CFD Modeling in the San Francisco Bay and Delta

E. Ateljevich * P. Colella† D.T. Graves ‡ T.J. Ligocki§ J. Percelay ¶
P.O. Schwartz‖ Q. Shu **

Abstract

We describe a two-dimensional shallow water model whose initial implementation simulates flows in the San Francisco Bay and Sacramento-San Joaquin Delta. This model, called REALM, is based on a Cartesian grid, embedded boundary discretization of the shallow water equations. We employ parallel computation and adaptive mesh refinement for rapid computation. Grid generation from digital elevation models faithfully represents the shoreline and the use of implicit functions and constructive solid geometry permits the representation of structures such as gates.

1 Introduction

Motivation for modelling the San Francisco Bay-Delta arises from water management concerns such as the rapid decline of pelagic organisms, increased demand for water, pollution from agricultural run-off, and the danger from levee breaks and island flooding. Policy makers ask difficult hydrodynamic, water quality, and biological questions that traditional computer models are not able to answer sufficiently quickly, flexibly or accurately.

Research conclusions lead to policy decisions through an open, public review process, which is facilitated by the use of open-source models. Specific research questions can involve large time and spatial scales or require detailed modeling involving complicated topograpghy and man-made structures. Static unstructured grid models have been applied to this domain for decision support in [10] using the pressure correction algorithm in [5].

The desire for a reliable, extensible open-source model employing high performance numerical algorithms has led the Department of Water Resources (DWR) to initiate a collaboration with Lawrence Berkeley National Lab and UC Berkeley to produce a high-performance computer model that can address the pressing questions outlined above. We call this model REALM (River, Estuary, And Land Model). The DWR has identified core physical processes and scenarios in the Bay-Delta that will be useful to simulate, which we describe in the next section. In the remainder of the paper we describe the extent of our progress using finite volume methods with adaptive Cartesian grids and embedded boundaries to satisfy these modeling requirements.

2 Physical Processes in SF Bay and Delta

Useful models must be capable of simulating shallow water flow and transport in an estuary. The shallow water equations include terms representing advection, gravity waves, point sources of mass, friction, wind, diffusion, and the effect of salt concentrations on density. Boundary conditions as well as friction and dispersion coefficients may all vary with space and time.

The Bay-Delta domain also contains certain regions, such as the Delta Cross Channel, Stockton Ship Channel and Carquinez Straits, where flow, water quality and particle trajectories are significantly affected by secondary circulations and stratification. Selective modeling in three-dimensions will sometimes be necessary. Similarly, in many regions the Delta flow is channelized and essentially one dimensional. The need for long simulation times over large spatial scales also requires the ability to selectively model 1D flows. Intertidal habitat modeling, flooding and sediment transport require the ability to model moving shorelines. Contaminant transport concerns require modeling the advective and diffusive transport of passive conservative constituents and extensibility to non-conservative reactions. Finally, data assimilation, ecological applications, and some types of sediment transport require the capability to represent the transport of particles, including particles exhibiting behavior.

These concerns shaped the following goals for the model:

- Simulation of the entire Bay-Delta at once.
- Simulations of one, two and three dimensional

*California Department of Water Resources
†Lawrence Berkeley National Lab
‡Lawrence Berkeley National Lab
§Lawrence Berkeley National Lab
¶L'Ecole Polytechnique
‖Lawrence Berkeley National Lab
**California Department of Water Resources

flows.

- Salinity and contaminant modeling.
- Simulation of tidal mud flats.
- Simulation of levee breaks, floods, and flooded islands.
- Faithful representation of gates, structures, and shoreline.
- Rapid grid generation.
- Simulation of the movements of fish eggs and young fish.

These goals led to the following design decisions:

- Embedded boundary methods.
- Grid generation from implicit functions.
- Conservative discretization.
- Adaptive mesh refinement.
- Parallel computation.

Embedded boundary methods have been in existence since the 1960s; for a discussion of some of the history, see [6]. Embedded boundary (EB) methods use a cut-cell discretization that faithfully represents the bathymetry and natural shorelines. Grid generation from implicit functions permits rapid prototyping of new problems from digital elevation models (DEMs) and the use of constructive solid geometry to represent detailed structures such as gates. Conservative discretizations remain robust in marginally resolved calculations that are typical in long-term studies. Embedded boundaries characterize the domain boundary independently of the Cartesian mesh; hence, our model is extensible to moving boundary problems such as island flooding and intertidal wetting and drying.

Adaptive mesh refinement (AMR, [3], [2]) focuses computational resources on areas of interest and parallel computation increases the computational resources for a given problem. Refinement can be static or dynamic – adapting to transient phenonmena such as floods or contaminant fronts. Our design decisions are consistent with a particle model that simulates the passive advection of fish eggs or sensors as well as simple behavior characteristics of young fish or active sensors. See Figure 1 for an illustration of how an adaptive mesh covers the San Francisco Bay and Western Delta and Figure 2 for a close up of AMR results with passive particle modeling.

A key facet of the project is the efficient reuse of established algorithms in a new applied setting. Adaptive mesh refinement requires careful implementation and load balancing when implemented on parallel, distributed memory systems [12]. In our AMR algorithm, data must be interpolated, coarsened and moved efficiently between multi-block elements at different refinement levels. Embedded boundaries complicate the data movement when non-conservative mass is redistributed [8] and EB algorithms require a description of the topology of the unstructured cut cells. These grid-related requirements are common across EB-AMR applications in fields as diverse as estuary modeling, gas dynamics, biology and astrophysics. EB-AMR would be expensive and redundant for modelers in individual research domains to replicate and test in software. Recognizing this, the LBNL partners in the present collaboration have successfully abstracted the EB-AMR components into a solver infrastructure (Chombo) independent of the estuary physics. The present collaboration is able to leverage these investments and serves as a test of the generalization in an applied setting with numerous practical complications.

3 Shallow Water Equations

We plan to integrate the equations for the conservation of mass and momentum of water, the advection and diffusion of salt and other constituents, and the movement of neutrally buoyant particles. Our initial model is two dimensional. Our shallow water formulation is based on the depth-integrated Navier-Stokes equations, with a hydrostatic treatment of pressure, Boussinesq assumption concerning salt-induced horizontal (baroclinic) density variation and friction. The formulation presented here defers the inclusion of diffusion in order to concentrate on the effects of geometry.

The problem domain is an estuary system of bays and channels. The water column is delineated on the bottom by a rigid boundary of known bathymetry and on the top by a free surface. For the present, the shoreline and the bottom do not change with time. Ultimately, the design anticipates a shoreline that evolves in time due to tides and flooding. The treatment at the boundaries includes the effect of three dimensional geometry on the two dimensional flows.

3.1 Conservation equations In terms of the height of the water column h, local velocities u and v and salt concentration s, the equations in conservation form are:

$$\frac{\partial U}{\partial t} + \frac{\partial F^x + F_d^x}{\partial x} + \frac{\partial F^y + F_d^y}{\partial y} = S, \tag{3.1}$$

where the vector of conserved variables:

$$U = \begin{pmatrix} h \\ hu \\ hv \\ hs \end{pmatrix} \tag{3.2}$$

represents the water mass/height h, momentum hu and hv and mass of salt per unit area hs. The flux across cell faces in the x- and y-directions combine the convective acceleration and gravity wave terms:

$$F^x = \begin{pmatrix} hu \\ hu^2 + \frac{g\rho h^2}{2\rho_0} \\ huv \\ hus \end{pmatrix} \text{ and} \tag{3.3}$$

$$F^y = \begin{pmatrix} hv \\ huv \\ hv^2 + \frac{g\rho h^2}{2\rho_0} \\ hvs \end{pmatrix}. \tag{3.4}$$

In these equations, g denotes the gravitational constant, ρ_0 denotes the density of fresh water, and

$$\rho = \rho(s(x, y, t))$$

denotes the density of salt water. The diffusive terms F_d^x and F_d^y combine momentum diffusion due to eddy diffusivity and a more general dispersion term for salt (using a diffusion constant D):

$$F_d^x = \begin{pmatrix} 0 \\ -\epsilon h \frac{\partial u}{\partial x} \\ -\epsilon h \frac{\partial v}{\partial x} \\ -Dh \frac{\partial s}{\partial x} \end{pmatrix} \text{ and} \tag{3.5}$$

$$F_d^y = \begin{pmatrix} 0 \\ -\epsilon h \frac{\partial u}{\partial y} \\ -\epsilon h \frac{\partial v}{\partial y} \\ -Dh \frac{\partial s}{\partial y} \end{pmatrix}. \tag{3.6}$$

The sources and sinks include the pressure component due to changes in bottom elevation, friction and any other local sources of mass or stress such as wind. Here we focus on bottom pressure and friction:

$$S = \begin{pmatrix} 0 \\ -\frac{g\rho}{\rho_0} h b_x - \tau_x \\ -\frac{g\rho}{\rho_0} h b_y - \tau_y \\ 0 \end{pmatrix}, \tag{3.7}$$

where $b(x, y)$ is the elevation of the bed and τ_x is a bottom stress given by a zero-order closure relation using the Manning's friction coefficient [4]:

$$\tau_x = \frac{\rho g h n^2}{\rho_0 h^{\frac{4}{3}}} u\sqrt{u^2 + v^2},\ \tau_y = \frac{\rho g h n^2}{\rho_0 h^{\frac{4}{3}}} v\sqrt{u^2 + v^2}.$$

We use a split treatment of the hyperbolic and diffusive components. First we solve the hyperbolic component of the equations, referencing the diffusive flux F_d only in a predictor steps involving Taylor expansions in space and in time. Subsequently, we use the hyperbolic advance as a source in the elliptical solver for diffusion. The presentation here emphasizes the hyperbolic component, which dominates in estuary flow.

3.2 Primitive form It is common to use "primitive" variables $W = W(U)$ that make the problem smoother or simplify the characteristic structure of the equations.

For shallow water, we consider the change of variables

$$W = [\delta, u, v, s]^T,$$

where δ is a perturbed surface above an unperturbed reference depth h_0 as shown in Figure 3. The primitive variable δ is related to depth by $h = h_0 + \delta$. One advantage of the perturbation formulation is that the water surface is generally smoother than the depth h. A second advantage is that h_0 generalizes in a way that will allow us to introduce aspects of the 3D geometry in the 2D discretization.

The primitive formulation can be derived by premultiplying the conservation equations 3.1 by $\nabla_U W$, and expanding $\frac{\partial F^d}{\partial x}$ using the chain rule:

$$\frac{\partial F^d(W, h_0)}{\partial x} = \nabla_U F^d \cdot \nabla_W U \frac{\partial W}{\partial x} + F^d_{h_0} \frac{\partial h_0}{\partial x},$$

where $F^d_{h_0} = \frac{\partial F^d}{\partial h_0}$, where the flux may have an explicit spatial dependence on geometry.

The substitution leads to a quasi-linear system:

$$\frac{\partial W}{\partial t} + A^x \frac{\partial W}{\partial x} + A^y \frac{\partial W}{\partial y} = \widetilde{S}, \tag{3.8}$$

$$A^d = \nabla_U W \cdot \nabla_U F^d \cdot \nabla_W U, \tag{3.9}$$

$$\widetilde{S} = \nabla_U W \cdot (S - F^x_{h_0} \frac{\partial h_0}{\partial x} - F^y_{h_0} \frac{\partial h_0}{\partial y}). \tag{3.10}$$

The matrices A^d for each coordinate direction d are given by:

$$A^x = \begin{pmatrix} u & h & 0 & 0 \\ \frac{c^2}{h} & u & 0 & \frac{\rho'}{2\rho} c^2 \\ 0 & 0 & u & 0 \\ 0 & 0 & 0 & u \end{pmatrix}, \tag{3.11}$$

$$A^y = \begin{pmatrix} v & 0 & h & 0 \\ 0 & 0 & v & 0 \\ \frac{c^2}{h} & v & 0 & \frac{\rho'}{2\rho} c^2 \\ 0 & 0 & 0 & v \end{pmatrix}. \tag{3.12}$$

Here ρ' denotes the change in density with respect to a change in salt concentration and $c = \sqrt{\frac{\rho}{\rho_0} gh}$ is the celerity of a shallow water gravity wave.

We use the primitive variables mostly in the predictor step of our algorithm, in which we extrapolate variables from cell centers to cell faces at the mid-point of the time step using a Taylor series. We also use the matrix A^d to derive approximate Riemann solvers, which determine the upwind fluxes across cell faces.

3.3 Boundary conditions The most common open water boundary conditions for shallow water are prescribed (Dirichlet) water surfaces, boundary velocities and salt concentrations all of which are typically time-varying. Boundary conditions at shores are reflective, with zero flow normal to the boundary. Extrapolation or radiation boundary conditions arise in problems where the effect of the far-field is not meant to influence the interior of the domain. Hydraulic devices such as weirs and culverts can be expressed as functions of water surface and flow affecting two boundaries simultaneously.

We assume "subcritical" flow at the boundaries, so that the normal-direction hydrodynamics develop one incoming and one outgoing characteristic and hence one boundary condition. In addition, the passive transport of salt and transverse velocity each require boundary information when flow is directed into the domain. We enforce the boundary conditions numerically using the method of partial Riemann problems. A Riemann state at $x/t = 0$ is chosen that conforms to the known information and that satisfies the Riemann wave relations across the waves that are directed into the domain.

4 Discretization

We use a finite volume discretization of the shallow water equations, based on a Cartesian grid with embedded boundaries. A primary new feature of the spatial discretization is the representation of spatial variation in bathymetry and consideration of this variation in the calculation and interpolation of cell and face-averaged quantities. Our choice of discretization allows features that arise from a three dimensional geometry (i.e., sloping bottoms and shorelines) to be represented within a 2D domain and computational scheme. The design is compatible with moving shorelines as well as a mixed one, two and three dimensional model. The scheme requires spatially varying parameters representing the geometry terms and a computational geometry capability for computing these parameters (see section 6).

Our control volume is divided by a reference surface into the two regions suggested by Figure 4. The shoreline is the intersection of a reference surface and the bathymetry $b(x, y)$. Below the reference surface, the geometry is detailed and 3D; above it is the perturbation region defined by δ and approximated by a rectangle.

4.1 Geometry notation In addition to the notation used in the shallow water formulation above, we will use the following notation to discretize the flow equations over cells with coordinate-aligned faces and embedded boundaries:

- ζ^x, ζ^y, ζ^B and ζ^c denote an average depth over a coordinate-aligned face, EB face or over a volume, expressed as a fraction of a maximum depth, H. Hence the area of a face normal to the x direction would be given by $A = \zeta^x H \Delta x$ and the volume of fluid in a cell is given by $V = \zeta^c H \Delta x^2$. $h_0 = \zeta H$ is the generalization of h_0 alluded to in section 3.2.

- $< z >_x$ and $< z >_y$ and $< z >_B$ denote the $z-$component of the centroid of a vertical face in the x and y directions or EB, measured from the datum indicated in Figure 4.

- n_x^B and n_y^B denote the components of the normal to the 2D embedded boundary, evaluated at the centroid of the boundary.

4.2 Flux discretization We define the fluxes by integrating two dimensional mass and momentum transport as well as pressure forces across the faces of a single layer of three dimensional cells bounded above and below by the free surface and bottom.

To make the following formulae easier to read, we define:

$$q_x = \frac{g\rho}{\rho_0}(\delta - < z >_x) H \zeta^x, \tag{4.13}$$

$$q_y = \frac{g\rho}{\rho_0}(\delta - < z >_y) H \zeta^y, \text{and} \tag{4.14}$$

$$q_B = \frac{g\rho}{\rho_0}(\delta - < z >_B) H \zeta^B, \tag{4.15}$$

which represent the average hydrostatic pressure forces on the portions of faces below the reference surface. The hydrostatic pressure in the δ region is included separately. Given this notation, the discrete fluxes are:

$$F^x = \begin{pmatrix} (\delta + \zeta^x H)u \\ (\delta + \zeta^x H)u^2 + q_x + \frac{\delta^2}{2}] \\ (\delta + \zeta^x H)uv \\ (\delta + \zeta^x H)us \end{pmatrix}, \tag{4.16}$$

$$F^y = \begin{pmatrix} (\delta + \zeta^y H)v \\ (\delta + \zeta^y H)uv \\ (\delta + \zeta^y H)v^2 + q_y + \frac{\delta^2}{2}] \\ (\delta + \zeta^y H)vs \end{pmatrix} \text{ and} \tag{4.17}$$

$$F^B = \begin{pmatrix} (\delta + \zeta^B H)un_x^B + (\delta + \zeta^B H)vn_x^B \\ (\delta + \zeta^B H)u[un_B^x + vn_B^y] + q_B + \frac{\delta^2}{2}]n_x^B \\ (\delta + \zeta^B H)v[un_B^x + vn_B^y] + q_B + \frac{\delta^2}{2}]n_y^B \\ (\delta + \zeta^B H)s[un_B^x + vn_B^y] \end{pmatrix}. \tag{4.18}$$

4.3 Bottom source The source components $-\frac{g\rho}{\rho_0}hb_x$ and $-\frac{g\rho}{\rho_0}hb_y$ in the original PDE can be interpreted as representing bottom pressure. Bottom pressure must be discretized in such a way that free-stream conditions ($[\delta, u, v, s]$ = constant) are preserved in a frictionless flow where the bed does not vary along streamlines.

Our discretization is based on this balance: we require that the pressure forces normal to the bottom in the x and y directions exactly balance the pressure forces on the faces of the computational cell under the conditions that the water surface is level and velocity is zero. For each flux term in 4.16, 4.17, and 4.18 we define an analogous face contribution by substituting a cell averaged $\delta = \delta_c, u = 0, v = 0$. We difference these contributions using the same divergence representations we use for the face fluxes. Our approximation is consistent with the source terms $-\frac{g\rho}{\rho_0}hb_x$ and $-\frac{g\rho}{\rho_0}hb_y$ in the original PDE and the algorithm preserves free-stream.

5 Solution Algorithm

Our time update is a finite volume predictor-corrector method: We construct accurate, upwinded estimates of the fluxes on cell faces and then update cell average values. Specifically, we employ the solution algorithm first used in [8], which itself was based on earlier work [7]. The technique requires the following steps:

1. Extrapolation in space and time of variables from cell centers to edge centers, momentarily neglecting the contribution of the transverse component of the operator, which results in dual "high" and "low" side estimates of the primitive variables at the cell faces.

2. Solution of a Riemann problem which converts the dual estimates of extrapolated variables to upwind fluxes. We use the primitive solver based on the linearized problem as described in [13]. The solution is modified to include salinity-induced density variation.

3. Differencing of transverse fluxes to obtain a correction corresponding to the previously neglected transverse component of the operator, which is applied to the original (dual) edge-centered primitive variable estimates. This again results in dual estimates, which are resolved by solving another Riemann problem. At this point we have edge-centered, time-centered estimates of the primitives and fluxes on regular cells.

4. At cut cells, where the shoreline interesects the computational cell, interpolation of time-centered estimates of primitive variables from edge centers to edge centroids and reconstruction of the flux (we do not interpolate the fluxes directly as in [8]).

5. At cut cells, calculation of a conservative (respectively, non-conservative) update based on time-centered, edge-centered (respectively, time-centered, edge-centroid) estimates of the flux. At cells with no shoreline, the two estimates agree, since the centroid is the edge-center.

6. Update of the solution in time, by calculating the divergence of the flux. At cut cells, use a volume weighted hybrid of the conservative and non-conservative flux estimate.

7. Use of the modified Euler (Heun) predictor-corrector formalism to estimate and add the source terms at the same time-centering as the flux.

8. Redistribution of mass locally to conserve total mass lost through using the hybrid update.

Hybridizing the conservative and non-conservative estimates of the flux permits a explicit time-step that doesn't decrease because of small cut cells: The weight given to the conservative update decreases as the area of the cell tends to zero. We mitigate the penalty this imposes on conservation by tracking the deficit/surplus and adding that mass (respectively, momentum, salt mass) to neighboring cells. The bottom source is hybridized the same way the fluxes are on irregular cells to preserve free-stream.

6 Grid Generation

The discretization described in this document requires the delineation of the 2D grid and the calculation of 2D and 3D moments (centroids, volumes and face fractions) included in the integral 2D fluxes. The design must also be efficient for 2D problems with both fixed and evolving boundaries, and must be sufficiently accurate for moving boundary applications. The present design satisfies these requirements with a geometry engine

based on the divergence theorem that can calculate moments of arbitrary order and degree of accuracy. Note that although we accurately calculate the moments associated with the embedded boundary, we do not actually model the boundary itself.

Our geometry generation is based on an implicit function. This function is a smooth function with continuous second derivatives that is zero where the boundary is encountered. For 3D bathymetry, such a function is given by:

$$\phi(x, y, z) = z - b(x, y) \tag{6.19}$$

which measures the difference between a point z and the elevation of the bed $b(x, y)$ at the same horizontal (x, y) location. This function clearly evaluates to zero at the bed itself.

To delineate a 2D boundary, we use the function:

$$\phi(x, y) = z_{ref} - b(x, y), \tag{6.20}$$

where z_{ref} is the elevation of a reference plane whose intersection with the 3D terrain gives the 2D domain.

The algorithm supposes the implicit function is available everywhere in space with smooth derivatives and continuous second derivatives. In practical applications only samples are available. We use data in the form of digital elevation models (DEMs). To work with DEMs we use bicubic interpolation to provide values and derivatives for the geometry calculation. We find that when high resolution data are used a small degree of smoothing (a Gaussian kernel spanning 1-2 grid cells in each direction) reduces the need to compute geometry at high resolution. For instance, on parts of the San Francisco Bay we use 200-800m cells for the computational grid, with geometry calculated from 30m DEMs smoothed on a radius of 60m. In this location, the computational grid is greatly oversampled by the DEM. Without the smoothing, the geometry would have to be calcultated at 15m and coarsened, which represents a tedious computational burden just to obtain a 500m mesh.

7 Particle Tracking

A particle model (Figure 2) will be used to visualize and calculate trajectories of neutrally buoyant particles travelling on streamlines:

$$\frac{ds_p}{dt} = u(\mathbf{x}, t), \tag{7.21}$$

where s_p is the position of the pth particle and $u(x, t)$ is the local velocity. The particle model is intended to be extensible to biology applications that may include drag, behavior and subgrid mixing (stochastic diffusion).

The particle tracking algorithms we considered draw from these approaches:

1. Direct solutions of (7.21) based on an ordinary differential equation method such as Euler or Runge Kutta.
2. Locally exact methods such as [9] or [11] which discretize the flow field in time and space and then track particle movement exactly as they enter and exit cell faces.

Under the ODE approach, the flow field may be interpolated along particle paths to any accuracy supported by the output of the hydrodynamic model. Care must be taken to monitor interaction with the embedded boundary and movement between multi-block patches assigned to different processors in parallel computations.

The locally exact approach assumes something about the flow over a cell (usually that it is linear in space or time) and then integrates from face to face by calculating exit times and locations. The method is more difficult to apply accurately to stochastic diffusion, but makes it easy to keep track of the topology and processor layout near boundaries. Face-to-face methods are stable.

Our method is a low order variant of the locally exact method. We assume piecewise constant cell values in space and time. For each particle, (7.21) is integrated exactly (equivalent to forward Euler) until an outer time step is exhausted or the particle reaches a new cell edge. If a particle crosses an edge, the velocity in the new cell is re-evaluated linearly in time.

Our first implementation focuses on off-line velocity computations where regridding does not occur adaptively. We are currently working on an in-line version with the possibility of adapting the grid to particle density and eventually moving shorelines. The linear time interpolation we perform when particles cross cell boundaries is complicated by mesh changes and load rebalancing for parallel computation. We intend to investigate the cost of this interpolation versus the cost of implicit methods using only velocities at the new time step. Adaptation makes piecewise-constant approximations in cells reasonable; without adaptation our method can be extended to higher order in space and time as in [11].

Particles are reflected when they hit embedded boundaries. Some biological applications require that particles stay in the fluid even in underresolved cases. When a particle moves in an irregular cell that intersects the embedded boundary, we construct a linear approximation of the EB using the 2D cell area and normal, check whether the particle has crossed the boundary

and reflect the particle back into the domain for the remaining part of the time step.

8 Summary and Future Work

Out of the goals presented earlier we have completed an initial implementaion of three major tasks in the context of parallel computation and adaptive mesh refinement: grid generation, two dimensional modeling, and an offline particle tracking model.

The accomplishment of several additional tasks will enable the enlargement of the domain to the entire Bay-Delta. First, our discretization depends on a notion of a reference depth, which we denoted H. However in the eastern Sacramento-San Joaquin Delta, the channel bottom is at a higher elevation then the water surface in the Bay, which does not permit the definition of a single reference surface; a reasonable nominal surface in the Bay would be near or below the channel bottom upstream in the Delta. Hence, the algorithm must be modified to allow a slowly varying H. Additionally, the eastern Delta contains mostly one dimensional hydrodynamics. Adaptive mesh refinement in principle could allow an efficient representation of channel networks using long, narrow patches of refinement. However, the current time-stepping algorithm would force unacceptably small time steps due to the small transverse distance in a channel. Nonetheless, the absence of flow in the transverse direction implies that this limitation can be overcome without loss of stability or accuracy for many applications. Finally, optimization for speed will make the model available for full delta studies over long simulation times.

Acknowledgements: Work at LBL was by supported by the U.S. Department of Energy Office of Advanced Scientific Computing Research, contract DE-AC02-05CH11231, and by the California Department of Water Resources Delta Modeling Section, contract 4600003803

References

[1] M. Aftosmis, M. Berger and J. Melton, *Robust and efficient Cartesian mesh generation for component-base geometry* AIAA Journal, Vol 36, 1998, pages 952-960 .

[2] M. J. Berger and P. Colella, *Local Adaptive Mesh Refinement for Shock Hydrodynamics*, J. Comput. Phys.,Vol. 82, 1989, pages 64-84.

[3] M. Berger and J. Oliger, *Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations*, J. Comput. Phys.,Vol. 53, 1984, pages 484-512.

[4] L. Brice, C. Y. Niño and M.C. Escauriaza, *Finite Volume Modeling of Variable Density Shallow-water Flow Equations for a Well-mixed Estuary: Application to the Riò Maipo Estuary in Central Chile*, J. Hydraulic Engineering, ASCE, Vol 43, pages 339-350.

[5] V. Casulli and P. Zanolli, *High Resolution Methods for Multidimensional Advection-Diffusion Problems in Free-Surface Hydrodynamics* Ocean Modelling, Vol 10, 2005, pages 137-151.

[6] P. Colella, *Volume-of-fluid Methods for Partial Differential Equations*, Godonov Methods: Theory and Applications, Ed. E. F. Toro, Kluwer Academic, 2001, pages 161-177.

[7] P. Colella, *Multidimensional Upwind Method for Hyperbolic Conservation Laws* , J. Comput. Phys.,Vol. 87, 1990, pages 171-200.

[8] P. Colella, D. T. Graves, B. Keen and D. Modiano, *A Cartesian Grid Embedded Boundary Method for Hyperbolic Conservation Laws*, J. Comput. Phys.,Vol. 211, 2006, pages 347-366.

[9] P. Kipfer, F. Reck and G. Greiner, *Local Exact Particle Tracing on Unstructured Grids*, Computer Graphics Forum, Vol. 22, 2003, pages 133-142.

[10] E. Gross, M. MacWilliams and W. Kimmerer, *Three-dimensional Modeling of Tidal Hydrodynamics in the San Francisco Estuary* San Francisco Estuary and Watershed Science, Vol 7, 2009.

[11] DW Pollock, *Semi-analytical Computation of Path Lines for Finite-Difference Models*, Ground Water, Vol 26, 1988, pages 743-750.

[12] C. Rendleman, V. Beckner, M. Lijewski, W. Crutchfield and J. B. Bell, *Parallelization of Structured, Hierarchical Adaptive Mesh Refinement Algorithms*, Computing and Visualization in Science, Vol. 3, 2000, pages 147-157.

[13] E. Toro, *Shock-Capturing Methods for Free-Surface Shallow Flows*, John Wiley and Sons,2006.

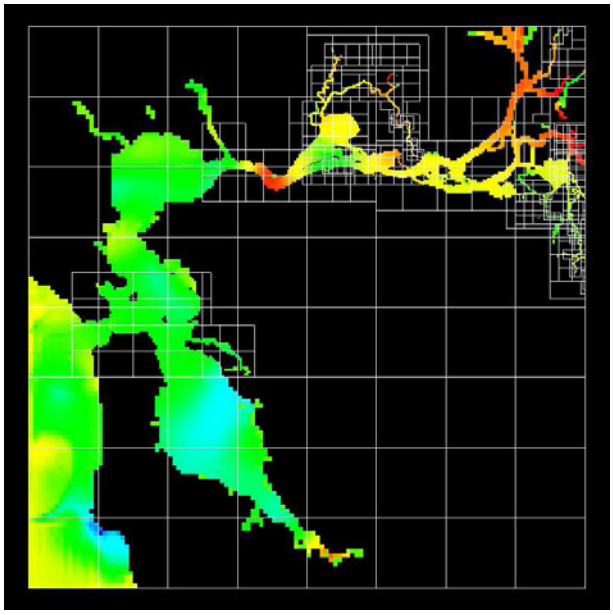

Figure 1: Adaptive mesh refinement on San Francisco Bay and Western Delta. The boxes represent multi-block elements. The largest boxes have a resolution of 750m. Three levels of refinement by a factor of two appear in this mesh, with the finest level covering channelized areas in the Delta on the east side of the map.

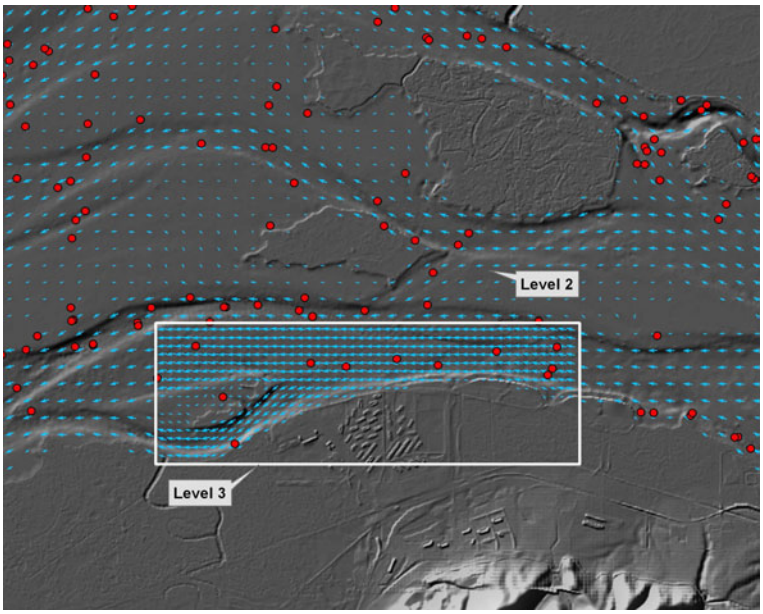

Figure 2: Particle tracking (red) near the Suisun marsh with velocity vectors illustrating two levels of AMR. The adaptation in this case resolves some complex local bathymetry.

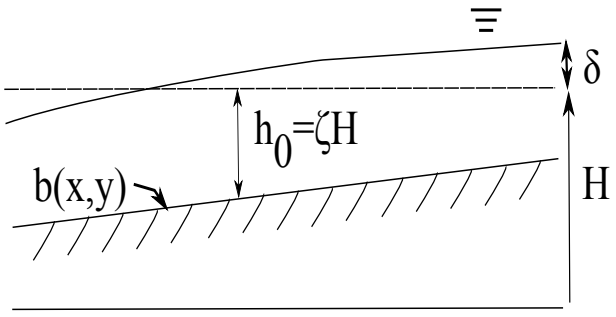

Figure 3: Side view of water column showing nominal and perturbed water column height.

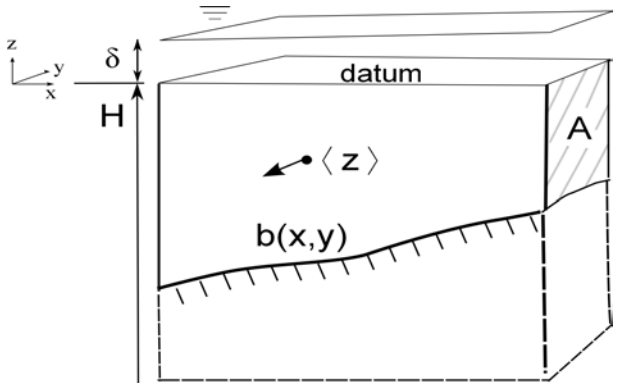

Figure 4: A computational cell illustrating aspects of the 3D geometry. Our 2D discretization requires the area of the face A, the z-component, denoted $< z >$, of the centroid on a vertical face, and the gradient of $b(x, y)$. This illustration also shows perturbation region, defined by δ, where we approximate the bathymetry with vertical walls. For cells that contain shoreline, we also require the centroid, normal, areas, and apertures on the top face (not shown.)