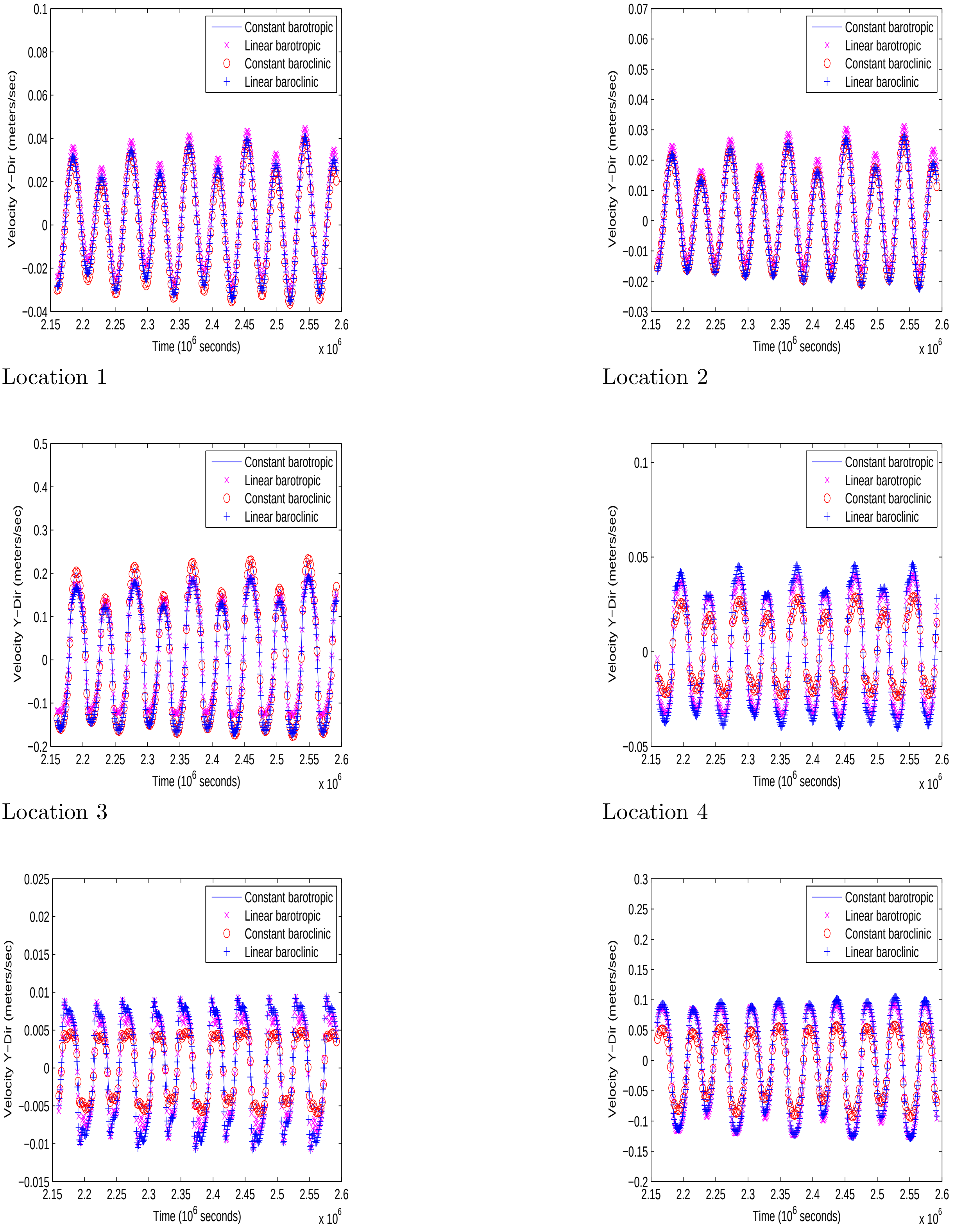
Location 7

Location 8

Location 9

Location 10

Figure 10: Velocity: u component solutions at locations 7-10. Tidal forcing given by (30). Comparison of $P0$ and $P1$ barotropic and baroclinic solutions for days 25-30.



Location 1

Location 2

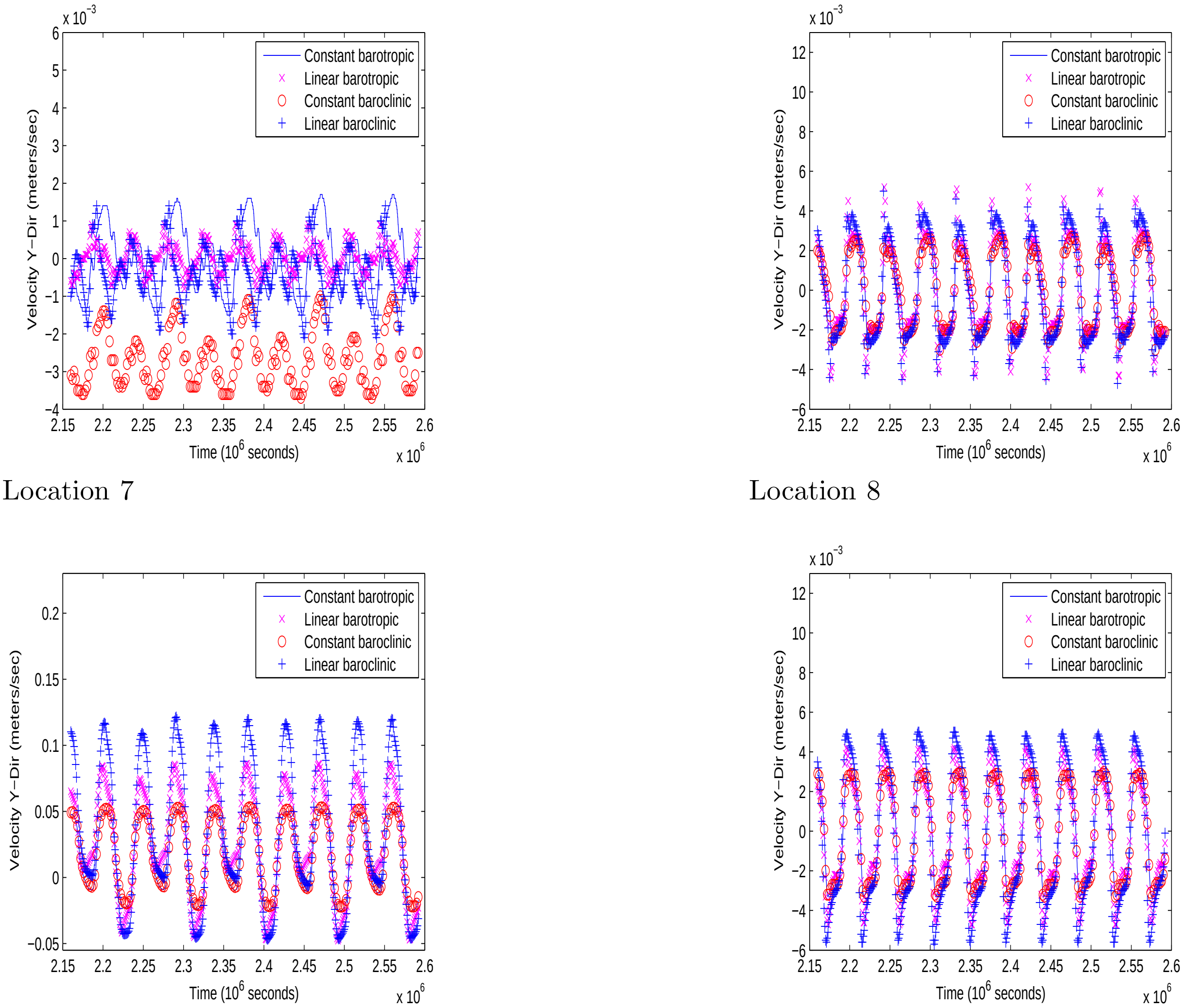
Location 3

Location 4

Location 5

Location 6

Figure 11: Velocity: v component solutions at locations 1-6. Tidal forcing given by (30). Comparison of P^0 and P^1 barotropic and baroclinic solutions for days 25-30.



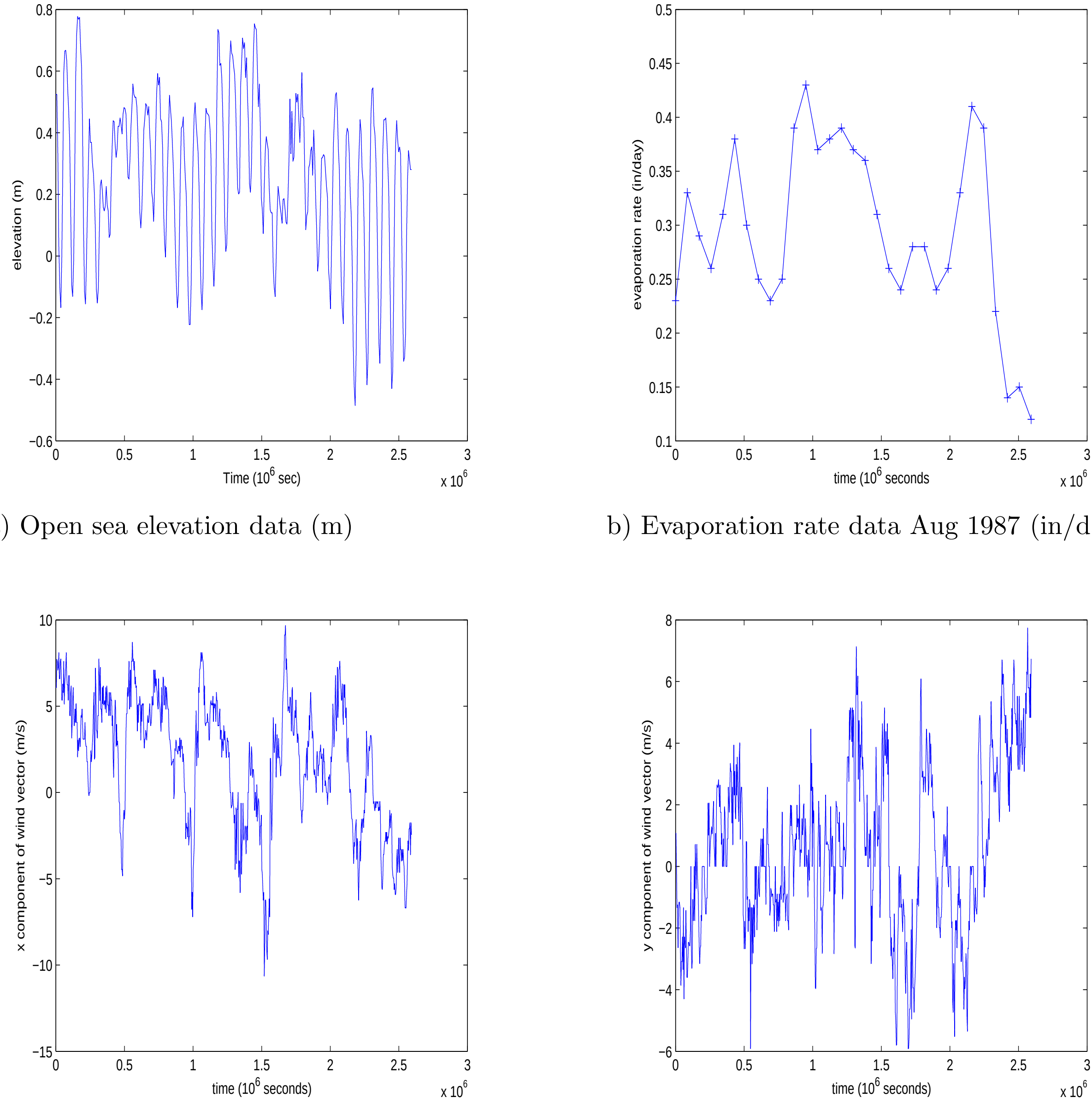
Location 7

Location 8

Location 9

Location 10

Figure 12: Velocity: v component solutions at locations 7-10. Tidal forcing given by (30). Comparison of $P0$ and $P1$ barotropic and baroclinic solutions for days 25-30.



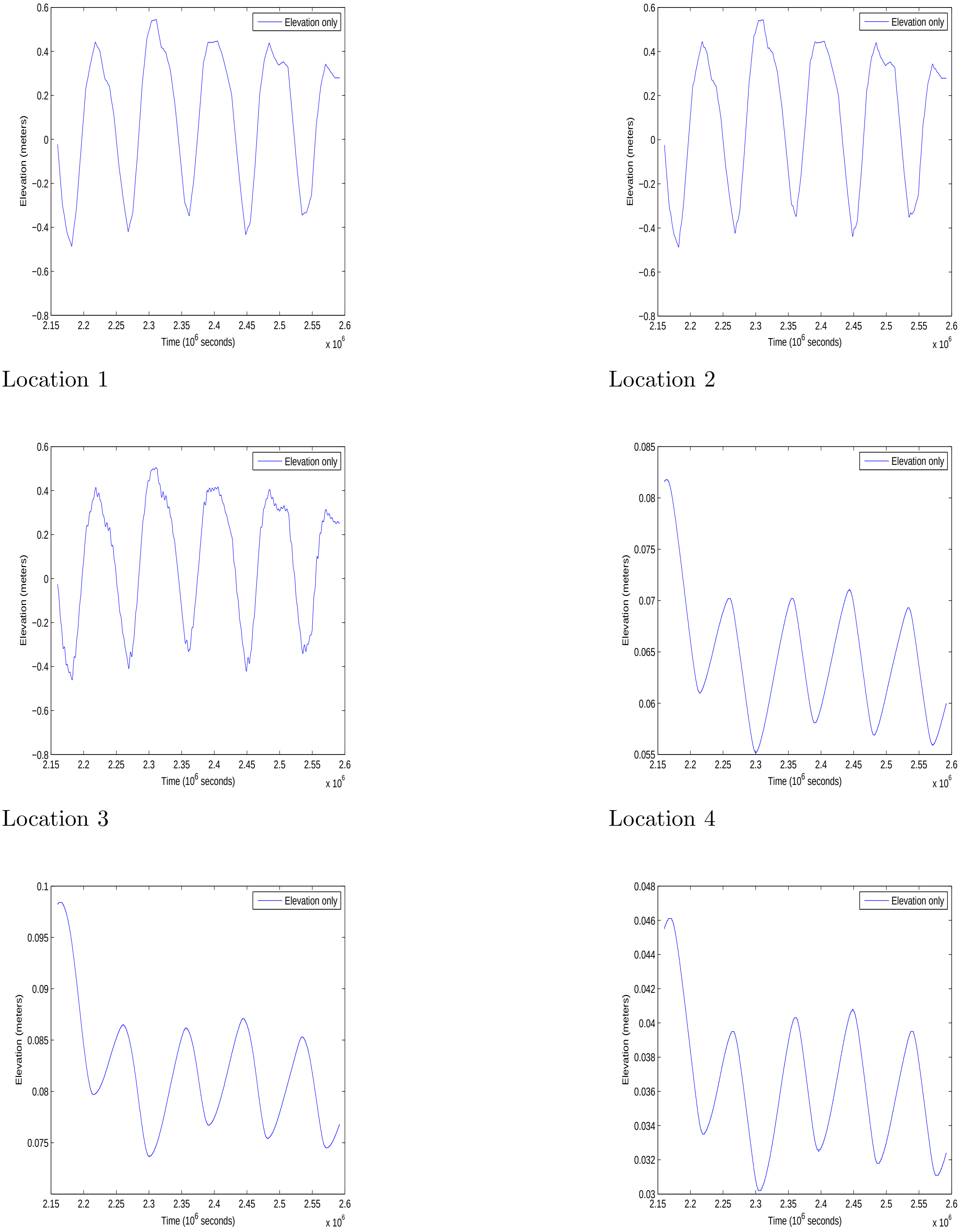
a) Open sea elevation data (m)

b) Evaporation rate data Aug 1987 (in/day)

c) Wind vector–x component (m/s)

d) Wind vector–y component (m/s)

Figure 13: Tidal forcing data, evaporation data, and wind vector data.



Location 1

Location 2

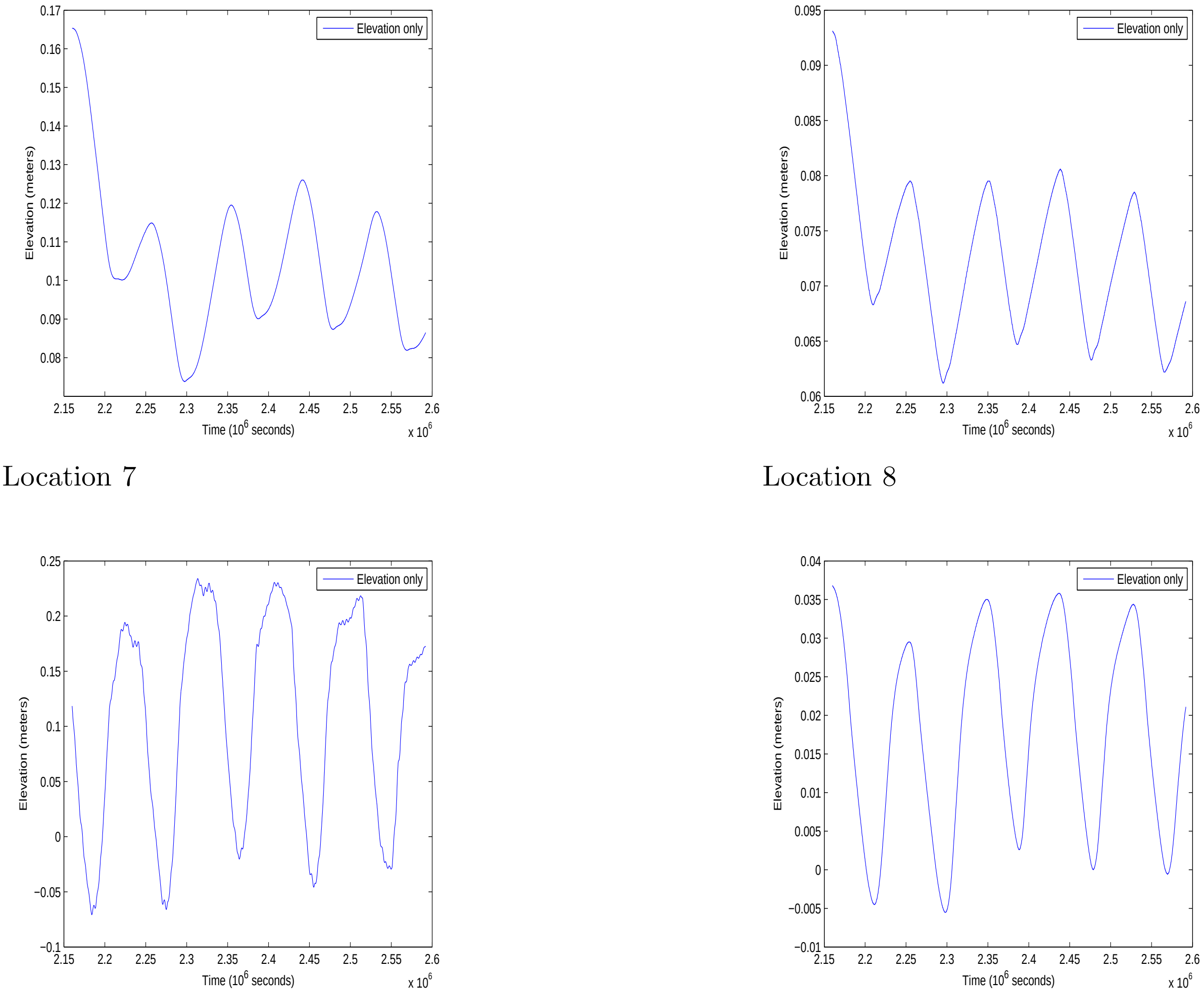
Location 3

Location 4

Location 5

Location 6

Figure 14: Elevation solutions at locations 1-6 for days 25-30. Tidal forcing given by Figure 13.



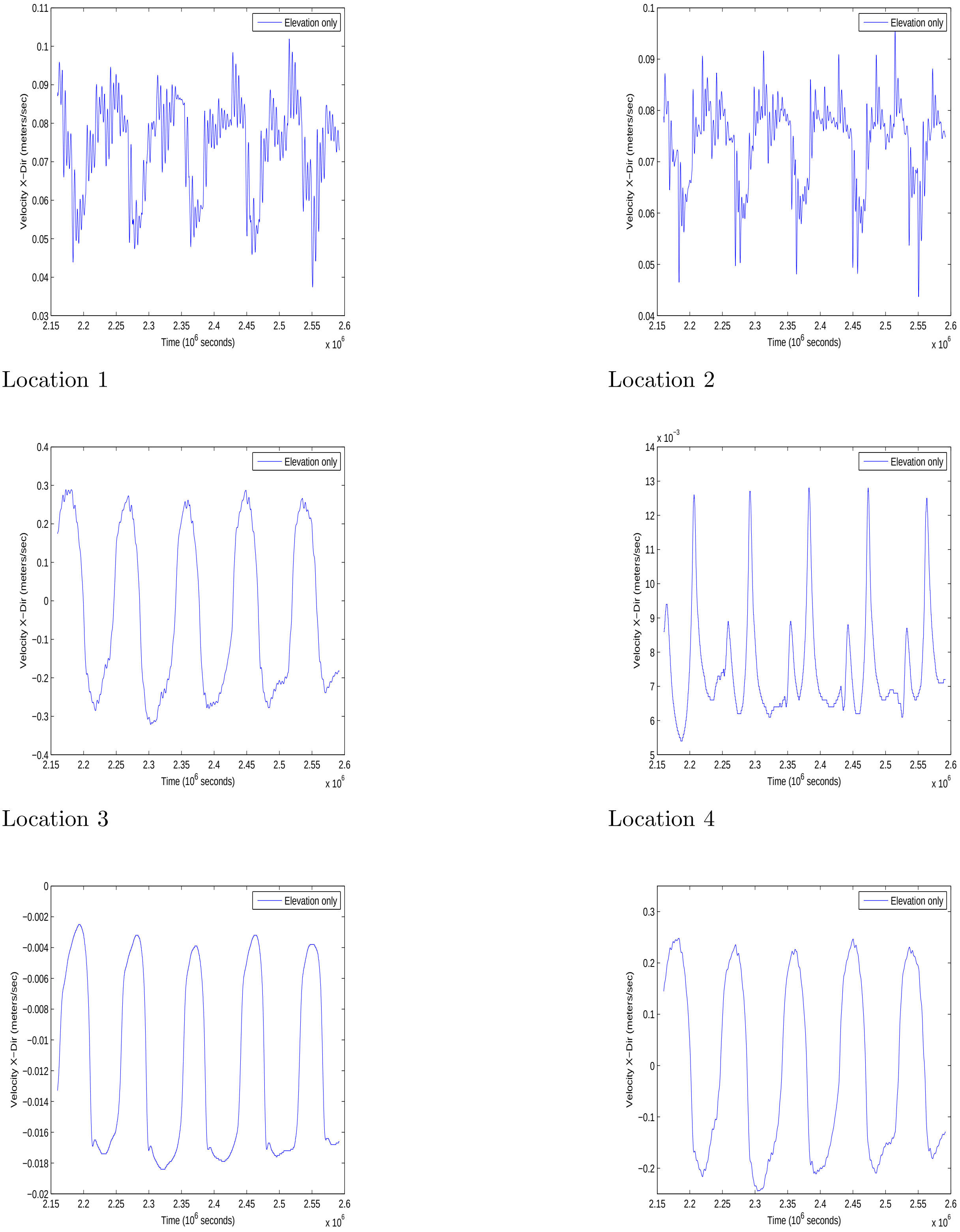
Location 7

Location 8

Location 9

Location 10

Figure 15: Elevation solutions at locations 7-10 for days 25-30. Tidal forcing given by Figure 13.



Location 1

Location 2

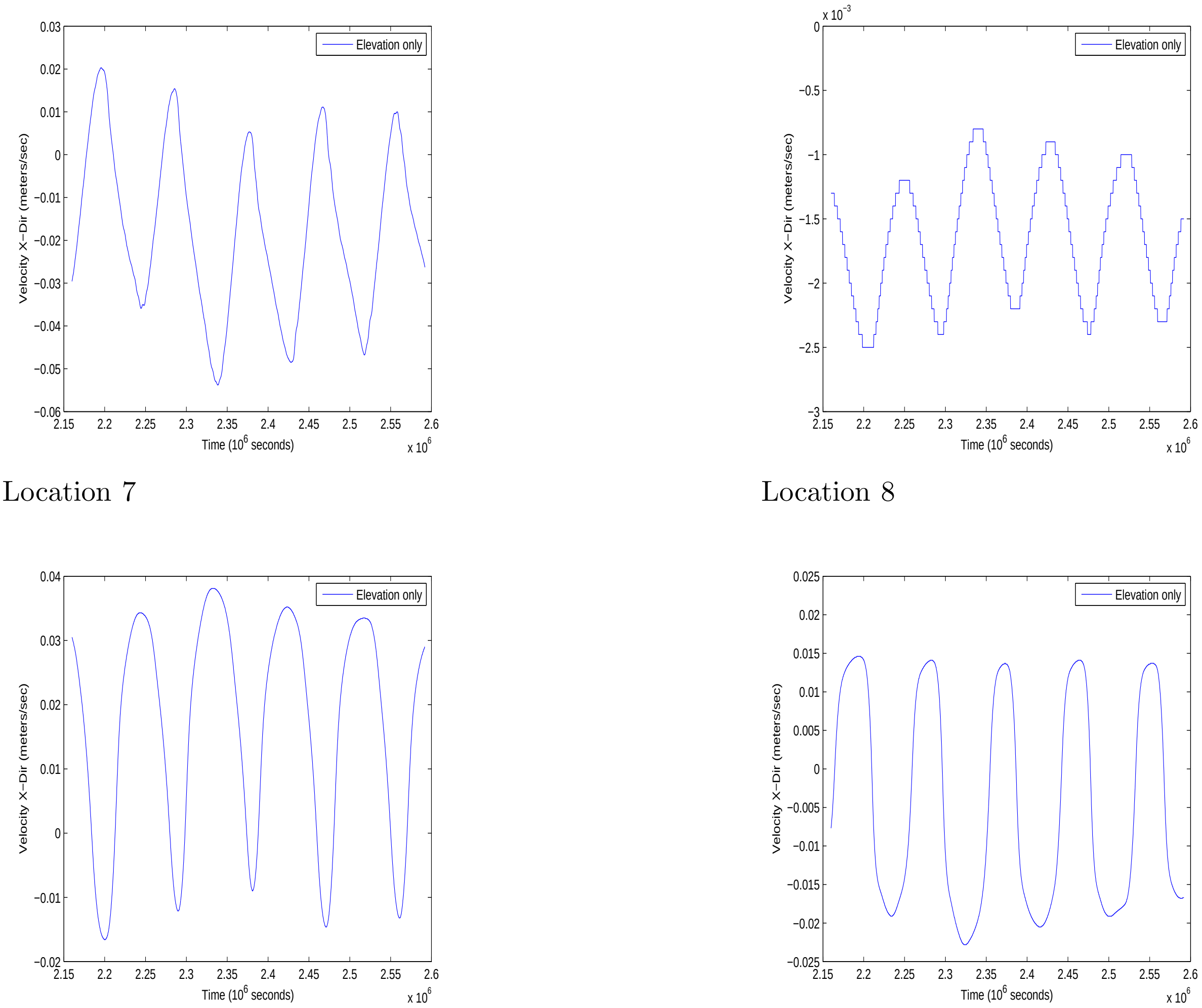
Location 3

Location 4

Location 5

Location 6

Figure 16: Velocity: u component solutions at locations 1-6 for days 25-30. Tidal forcing given by Figure 13.



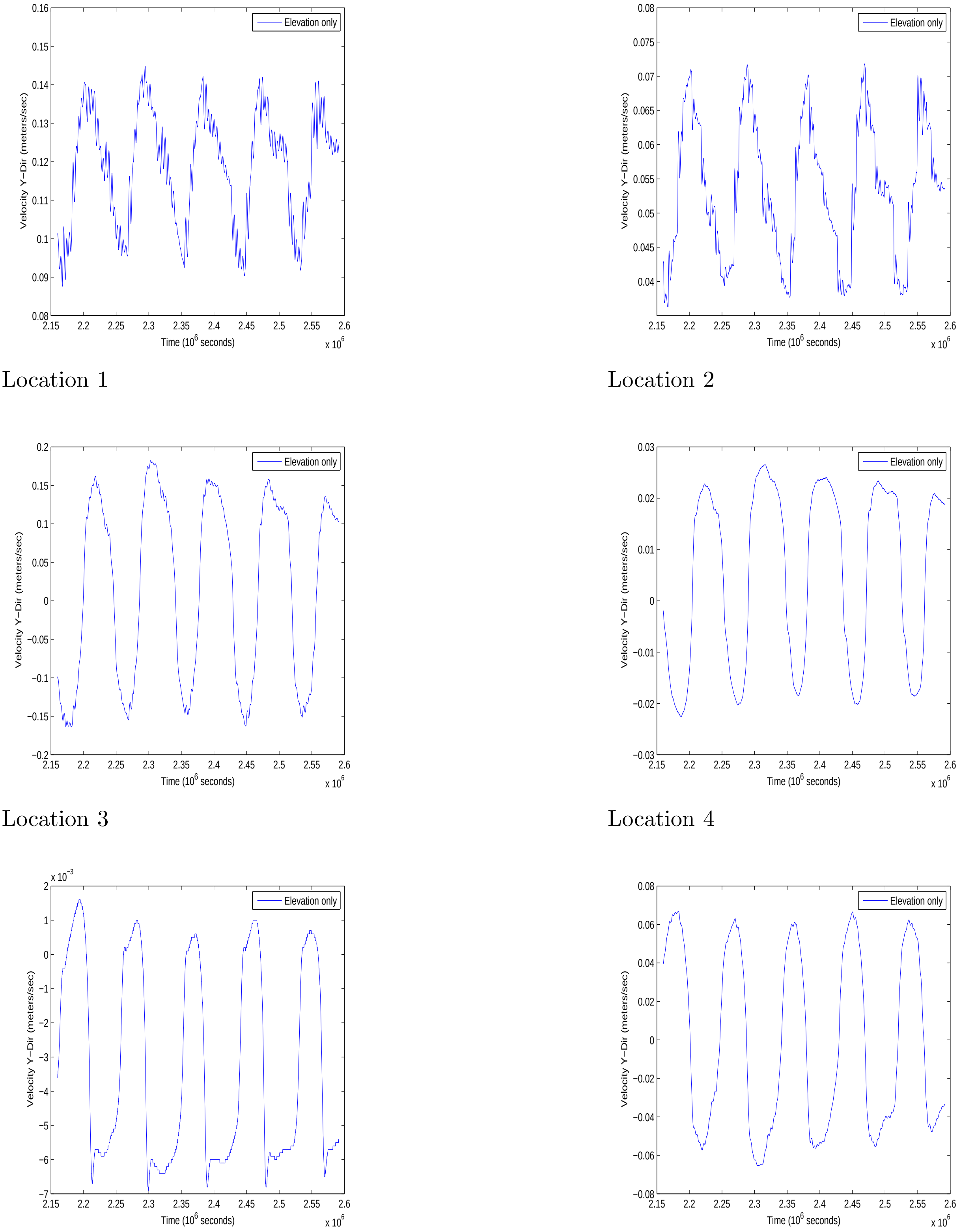
Location 7

Location 8

Location 9

Location 10

Figure 17: Velocity: u component solutions at locations 7-10 for days 25-30. Tidal forcing given by Figure 13.



Location 1

Location 2

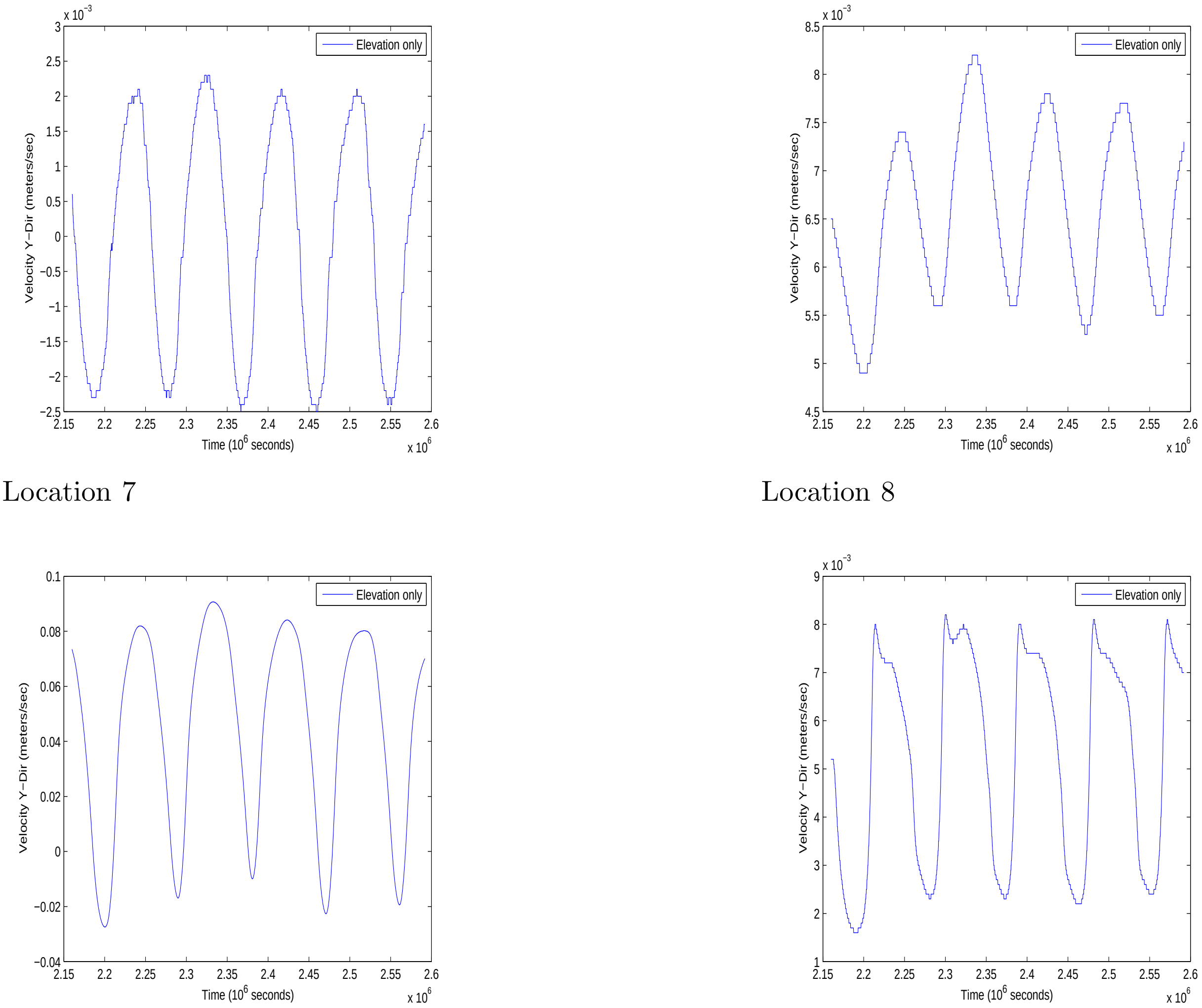
Location 3

Location 4

Location 5

Location 6

Figure 18: Velocity: v component solutions at locations 1-6 for days 25-30. Tidal forcing given by Figure 13



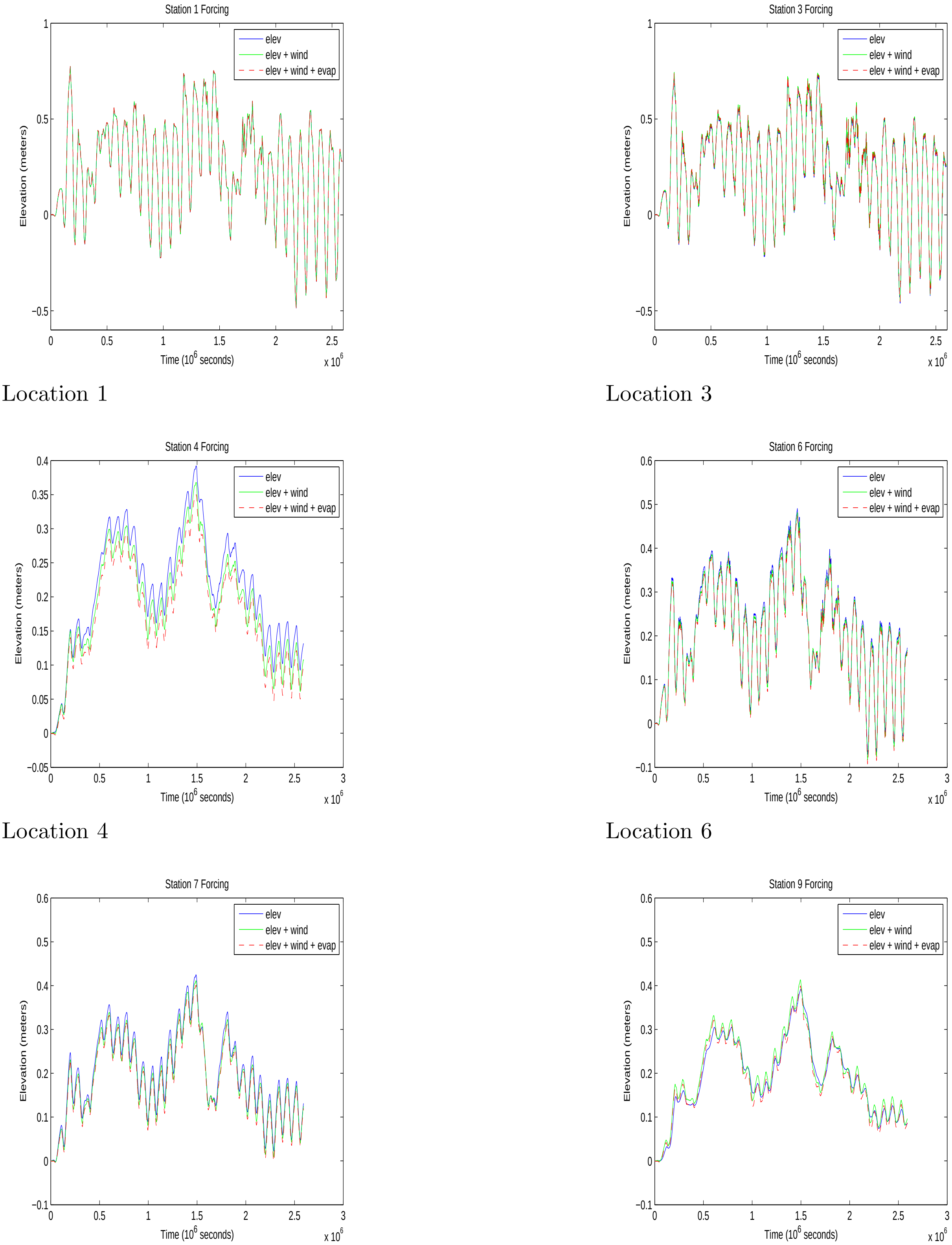
Location 7

Location 8

Location 9

Location 10

Figure 19: Velocity: v component solutions at locations 7-10 for days 25-30. Tidal forcing given by Figure 13.



Location 1

Location 3

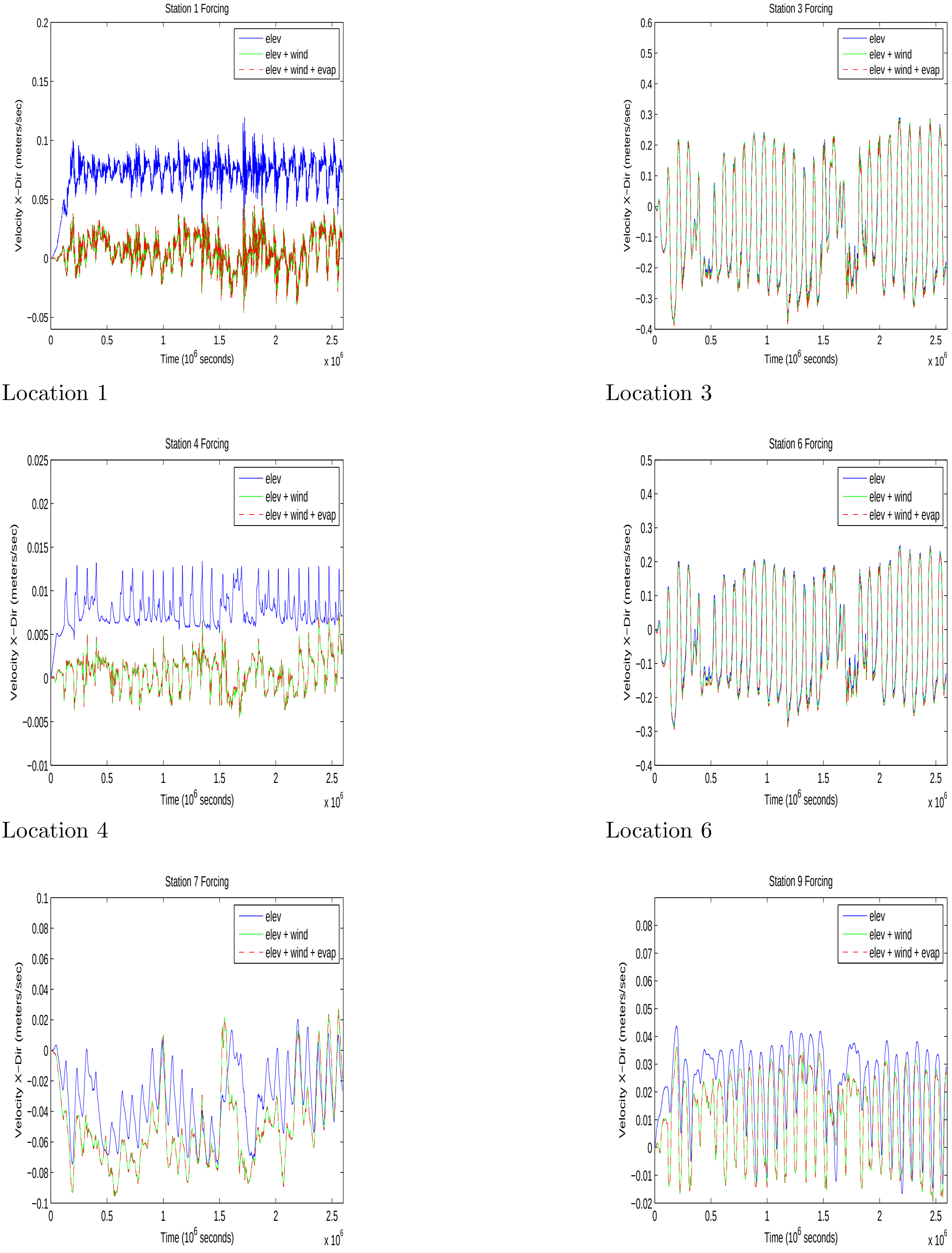
Location 4

Location 6

Location 7

Location 9

Figure 20: Elevation solutions at locations 1, 3, 4, 6, 7 and 9 for 0 to 30 days. Comparison of solutions with tidal forcing only, tides plus winds, and tides, winds and evaporation.



Location 1

Location 3

Location 4

Location 6

Location 7

Location 9

Figure 21: Velocity (x component) solutions at locations 1, 3, 4, 6, 7 and 9. Comparison of solutions with tidal forcing only, tides plus winds, and tides, winds and evaporation.