Heuristic and Eulerian Interface Capturing Approaches for Shallow Water Type Flow and Application to Granular Flows

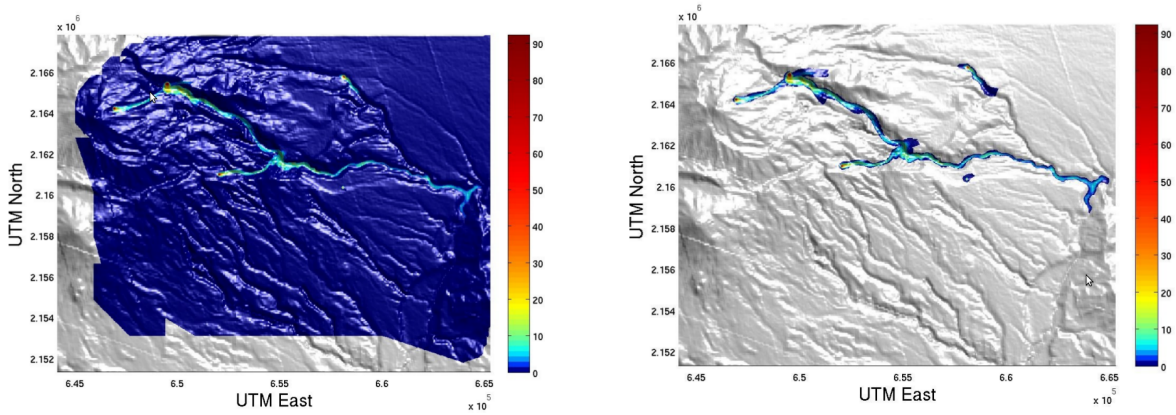
Hossein Aghakhani, Keith Dalbey, Abani Patra, David Salac

Abstract

Determining the wet-dry boundary and avoiding the related spurious thin-layer problem has historically been a challenge when solving the depth-averaged shallow-water (SW) equations (or similar granular flow's equations), and has been the focus of much research effort. In this paper, we introduce the use of level set and phase field based methods to solve this issue and related problems. We also propose new heuristic methods to address this problem. We implemented all of these methods in TITAN2D, which is a parallel adaptive mesh refinement toolkit designed for numerical simulation of granular flows. Results of the methods for flow over a simple inclined plane and Colima volcano are used to illustrate the methods. For the inclined plane we verified the results with experimental data and for Colima volcano they have been compared with field data. We successfully captured the interface of the flow and solved the accuracy and stability problems related to the thin layer problem in SW numerical solution. The comparison of results shows that although all of the methods can be used to address this problem, each of them has its own advantages/disadvantages and methods have to be chosen carefully for each problem.
Keywords: Shallow water flow, Thin layer, Wetting/Drying, Phase field, Level Set

1 Introduction

Shallow water (SW) flows include a wide range of fluid flows. In SW flows the fluid depth is much smaller ($\mathcal{O}(10^{-1})$) than the characteristic length of the fluid body. In deriving the corresponding governing equations, the shallowness of flow allows us to approximate the effect of the variation of state variables in the direction perpendicular to the basal surface and replace it with an integrated average [41], which reduces a three dimension flow problem into a two-dimensional one. This approximation holds for many circumstances in geophysical flows and the same conservation equations with some minor variations can be used to study varying physical situations. Eglit and Sveshnikova [19] modified the depth-averaged Saint Venant equations to simulate granular snow avalanches and almost a decade later Savage and Hutter [41] popularized these in the modeling of many geophysical mass flows related to landslides, avalanches and debris flows. Since this type of flow has free moving boundaries one of the basic difficulties of the numerical solution is identifying the location of flow interface. Furthermore, the governing equations are valid only in the wet areas so we need a strategy to discriminate between wet and dry areas in the numerical simulation. In the SW context this is usually called the *wetting and drying (WD)* problem. In our previous work, see Ref. [35], we showed how the speed of waves has to be modified for flow near the vacuum region based on Toro [47] to mitigate stability concerns. However, this still leads to the formation of a non-physical thin layer in the numerical solution (see figure 1 for an illustration). This non-physical thin layer could extend large distances from the realistic main body of the flow, which can cause inaccurate construction of the boundary, loss of conservation or severe numerical instabilities in the numerical solution. Beside the numerical issues, determining probable flow extents through numerical simulation is critical for many SW problems, especially for geophysical flow. For example, in preparing a hazard map for a volcano or a flood it is crucial to know the location of the front of the flow to answer basic questions such as – Does the flow reach a specific location? What is the distance of high risk locations from civil infrastructure? The answer of all the above questions is not possible without good information about the interface of the flow along its flow path. A demonstration of possible issues in



(a) Without thin layer control

(b) Only flow with depth greater than 0.5 m is displayed.

Figure 1: Thin layer problem in maximum over time flow depth simulation of the 1955 debris flow at Atenquique, Mexico [16].

shown in Figure 1. In this figure the numerical simulation of block and ash flows in Atenquique, which is a village near the Colima volcano in Mexico, using a SW like model based on a granular flow assumption is shown. The left figure shows the numerical solution of flow height without using any control for the thin layer problem and the right figure shows the same result using a naive control – plotting the regions with flow height $h > 0.5m$. If no heuristic control on the numerical solution is used the thin layer causes instability and inaccuracy in the obtained results. To summarize, the following major difficulties arise in numerical solutions of SW flows related to the WD or thin layer problem:

1. Ambiguous and subjective computation of flow spreads in SW flows.
2. Unphysically fast estimate of the flow speed. As the solution of SW equations are momentum, $\{hV_x, hV_y\}$, to find velocities, $\{V_x, V_y\}$, used during the solution process the momentum must be divided by flow depth, h, which causes a numerical error if h is small and results in overly large velocities.
3. Unphysical thin layer (orders of magnitude thinner than a grain of sand). This results from unphysical speeds and the numerical wicking away of material.
4. Loss of numerical stability. Due to the above reasons wave-speeds can become infinite at the flow boundary which means that flow equations lose their hyperbolicity at the vacuum state interface.

In addition to scaling issues, correct identifying of interface regions will allow us to construct models for the other interesting physics (e.g. entrainment) that happens at the interface.

1.1 Background and Review

A review of the literature shows numerous attempts have been made to overcome these problems [45, 5, 3, 8, 46, 28, 18, 11]. Here we briefly review the problem and recent approaches and refer the interested reader to available references for more information. Generally, all of the approaches can be categorized into two basic groups which we call heuristic methods and interface tracking/capturing methods. The basic idea of heuristic methods is to reconstruct the flow interface based upon some simple heuristic or some algebraic relations. In this approach, the reconstruction could take into account the physics of the problem, but this is not required. For example, to mitigate the thin layer problem a threshold could be defined to control the thin layer, the entire domain could be filled with a shallow level of fluid, the flow could be reconstructed, or the flux could be adjusted. It is possible to use any combination of these heuristic fixes. Most of attempts

that are made to address the thin layer problem are categorized under this sort of method [3, 8, 11, 28]. In interface tracking/capturing approaches an auxiliary set of equations is coupled to the original problem. This auxiliary set of equations is used to follow the flow interface over time. Unlike heuristic method in interface tracking/capturing methods one can find the interface of a moving boundary flow in a rigorous way. The particular choice of the auxiliary equations leads to either interface tracking, or Lagrangian, methods or interface capturing, or Eulerian, methods. In Lagrangian methods the front is replaced with a set of interfacial points which explicitly define the location of the interface. During each time step these points move due to the numerically computed velocity field. Marker And Cell (MAC) [25], Simplified Marker And Cell (SMAC) [14] and the Surface Marker [51] methods are some examples of Lagrangian interface tracking techniques. The accuracy of the method is highly dependent on the number of particles that form the boundary. High computational cost, a tendency to form numerical instabilities and the inability to track complex topological changes are the most important drawbacks of Lagrangian techniques. For more information about this group of methods please see [22, 49, 34, 2, 26]. In the current paper we focus more on interface capturing or Eulerian methods. In such method the interface is implicitly defined by an additional scalar field defined in the entire computational domain. This additional field is coupled to the other governing equations and evolves due to the underlying fluid flow field. The level set, phase field and Volume of Fluid (VOF) methods are very well-known methods of implicit/Eulerian methods. Hirt and Nichols [26] were the first to propose a VOF method. In this method, the scalar variable is the fraction/volume of a particular fluid in each cell. To construct the interface based on this method an interface surface reconstruction technique is performed at each time step. Youngs [53] achieved a significant improvement in the VOF by adding a piecewise linear interface calculation (PLIC) representation of the fluid boundary. Youngs' method has been shown to be robust and efficient, but it is only first-order accurate. More advanced VOF methods are also available in [21] and Gopala and van Wachem[23], but these method require more complicated logic to reconstruct the interface. The phase field method is an Eulerian interface capturing scheme where the interface is implicitly related to an order parameter which shows the phase variation on the domain [2]. In this method the interface is a diffusive region between phases. The Cahn-Hilliard and Allen-Cahn formulations are two principal formulation for this method [10, 9, 52]. The difference between these two forms is that the first one is mass conservative while the second is not. Mathematically, the conservative form has a forth order derivative while the other one has a second order derivative. This difference in the highest order of derivative will make the later form much easier to implement. To our best knowledge no published work has used the phase field method to capture an interface in the shallow water equations. The Level Set method is another Eulerian interface capturing technique. It was introduced by Osher and Sethian in 1988 [34]. In this method, the scalar variable is typically a signed distance function which indicates the distance of a grid point to the interface. The only work that used the level set method to mitigate the wetting-drying problem is Quecedo and Pastor [39]. They developed a Taylor-Galerkin approach to solve the SW equations. They presented two different options for handling the wetting-drying areas. The first approach used was described as a "simple yet efficient" method which falls into heuristic method type approaches. Although they claimed that they used level set as the second approach, they did not report any results using the method. They finally concluded that the level set is expensive and unnecessary for their problem.

1.2 Contribution

In this paper, we describe an approach to the thin layer issue (problems 1-4 discussed in the introduction section), in two parts. The first part is to describe the interface of the flow and the next is to make suitable modifications to meshes, state variables and fluxes based on the result of the first part to resolve the aforementioned issues. To capture the interface we study three different methods. The first one is a new multifaceted Heuristic approach that uses a very small dimensionless threshold, which we call it GEOFLOW_TINY, to distinguish between wet and dry areas. The other methods of interface capturing that we implement are the phase field and level set approaches. To the best of our knowledge the two latter methods are for the first time being applied to geophysical flow. As there are different types of level set and phase field methods available in the literature we initially use the standard level set method and the

Allen-Cahn description of the phase field, and modified them as necessary for our problem. The details of implementation of all three methods is explained in related next parts. After finding the interface we make suitable modifications to address the other issues. In the heuristic approach, after finding the wet and dry cells by using GEOFLOW_TINY, the cells are divided into three groups: wet, dry and partially wet cells. Then we adjust the fluxes for the partially wet cells and then finally update the state variables for wet cells. In the level set and phase field approaches we used the result of these two method for mesh refinement, and by selecting the cells that are close to interface we are able to control the thin layer problem. Details of each approach and how we select the cells for refinement discussed in the relevant section. We verified the numerical results by comparing to experimental results of a granular SW flow over an inclined plane ending in a horizontal surface as well as the field data for the Colima volcano. Results of these tests indicate that the heuristic approach is less expensive than the phase field and level set methods, but the interface using the heuristic method moves faster than the experimentally obtain interface. Additionally, our analysis shows that finding a good threshold that is appropriate for all problems is not easy. On other hand, the interface capturing methods have stronger mathematical and physical structure, and can be coupled with the conservation equations but are computationally more expensive and harder to implement. We used several techniques to decrease the computational cost. For example, in the phase field we used the Allen-Cahn formulation which is easier to implement and computationally less expensive compared to the Cahn-Hilliard formulation, but it is not mass conservative. To preserve the mass we have included a Lagrange multiplier [31, 52] to satisfy this constraint. To decrease the computational cost even more we used operator splitting for time integration. The obtained results show that all of the methods can address the thin layer problem and other related issues, however the phase field results have demonstrated better consistence with the experimental and field data. Finally, although we have implemented these methods for granular type SW flows the methods themselves are applicable for other type of SW flows.

The structure of the rest of this paper is as follows. In the next section the shallow-water equations for geophysical flows and the common features of the solver that we used for all of the three methods are introduced. In section 4 the different methods that were employed to mitigate the wetting-drying problem are explained. Discussion and conclusions complete the presentation.

2 Governing Equations

The Savage-Hutter equation for geophysical flows was first introduced in the late 1980's. The original model has subsequently been improved upon by Savage-Hutter themselves as well as others [29, 27, 24, 17, 38, 42]. In our earlier work [37, 35, 36], we developed the TITAN2D depth-averaged geophysical flow simulator. TITAN2D is a parallel computational tool with dynamic re-partitioning, high order, slope-limiting, upwinding, two-dimensional Godunov solver (without splitting), with adaptive mesh refinement and Geographic Information System (GIS) integration which lets it use Digital Elevation Models (DEMs) of real terrain. While our new multi-faceted thin-layer mitigation strategies were developed in the context of TITAN2D's capabilities, much of this approach should be appropriate for use in depth-averaged flow solvers with different numerical implementations. The depth-averaged equations that TITAN2D solves are:

$$\begin{array}{lcllcl} \dfrac{\partial h}{\partial t} & + & \dfrac{\partial (V_x \cdot h)}{\partial x} & + & \dfrac{\partial (V_y \cdot h)}{\partial y} & = S_h, \\ \dfrac{\partial h V_x}{\partial t} & + & \dfrac{\partial \left(V_x \cdot h V_x + 0.5 k_{ap} g_z h^2\right)}{\partial x} & + & \dfrac{\partial (V_y \cdot h V_x)}{\partial y} & = S_x, \\ \dfrac{\partial h V_y}{\partial t} & + & \dfrac{\partial (V_x \cdot h V_y)}{\partial x} & + & \dfrac{\partial \left(V_y \cdot h V_y + 0.5 k_{ap} g_z h^2\right)}{\partial y} & = S_y. \end{array} \tag{1}$$

In these equations:

- The coordinate system (see Figure 2) is aligned such that x and y are directions tangential to the surface of the 3D terrain and z is normal to the surface.

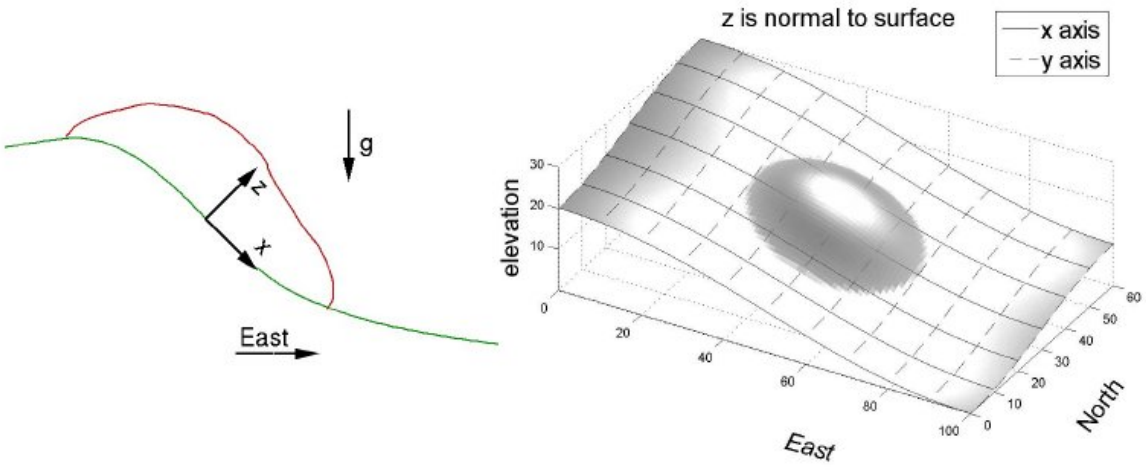


Figure 2: In the local coordinate system the z direction is normal to the surface; the x and y directions are tangential to the surface.

- The effect of terrain elevation is represented by gravitational source terms.
- h is the flow depth in the z direction.
- hV_x and hV_y are the components of "momentum" in x and y directions, respectively.
- k_{ap} is a highly nonlinear term that shows the effect of the earth pressure coefficient, which has active (diverging $\mp = -$), passive (converging $\mp = +$), and neutral (neither diverging, nor converging, $k_{ap} = 1$) conditions:

$$k_{ap} = 2\frac{1 \mp \sqrt{1 - \cos^2(\phi_{int})\left(1 + \tan^2(\phi_{bed})\right)}}{\cos^2(\phi_{int})} - 1. \tag{2}$$

Note that in the momentum equations $k_{ap} g_z \frac{h}{2} h$ is the contribution of hydrostatic pressure to the momentum fluxes. S_h is a source of mass, i.e. material that either effuses or erodes out of the ground. S_x is the sum of a gravitational driving force, the friction that resists motion of the material relative to the bed, and the friction that resists the internal shearing motion of the material:

$$\begin{aligned} S_x &= g_x h - \frac{V_x}{\sqrt{V_x^2 + V_y^2}} \max\left(g_z + \frac{V_x^2}{r_x}, 0\right) h \tan(\phi_{bed}) - \operatorname{sgn}\left(\frac{\partial V_x}{\partial y}\right) h k_{ap} \frac{\partial (g_z h)}{\partial y} \sin(\phi_{int}), \\ S_y &= g_y h - \frac{V_y}{\sqrt{V_x^2 + V_y^2}} \max\left(g_z + \frac{V_y^2}{r_y}, 0\right) h \tan(\phi_{bed}) - \operatorname{sgn}\left(\frac{\partial V_y}{\partial x}\right) h k_{ap} \frac{\partial (g_z h)}{\partial x} \sin(\phi_{int}). \end{aligned} \tag{3}$$

TITAN2D solves the above system of equations using a finite volume Godunov method:

$$U_i^{n+1} = U_i^n - \frac{\triangle t}{\triangle x}\{F_{i+\frac{1}{2}}^n - F_{i-\frac{1}{2}}^n\} - \frac{\triangle t}{\triangle y}\{G_{i+\frac{1}{2}}^n - G_{i-\frac{1}{2}}^n\}. \tag{4}$$

In the above equation, G and F are the flux terms at the inter-cell boundaries which are computed by the Harten-Lax-Van Leer (HLL) [48] Riemann solver that is explained in the next section.

3 SW Solution and Adaptive Strategy for WD Interface

3.1 Using Appropriate Riemann Fluxes

Following the recommendation of Toro [47], TITAN2D uses the HLL Riemann average of fluxes. Let U be a state variable, F be a flux of that state variable, s be a maximum wave speed, and R and L subscripts which denote "right" and "left" values respectively. The HLL Riemann flux is then

$$F_{HLL} = \begin{cases} F_L & s_L \geq 0 \\ F_R & s_R \leq 0 \\ \dfrac{s_R F_L - s_L F_R + s_L s_R (U_R - U_L)}{s_R - s_L} & \text{otherwise} \end{cases} \tag{5}$$

For the Savage-Hutter system of equations the characteristic speeds are: $s_{1,3} = v \pm a$ and $s_2 = v$ where v is the velocity of fluid in the corresponding direction and $a = \sqrt{k_{ap} g h}$. The above HLL solver is not enough for solving this system of equations. Fraccarollo and Toro [20] note that at the wet-dry boundary the Riemann solution consists of a single rarefaction wave whose speed (when averaged over the cell length) is bounded above by the cell average flow speed plus **twice** the "speed of sound," $s = v + 2a$, where $a = \sqrt{k_{ap} g h}$ for shallow water type granular flows. They also note that "an overestimate of the true wave speeds results in enhanced stability" while an "underestimate of the true wave speeds could be fatal" to stability. They became the first to solve this problem by constructing an approximate Riemann solver.

3.2 Adaptive Meshing

In TITAN2D each grid cell is a square or very nearly so. Rather than using a uniform mesh different sizes of cells are allowed, with each successive "generation" covering one fourth the area of its "parent" cell. Note that only one generation of irregularity is allowed between a cell and its neighbors. At the beginning of the simulation, *i.e.* before the first update, the boundaries of all piles are maximally refined. An initial mesh for a simulation at Colima Volcano, Mexico, is displayed in Figure 3. During normal adaptivity cells are selected

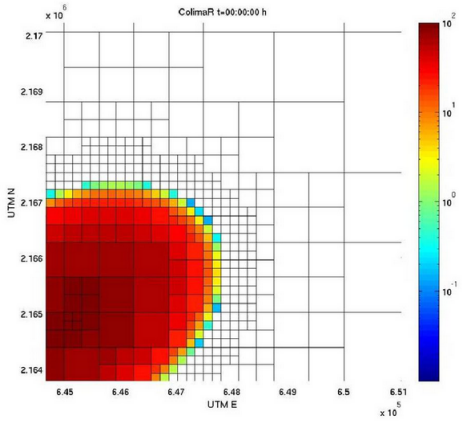


Figure 3: This figure shows four layers of the maximally refined elements near the interface that we call them buffer layer. Creating buffer layers is one of the key strategies that we do to control the thin layer problem in all of the methods that we studied in this work.

for refinement if they meet either of two requirements:

1. They have large, as compared to the average, inter-cellular fluxes.

2. They are at or within a few cells near the interface.

The former condition allows for the accurate capture of sharp changes in state variables. The latter results in banded regions of the flow separated into "rings" of maximally refined buffer cells. This limits artificially high transportation of material from the inside of the interface to outside of the interface and guarantees that material will only flow into maximally refined dry cells at the wet-dry front. However, some material that are outside of the recognized boundary may be left outside the outermost band of buffer cells. These cells are not updated until the interface passes them. The rings of buffer cells strategy decreases the numerical wicking problem and when combined with interface capturing scheme it is sufficient to decrease other thin layer problems to a level that prevents the loss of stability. Clearly this strategy depends on the way that we describe the interface. In the heuristic approach, after using a threshold to find the wet and dry areas the number of rings are a number of cells wide equal to or greater than the number of iterations between mesh adaptations (we adapt the mesh after every 5 to 10 time steps). In the level set approach we can use a narrow band method for adaption and selecting of these buffer rings, but we simply select the elements such that their distance to the interface are less than a normalized distance that is obtained from the maximum distance to the interface. Note that in the level set method distance of the cells to the interface is the level set function itself and we do not pay any additional computational cost for that information. In the phase field method we have a diffusive region between two phases that we can control with a capillary width (please refer to section 4.2). We are thus able to select those cells such that $|\phi| < 1$ where ϕ the phase field value. We set the capillary width equal to the number of time steps between successive adaptivity steps. Unrefinement takes place immediately after refinement; a group of four "sibling" cells will be selected to merge into their mutual "parent" cell if the sum of their inter-cellular fluxes is very low compared to the average flux. Cells that have either just been refined or are otherwise in any of the current bands of buffer cells are immune to unrefinement.

4 Interface Capturing Strategies

In the aforementioned parts the major difficulties of numerical analysis of the shallow water equations were discussed, and it was shown that most of them are directly or indirectly related to the WD problem. For more intuition, the result of a simulation for Atenquique debris is displayed in figure 1. The left picture is the obtained result from the solver and the right picture is the same result, but flow height is displayed only for flow height greater than 0.5 m. This picture shows for the pure solver the dominant part of the domain is filled with negligible flow depth and this makes the actual interface mix with the thin layer region. In this section three methods to solve the WD problem which mitigate the numerical difficulties of SWE are explained.

4.1 First: Heuristic methods

Usually, heuristic methods use one or a combination of the following four strategies [45]:

1. Filling the entire computational domain with a thin layer of fluid,
2. Using a depth scale to check whether a cell or a node is wet or dry (or possibly partially wet), and then making a decision to add or remove it from the computational domain,
3. Employing some extrapolating scheme from the wet cells into their neighbor cells to approximate the location of the interface. This method is usually called volume/free-surface relationship (VFR) in the literature,
4. Permitting fluid height to be negative, which means that it is below the topographical surface.

Table 4.1 compares these heuristic methods.

Strategy	Mass Conservation	Physics
Thin film	Adequate, but requires solution reconstruction	Produces a smooth and realistic wetting front
Cell removal	Dependent on numerical method for solving the equations	Excellent, performs better on advancing front than receding front
VFR	Conservative, with aid of some correction procedure	Very good in wide verity of problems
Allowable negative depth	Conservative, but performance depend on WD parameters	Same as mass conservation

Note that the heuristic method we used in this study can be categorized as a VFR method.

4.1.1 Selecting appropriate threshold

In this approach we use a very small threshold for flow height to segregate the wet, partially wet and dry cells. The critical contribution of this scaling is to allow other facets like flux adjustment and boundary reconstruction to identify where the flow is non-physically thin, *i.e.* where should they consider the boundary of the flow to be. Consistency is the key requirement of whatever strategy we choose to determine the scaling. That is, the chosen strategy must be able to generate the same "appropriate" value for depth scale at the beginning and at the end of a simulation. In a "typical" geological simulation, say of the collapse of a volcanic dome which would be modeled in TITAN2D as a"pile source", the initial body of mass can be quite deep, but at the end of the simulation the material will likely be spread over a large area. Hypothetically, if one were to scale by maximum flow depth at the beginning and end of the simulation, they would obtain vastly different values. More importantly, if the only source of mass was an effusion of material out of the ground, the maximum initial flow depth would be zero, even if the total volume during the course of the simulation was the same as the previously mentioned "pile source". On the other hand, scaling by the cube root of the total volume of the flow being simulated is entirely consistent and is the flow depth scaling factor used by TITAN2D. We do **not** claim that the cube root of volume is any more or less appropriate as a scaling factor than maximum initial flow depth in other shallow water contexts, for example storm surge simulation. Having chosen a consistent scaling factor we were then able to define associated non-dimensional depths for negligibly thin and merely thin flows. If one were to assume that a particular geophysical mass flow event involved a volume of 10^8[m^3], it would then be reasonable to state that flow depths of less than 5[cm] were both negligible and non-threatening. This roughly equates to non-dimensional negligible flow depth, which we call `GEOFLOW_TINY`, with the value

$$\texttt{GEOFLOW\_TINY} = 0.0001. \tag{6}$$

Using this value, and assuming the volume used in a laboratory scale test was 1[cm^3], the resultant negligible flow depth would be 0.0001[cm]. As can be observed from these two examples, the chosen value of `GEOFLOW_TINY` is physically appropriate across a very large range of volumes. We therefore use a theoretical contour at this depth as the boundary of the simulated flow. In the introduction section we stated that unless steps are taken to prevent it, the numerical wicking (Problem 2) will cause the flow to spread 1 cell every time-step even though the product of flow speed and time-step size is less than the cell length. In addition, the familiar continuum equations loose hyperbolicity (wave speeds tend to infinity) at the boundary between a continuous material and a vacuum. Having implicitly defined the flow boundary, we prevent calculation of state variables outside the flow, *i.e.* in regions with flow depth below `GEOFLOW_TINY`. This reduces both non-physical flow spreading and computational cost. Note that state variables in cells with flow depth less than `GEOFLOW_TINY` are **not** zeroed.

4.1.2 Interface Reconstruction

As noted above, the use of a standard Eulerian grid imposes discrete fixed increments to the flow extent which contributes to the wicking problem (Problem 2) at the boundary. Use of adaptive mesh refinement reduces

this error by reducing the amount of the increment. The Lagrangian approach does not have this limitation, which Tai et al. [44] took advantage of when they augmented their one dimensional NOC scheme with Lagrangian front tracking. However, implementing a hybrid Eulerian-Lagrangian scheme in two dimensions is significantly more complex. Therefore, as part of our multi-faceted approach we implemented a very simple and inexpensive interface reconstruction and predictive Lagrangian front tracking scheme. Knowledge of the interface allows us to generate a representative average for an individual cell edge over the time-step and for values of state variables which are then used to compute inter-cellular fluxes into/out-of partially wet cells. The interface reconstruction scheme is illustrated in figure 4. Specifically, each partially wet cell is assumed

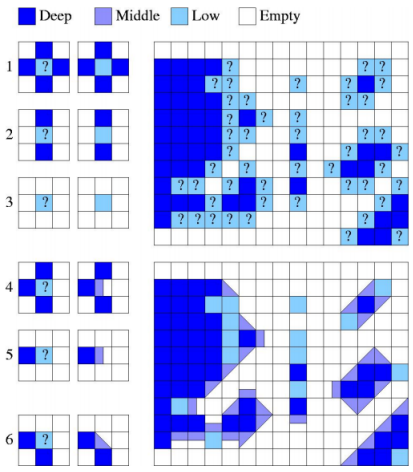


Figure 4: Interface Reconstruction in TITAN2D.

to be split by a straight line into a completely dry and a completely wet part. For the sake of simplicity, we restrict this line to one of four orientations: east-west, north-south, or parallel to either diagonal of the square cell. However, all placements/translations of the wet-dry line are allowed. At the beginning of each time-step, the orientation of the line for the entire time-step is set based solely on which of the cell's neighbors have flow depth greater/less than the `GEOFLOW_TINY` threshold[1]. Orientation of the wet-dry line, and which side of it is wet, is indicated by a single integer representing the geometrically determined "most wet node." The only nine possible values for the most wet node are any of the cell's corners, edge midpoints or its center. The most wet node is assumed to have a flow depth that is the maximum of the cell or any of its four neighbors. The placement of the line is such that the volume of material in the cell under a plane passing through the most wet node and the wet dry-line is the same as the unadjusted cell. Since we now know where the wet-dry line is at the beginning of the time step, we can compute which of the cell's edges are completely wet, completely dry, or partially wet, and the fraction of wetness for the partially wet edges. Given the orientation and beginning of time-step location of the wet-dry line and the cell's state variables, it is also fairly straight forward to predict the line's end of time-step location. This is done by convecting the wet-dry line at the shock speed, $s = v + a$ where the speed of sound has been adjusted to account for only part of the cell being wet, i.e. $a = \sqrt{k_{ap} g h \frac{A}{A_{wet}}}$. Here A is the whole cell's area and A_{wet} is the portion of the cell's area that is wet. Since the wet-dry boundary within each partially wet cell is assumed to be a straight line with, during the time-step, fixed orientation, convecting a single point, in this case the midpoint, on the line is equivalent to convecting the entire wet-dry line. A spatial and time average of the wetness factor for

[1] As stated in Subsection 3.2, the refinement strategy ensures that the flow front will always be maximally refined. This simplifies the coding since all cells at the front can be safely assumed to have only one neighbor on each side.

the edge over the time-step can therefore be easily computed as

$$W = \left(\frac{0 \cdot \Delta t_{dry} + \frac{1}{2}(w_{beg} + w_{end})\Delta t_{part} + 1 \cdot \Delta t_{wet}}{\Delta t} \right) \left(\frac{A}{A_{wet}} \right) \tag{7}$$

where $\Delta t = \Delta t_{dry} + \Delta t_{part} + \Delta t_{wet}$ is the entire time-step, Δt_{dry} is the portion of the time-step for which the edge is completely dry, Δt_{part} is the portion of the time-step for which the edge is partially wet and partially dry, Δt_{wet} is the portion of the time-step for which the edge is completely wet, w_{beg} is the edge's fraction of wetness at the beginning of the time-step, and w_{end} is the edge's fraction of wetness at the end of the time-step.

4.1.3 Adjusting Fluxes in Partially Wet Cells

The state variables used to compute the physical fluxes are the whole cell average values multiplied by the edge wetness factor. Note this results in the zeroing of this cell's physical fluxes for its sides that will be completely dry for the entire time-step and increasing them for sides that will be completely wet for the entire time-step. The numerical fluxes are then taken to be the HLL Riemann average of the "adjusted" physical fluxes from the cells on both sides the edge. The wetness factor adjustment of fluxes delays/decreases the transfer of material from partially wet cells to completely dry cells. This delay significantly reduces the amount of non-physical flow spreading, and requires negligible additional computation and memory usage. In terms of memory usage, this scheme only requires one additional integer indicating the most wet node, and two additional decimal numbers for the cell's fraction of wet area and the location of the wet-dry line's midpoint (implemented as a single number ranging from zero to one). The flux adjustment in partially wet cells mitigates the numerical wicking (Problem 2) at the boundary but not within the flow (that requires either a uniform grid or an adaptive strategy based on fluxes and the rings of buffer cells strategy).

4.2 Second: Phase Field Method

As noted earlier, another approach that is employed for capturing the interface of a SW type flow is the phase field method. In this method the state variables are augmented by a continuous order parameter. This order parameter, φ, implicitly represents the interface in the domain. To this aim, a new transfer equation must be solved which is coupled with the other state variables. Papers [13, 2, 6, 30] are good references about the history and evolution of the method. Phase diffusion methods and particularly the phase field method is based upon the notion that the interface between phases is a diffusive region rather than a sharp interface. The value of ϕ is constant within a bulk phase but changes smoothly between the phases. In this work ϕ is 1 for the fluid phase $\{\mathbf{x} : \phi(\mathbf{x}, t) = 1\}$, and it is -1 for void regions $\{\mathbf{x} : \phi(\mathbf{x}, t) = -1\}$, and is between -1 to 1 on the diffusion region $\{\mathbf{x} : -1 < \phi(\mathbf{x}, t) < 1\}$. We can implicitly assume the interface of the flow is where $\{\mathbf{x} : \phi(\mathbf{x}, t) = 0\}$. In this study we used the Allen-Cahn form of the phase field.

$$\frac{\partial \varphi}{\partial t} + \overrightarrow{V} \cdot \nabla \varphi = \gamma(\triangle \varphi - F'(\varphi)), \tag{8}$$

where

$$F(\varphi) = \frac{1}{4\eta^2}(\varphi^2 - 1)^2, \text{ so } \quad F'(\varphi) = \frac{\delta F}{\delta \varphi} = \frac{1}{\eta^2}\varphi(\varphi^2 - 1). \tag{9}$$

In above equation η is a constant that regulates the capillary width or diffusion width, γ denotes an elastic relaxation constant, and $F(\varphi)$ is a mixing free energy which is the well-known double-well potential function and represents the interactions of different volume fractions of individual species [7, 32]. This formulation is easier to solve, but is not mass conservative. To conserve the mass, a Lagrange multiplier in form of $(\varphi^2 - 1)\xi(t)$ is added as a source term to the right hand side of equation (8) [31, 52]. The reason that we select this form for the Lagrange multiplier is because we do not want to change φ on the bulk region as well as when on the interface i.e when $\varphi = \pm 1$. So the final for of equation $\xi(t)$ would be:

$$\frac{\partial \varphi}{\partial t} + \overrightarrow{V} \cdot \nabla \varphi = \gamma(\triangle \varphi - F'(\varphi) + (\varphi^2 - 1)\xi(t)), \tag{10}$$

and the constrain that has to be satisfied is:

$$\frac{D}{Dt}\int_\Omega \varphi h \;\; dx = 0, \tag{11}$$

which basically means the volume of pile must be constant during the simulation. Consequently $\xi(t)$ is derived as follow:

$$\frac{D}{Dt}\int_\Omega \varphi h \;\; dx = h\int_\Omega \frac{D}{Dt}\varphi \;\; dx + \underbrace{\varphi\int_\Omega \frac{D}{Dt}h \;\; dx}_{=0 \;\; \text{continuity}} = h\int_\Omega \gamma(\triangle\varphi - F'(\varphi) + \varphi(\varphi^2-1)\;\xi(t)) \;\; dx, \tag{12}$$

the first term in above relation will be neglected by applying the Gauss' theorem with considering the boundary conditions($\frac{\partial\varphi}{\partial n}|_\Gamma = 0$,):

$$\int_\Omega h\gamma\nabla.\nabla\varphi \; dx = \int_\Gamma h\gamma\nabla\varphi \; ds = 0, \tag{13}$$

so to always satisfy the constrain (11) we end up with:

$$\frac{1}{\eta^2}\int_\Omega \varphi(\varphi^2-1) \;\; dx = \xi(t)\int_\Omega (\varphi^2-1) \;\; dx \Rightarrow \xi(t) = \frac{\int_\Omega \varphi(\varphi^2-1) \;\; dx}{\eta^2\int_\Omega(\varphi^2-1) \;\; dx} \tag{14}$$

We set $\eta = n\;\delta x$, where δx is equal to the smallest cell length of all elements and n is the number of the buffer layers explained in section 3.2. For time integration of equation (10) we use operator splitting. The Euler explicit method is used for time integration of all terms except the Laplacian terms, which are updated implicitly. For a stable explicit time integrator, the size of the time step has to satisfy the CFL condition.

$$\frac{\varphi^{i+.5}-\varphi^i}{\triangle t} = \gamma(-F'(\varphi^i) + \varphi^i(1-(\varphi^i)^2)\;\xi(t^i)) \tag{15}$$

$$\frac{\varphi^{i+1}-\varphi^{i+.5}}{\triangle t} = \gamma\nabla.\nabla\varphi^{i+1} \tag{16}$$

We used GMRES solver of the PETSc [4] library to solve equation (16). Since the Krylov subspace solvers just needs the result of matrix-vector multiplication, we used matrix-free method to compute the Laplacian term to decrease the memory cost of implicit solver.

4.3 Third: Level Set Method

The last method that we use to capture the interface for a SW flow is the Level Set method. The Level Set method is another Eulerian interface capturing method, originally introduced by Osher and Sethian in 1988 [34]. The basis of the method is to capture the interface by means of solving a hyperbolic Hamilton-Jacobi PDE on the computational domain which follows the boundaries. The level set variable $\Psi(X,t)$ implicitly represents the interface. In this method, $\Psi(X,t)$ is a signed distance function which means it is zero on the interface, and its absolute value changes in the domain with respect to the distance to the zero level, and its sign is positive outside of the interface, and is negative inside of the interface. Given the initial condition $\Psi(X,t)_0$ the advection of the interface ($\Psi(X,t)=0$) only depends on the normal velocity F:

$$\frac{\partial\Psi}{\partial t} + F|\nabla\Psi| = 0, \tag{17}$$

substituting $F = \overrightarrow{V}\overrightarrow{n}$ which $\overrightarrow{n}$ is the normal vector, and is equal to $\frac{\nabla\Psi}{|\nabla\Psi|}$ leads to:

$$\frac{\partial\Psi}{\partial t} + \overrightarrow{V}\nabla\Psi = 0 \tag{18}$$

4.3.1 Reinitialization

The solution of equation (18) might no remain a signed distance function. To keep Ψ a signed distance function, we need a procedure that is called reinitialization. There are several techniques for this purpose [34], but what we implemented here is a composition of the method presented in Chopp's work [15] with some modifications for the points close to the interface, and a PDE based reinitialization for the further places[43]. PDE based reinitialization is easy for parallelization, though it has been known to induce mass loss. Chopp suggests a bicubic interpolation for the points near the interface [15], but here we use a bilinear interpolation which is not as accurate as what he suggested, but it is accurate enough to preserve the interface. For points far from the interface we do not need a very accurate Ψ, so we use a PDE based reinitialization which is easier to implement and has lower computational cost. In our approach, we first find Ψ for the points that are adjacent to the interface, then use these points to update the signed distance function for rest of the domain using a PDE based method. In this way, the interface ($\Psi = 0$) is preserved with an affordable computational cost and effort. Moreover, since just Ψ is required for the points that are located near the interface, there is no need to update it for whole of the domain, so in this work we select $\Psi_{thresh} < 10\ \delta x$, where δx is the smallest size of the elements, as the threshold for updating Ψ in the PDE-based reinitialization process. The signed distance function is updated for adjacent points to the interface in the following steps:

1. First we find the adjacent points to the interface, and store them in a list that is usually called the accepted point list in the literature [15].

2. The next step is to determine the interpolant for the level set function, $p(X)$, that returns $\Psi(X)$ for a point X.

3. The last step is to compute the distance of the accepted points based on the following conditions:

$$p(X_{int}) = 0, \tag{19a}$$

$$\nabla p(y) \times (X_0 - X_{int}) = 0, \tag{19b}$$

   where X_{int} is the nearest point on the interface, and X_0 is the point in the accepted list that we want to compute its distance to the interface. The first condition (19a) requires that X_{int} is on the interface, and the second condition states that the interface normal at the closest point must pass through X_0. In this part, we followed the notation of Chopp's paper [15].

In the first step, we just need to find those points where the sign of Ψ is different from their neighbors, and store them in a list that we call them accepted points list. For the second step, there are different ways to perform the interpolation. In many applications, like in this study, the initial interface has a specific geometrical shape such as a circle or ellipse, so we do not need to find this interpolant approximately, and its equation can be used directly as $p(X)$. For example, in this study the initial pile has an elliptic shape, so the initial interface follows the elliptic equation.

If we do not know the equation of the interface, which is usually the case, we need to approximate the interface using interpolation. Chopp suggested a bicubic approximation [15] to achieve a fully second order accurate result, but this method requires the solution of a 16 by 16 system of equations for cell containing the interface to determine the coefficients of the bicubic interpolation. The purpose of the interpolation based method is to ensure that the interface does not undergo spurious motion. Therefore, the use of bilinear interpolation should be sufficiency. The general form of p is:

$$p = a_0 + a_1\ x + a_2\ y + a_3\ xy. \tag{20}$$

A general way to find a_0 to a_3 for a grid with different element sizes is to solve a system of 4 by 4 equations (21) resulted from value of Ψ and the positions of the points.

$$\begin{bmatrix} 1 & x_1 & y_1 & x_1y_1 \\ 1 & x_2 & y_2 & x_2y_2 \\ 1 & x_3 & y_3 & x_3y_3 \\ 1 & x_4 & y_4 & x_4y_4 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} \Psi_1 \\ \Psi_2 \\ \Psi_3 \\ \Psi_4 \end{bmatrix} \tag{21}$$

In our case the elements near the interface are of uniform size, and the four cell centers can be considered as the vertices of a rectangle, so the coefficients of equation (20) can be computed as following:

$$a_0 = \frac{(\Psi_1 x_2 y_2 - \Psi_2 x_2 y_1 - \Psi_3 x_1 y_2 + \Psi_4 x_1 y_1)}{(x_2 - x_1)(y_2 - y_1)}, \tag{22a}$$

$$a_1 = \frac{(-\Psi_1 y_2 + \Psi_2 y_1 + \Psi_3 y_2 - \Psi_4 y_1)}{(x_2 - x_1)(y_2 - y_1)}, \tag{22b}$$

$$a_2 = \frac{(-\Psi_1 x_2 + \Psi_2 x_2 + \Psi_3 x_1 - \Psi_4 x_1)}{(x_2 - x_1)(y_2 - y_1)}, \tag{22c}$$

$$a_4 = \frac{(\Psi_1 - \Psi_2 - \Psi_3 + \Psi_4)}{(x_2 - x_1)(y_2 - y_1)}, \tag{22d}$$

$$\tag{22e}$$

where $\Psi_1 = (x_1, y_1), \Psi_2 = (x_1, y_2), \Psi_3 = (x_2, y_1)$ and $\Psi_4 = (x_2, y_2)$ as can be seen in the figure 5. After finding

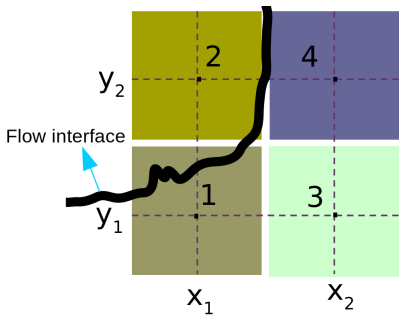


Figure 5: Configuration of the four elements that have been selected for bilinear interpolation. In this schematic example the interface passes through two hypothetical edges of (2-4) and (1-2) edges. This is just for demonstration purpose and has no effect on the equations.

the interpolation function p, we can compute the distance from a point to the interface using equations (19a) and (19b). The solution is calculated using a variant of Newton's method as presented in [15]:

$$\delta_1 = -p(X^k)\frac{\nabla p(X^k)}{\nabla p(X^k).\nabla p(X^k)}, \tag{23a}$$

$$X_{1+\frac{1}{2}} = X_k + \delta_1, \tag{23b}$$

$$\delta_2 = (X^0 - X^k) - \frac{(X^0 - X^k).\nabla p(X^k)}{\nabla p(X^k).\nabla p(X^k)}\nabla p(X^k), \tag{23c}$$

$$X_{k+1} = X_{1+\frac{1}{2}} + \delta_2. \tag{23d}$$

This is an iterative method that starts from X_0, which is the point that we want to find its distance to the interface. On the nearest point on the interface is determined, the distance can be computed, and sign of Ψ is the same as it was before reinitialization. For interpolation as shown in equation (20), we need four points. When we create the accepted list, instead of storing points we store a pair of points that are neighbors where their sign of Ψ differs. Then to find two other points, it is enough to select a direction perpendicular to the direction that connects the selected pair, and find the neighbors of the first two points in this direction.

Before proceeding to the PDE-based step, we would like to make some remarks about the aforementioned step. In parallel computation the domain is decomposed between multiple processors, with each processor holding a port responsible for a part of the domain. The information on neighboring processors is stored in "ghost"-cells, which are updated using inter-processor communication. In TITAN2D only the neighboring

ghost cells in the x and y directions are provided. Thus, a cell at location (i, j) does not have direct access to the information stored at location $(i+1, j+1)$. This causes issues in the rare instances that the interface is near the corner of a processor's domain, figure 6. To avoid excessive inter-processor communication the bilinear interpolation described before is modified to purely linear interpolation method requiring only three points:

$$p = a_0 + a_1 x + a_2 y. \tag{24}$$

When choosing the three points needed for interpolation we ensure that only the points in Cartesian directions are used. With these three points the interpolant is given by:

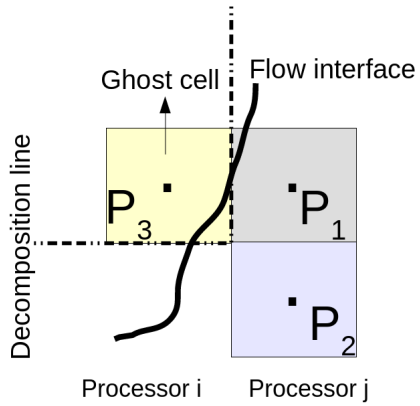


Figure 6: In this figure p_1 and p_2 are updated in processor j and p_3 is updated in processor i, but its information is available in processor j as ghost cell.

$$p_1 = \begin{bmatrix} x_1 \\ y_1 \\ \Psi_1 \end{bmatrix}, \quad p_2 = \begin{bmatrix} x_2 \\ y_2 \\ \Psi_2 \end{bmatrix}, \quad p_3 = \begin{bmatrix} x_3 \\ y_3 \\ \Psi_3 \end{bmatrix},$$

$$\overrightarrow{n} = \overrightarrow{p_2 p_1} \times \overrightarrow{p_3 p_1} = \begin{bmatrix} n_1 \\ n_2 \\ n_3 \end{bmatrix} = \begin{bmatrix} (y_2 - y_1)(\Psi_3 - \Psi_1) - (y_3 - y_1)(\Psi_2 - \Psi_1) \\ (x_3 - x_1)(\Psi_2 - \Psi_1) - (x_2 - x_1)(\Psi_3 - \Psi_1) \\ (x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1) \end{bmatrix}, \tag{25}$$

$$a_0 = \frac{n_1}{n_3} x_1 + \frac{n_2}{n_3} y_1 + \Psi_1 \qquad a_1 = \frac{-n_1}{n_3}, \quad a_2 = \frac{-n_2}{n_3} \tag{26}$$

Another point that we want would like make is that standardized structures, such as the set container of the C++ standard library, are very proper data structures for storing the adjacent points near the interface. The reason for this is that this container keeps its item in a sorted way, and does not allow the storage of duplicate items. This simplifies the coding effort as we do not need manually to check list every time that we want to save a new element.

After finding Ψ for the points adjacent to the interface, Ψ in other points can be updated with a PDE based reinitialization using an upwinding scheme. We use the following PDE for this aim:

$$\frac{\partial \Psi}{\partial \tau} + \mathrm{sgn}(\Psi_0)(|\nabla \Psi| - 1) = 0. \tag{27}$$

In equation (27), τ is a pseudo-time and the equation should be solved until it converges reasonably. This PDE adjusts Ψ such that $|\nabla \Psi| = 1$. We solve equation (27) by the method introduced in [1]:

$$\Psi_{ij}^{n+1} = \Psi_{ij}^{n} - \triangle t \left(max(F, 0) \bigtriangledown_{ij}^{+} + min(F, 0) \bigtriangledown_{ij}^{-} \right), \tag{28}$$

where

$$\nabla_{ij}^{+} = \big[\max(D^{-x}\Psi_{ij}^{n})^2 + \min(D^{+x}\Psi_{ij}^{n})^2 \quad \max(D^{-y}\Psi_{ij}^{n})^2 + \min(D^{+y}\Psi_{ij}^{n})^2\big]^{1/2} \tag{29}$$

$$\nabla_{ij}^{-} = \big[\min(D^{-x}\Psi_{ij}^{n})^2 + \max(D^{+x}\Psi_{ijk}^{n})^2 \quad \min(D^{-y}\Psi_{ij}^{n})^2 + \max(D^{+y}\Psi_{ij}^{n})^2\big]^{1/2} \tag{30}$$

In the above equations D^{+}, and D^{-} are the backward and forward differences, respectively, in the corresponding directions. We solve equation (27) until $\sum_{ij} |\Psi_{ij}^{n+1} - \Psi_{ij}^{n}| < .5(\delta x)^3$ over all points being updated.

5 Results

We use two cases to illustrate the interface capturing schemes. The first case is flow down an inclined plane while the second is past eruptions at the Colima Volcano. The first case provides detailed and accurate experimental data. The second is more characteristic of typical field observations available to calibrate and validate TITAN flows. Since the computational cost of the methods are not same, we have also compared these methods from a computational point of view by the required simulation time and the number of elements generate in the adaptive meshing process during the simulation.

5.1 Inclined plane

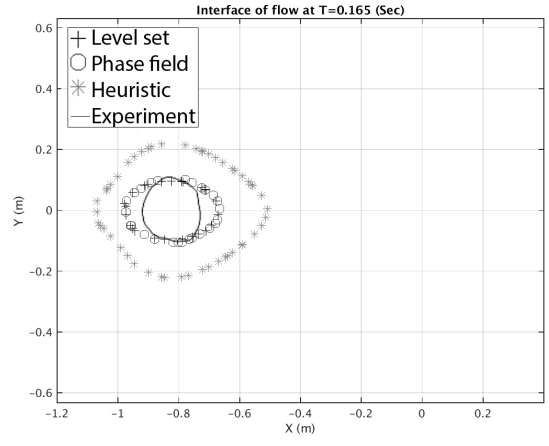
In this section the results of the inclined plane are presented. The input parameters and initial conditions of the flow could be seen in table 1.

Pile shape	Cylindrical
Maximum pile height	.061 m
Major extent of the pile	.0525 m
Minor extent of the pile	.0525 m
Bed friction angle	32.47^o
Internal friction angle	37.3^o
Angle of incline	38.5^o
Physical time simulated	90 sec

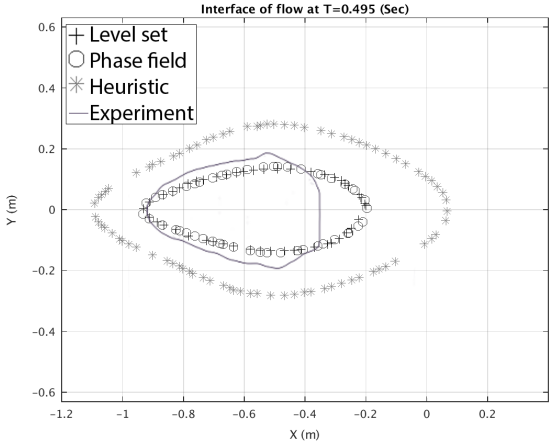
(a) Experimental setup

(b) Numerical simulation

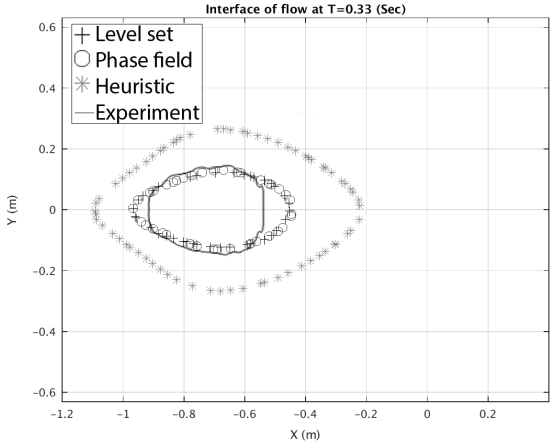
Figure 7: Initial configuration of the pile on the incline.



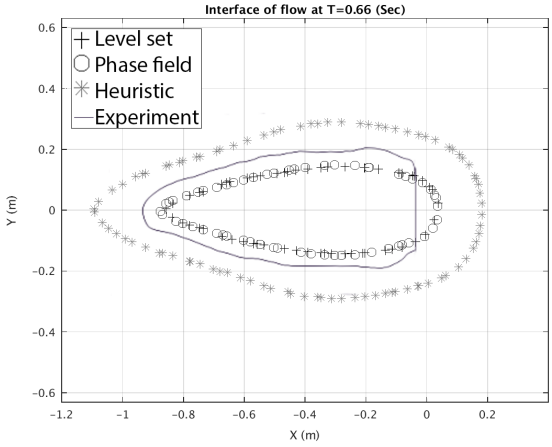
(a) T=0.165 sec



(c) T=0.495 sec



(b) T=0.33 sec



(d) T=0.66 sec

Figure 8: Pile height contour and interface location at different time steps.

The experimental results were reported in [50], and we have used them to verify our different numerical results with them in our previous works. Figure 7 shows the initial configuration of the pile in both the experimental setup and the numerical study. As can be seen in the experimental setup, there are markers in the inclined plane that are used for measurements. Webb [50] used a digital camera and image processing techniques to compute the extents of the flow in x and y directions as well as its area and perimeter, and then compared the obtained results with the numerical results in figure 9. The raw data from observation was re-analyzed in the course of this investigation and the actual flow depth data was updated to account for a small error in the original calculations reported in [50]. We also plotted the interface of the flow for the three interface capturing schemes and the experimental results in figure 8. The plotted interfaces in figure 8 are for $\phi = 0$ and $\Psi = 0$ for the Phase Field and the Level Set methods respectively and $h = \sqrt[3]{Vol} \times GEOFLOW\_TINY$ for the heuristic method where Vol is the volume of flow and $GEOFLOW\_TINY = .0001$, as was described in section 4.1.1.

Figure 8 shows that the results of the Eulerian interface capturing schemes are very similar for the inclined plane, and are closer to the experiment than the heuristic method. In all four plotted time steps, the heuristic method appears to move ahead of the true interface. It is possible to obtain better agreement by tuning the value of $GEOFLOW\_TINY$ but in that case we will end up with a $GEOFLOW\_TINY$ that depends on

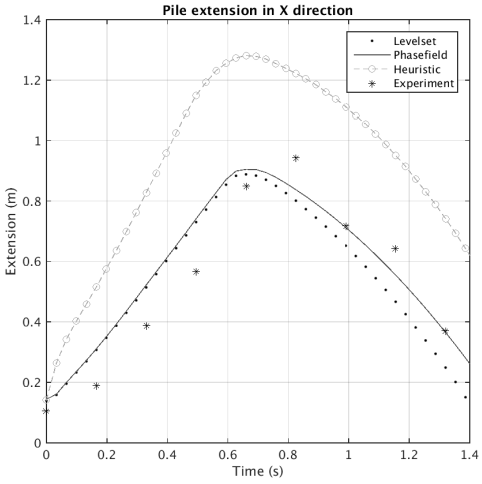
the particular flow rendering the use of TITAN for anything but hindcasts problematic. A primary goal of this study is to find a method that can predict the interface of a geophysical flow independent of the scale of the flow, and for this reason we selected two very different scales of flows (i.e flow over the inclined plane and Colima volcano), and the $GEOFLOW\_TINY$ that we use here have been tested in much of our previous work, and has showed reasonable results [35, 36].

In figure 9a the extent of flow in X direction was plotted. This plot shows the length of flow during its travel. The maximum of the flow length happens when the flow reaches the horizontal plane. All of the methods are successful in predicting the time that flow reaches the horizontal plane, but like the interface results (figure 8) the heuristic method over-predicts the length of the flow. The Phase Field and the Level Set results are a better match with the experimental result.

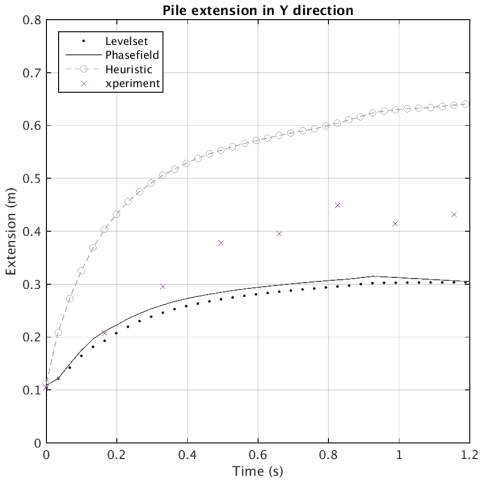
The width of the flow or extent of the flow in Y direction can be seen in figure 9c. There is larger difference between the experiment and the Eulerian interface capturing methods results in this figure than the other results, and based on our previous work [36], we believe the main reason for this difference is due to the accuracy of the solver. In that work, we showed with more accurate simulations using a higher order Discontinuous Galerkin method we can better capture the width of the flow. Future work will address the combination of such higher order methods and the interface capturing schemes.

The other measure we use to compare the results is the area of the flow. In figure 9b one can see that the Eulerian methods capture the area of the flow fairly well, especially before the flow hits the horizontal surface, while the heuristic method over-estimates the area of the flow. The last result is that of the perimeter of the flow (figure 9d), which shows a similar behavior to the other results that have been discussed earlier.

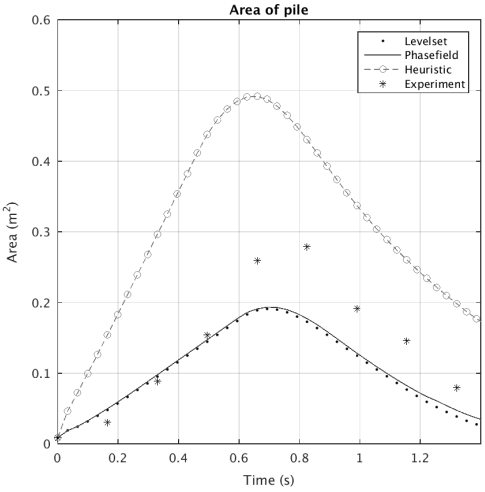
From these results, all three methods successfully overcome the thin layer problems that were introduced earlier (problems 1-4 section 1), while the Eulerian interface capturing methods generally captured the flow with higher accuracy than the heuristic method.



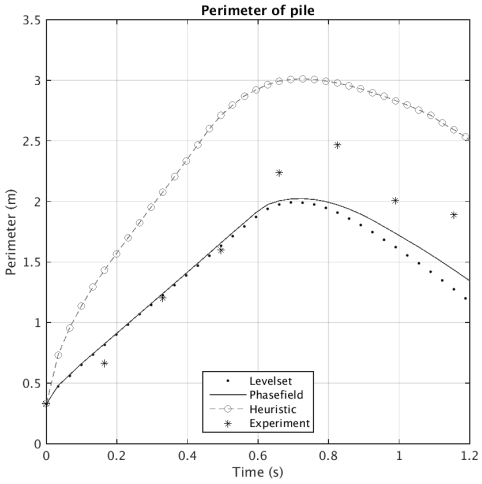
(a) Extension of pile in X direction.



(c) Extension of pile in Y direction.

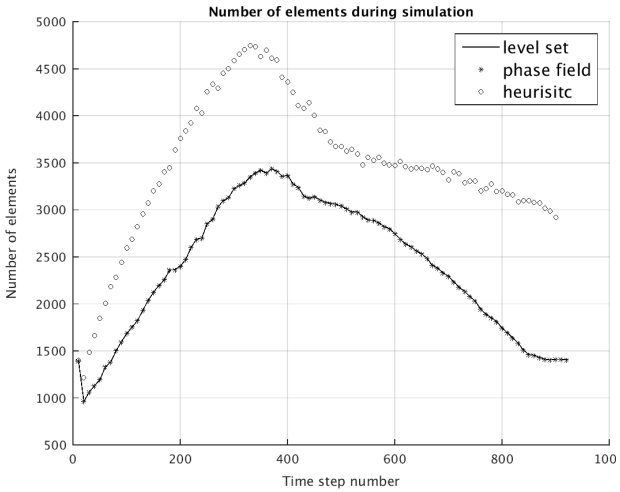


(b) Area of pile.

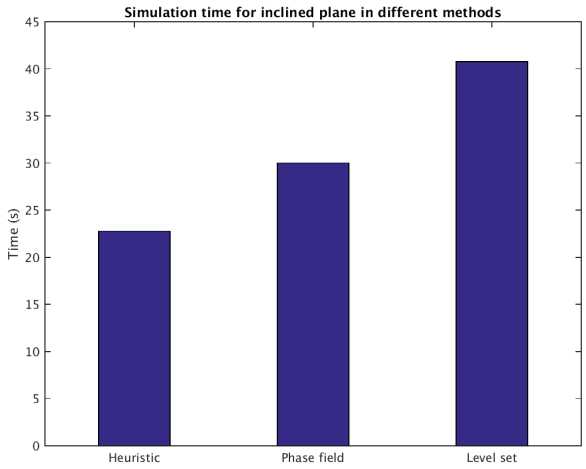


(d) Perimeter of pile.

Figure 9: Comparison of the methods for flow on inclined plane.



(a) Number of the elements for different method of interface capturing during the simulation.



(b) Time spent for each simulation.

Figure 10: Comparison of the methods based on computational cost.

Finally, consider the computational cost of the three methods, as shown in figure 10. The first plot is the number of elements during the simulation and the second one is the wall clock time spent for each simulation Since we are using the results of the interface capturing methods for adaptive mesh refinement as described in section 3.2, the number of elements during the simulation changes and depends on the different interface capturing methods we use. Based on these results, as can be predicted from similarity of the results in the Eulerian interface capturing methods figures 9 and 8, both Eulerian interface capturing methods have very similar number of elements during the simulation while the interface capturing with the heuristic method results in a larger number of elements. Despite the higher number of elements in the heuristic method, the overall time needed to complete a simulation, figure 10b, is lowest for the heuristic method. Overall the heuristic method took 70% of the time required for the Phase Field method and only 55% of the time required for the Level Set method. All of the simulations were performed on one Intel(R) Xeon(R) CPU E3-1220 v3 3.10GHz. Clearly, there is a tradeoff in accuracy and cost.

5.2 Colima Volcano

In this section, we verify our different interface capturing methods using field data from an eruption of the Colima volcano. The Colima volcano in Mexico is one of the most active volcanoes in North America, and this eruption happened on 16-17 April 1991 [12].

Pile shape	Parabolic
Location of pile in X	644956.0
Location of pile in Y	2157970.0
Maximum pile height	30 m
Major extent of the pile	55 m
Minor extent of the pile	55 m
Bed friction angle	37^o
Internal friction angle	20^o
Physical time simulated	3600 sec

The topography of the Colima volcano is such that small changes in the initial location of the pile leads to a completely different path of flow, so good performance of any of the interface capturing methods for this

case promises a reliable method for other volcanoes. Unfortunately, only in very rare cases is data during a volcanic eruption available, and thus we cannot perform a direct comparison between the numerical results with time-resolved field data. Usually for such applications, the deposit of the volcanic event is used to find the outline of the flow and to validate the numerical results.

In figure 12 we compared the results of the three methods with the outline of the flow from the eruption obtained from the field data [40]. In the numerical simulations the outline of the flow is the boundary between wet and dry areas of the domain for whole the simulation. In other words, the points where the flow has passed are inside the outline, while the rest of the points are located outside of the outline of the flow.

As in the inclined plane case, we use the same definition for the interface. Thus $\phi = 0$ and $\Psi = 0$ are used for the Phase Field and the Level Set methods, respectively, and $h = \sqrt[3]{Vol} \times GEOFLOW\_TINY$ is used for the heuristic method to determine the wet and dry areas.

The resolution of the digital elevation model (DEM) of Colima volcano that we use here is 5 meters, which is the finest DEM that we have ever used for this volcano. The outline of this eruption is available in [33]. We used the Keyhole Markup Language (KML) to plot the results on real terrain information by using the Google earth application 12.

In figure 11 one can see the plotted outline of the eruption on Colima volcano. This outline is compared to the numerically determined results in figure 12.

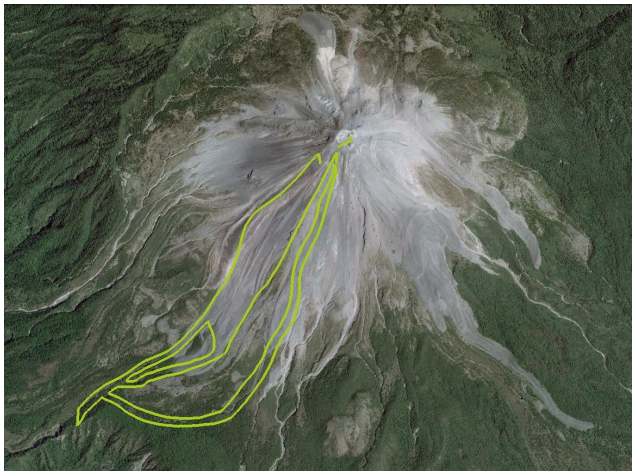
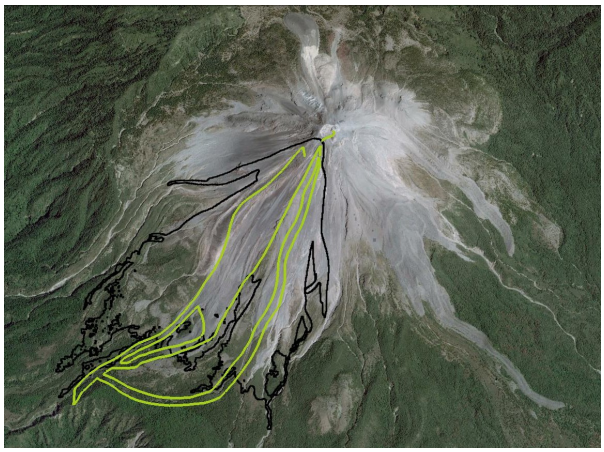
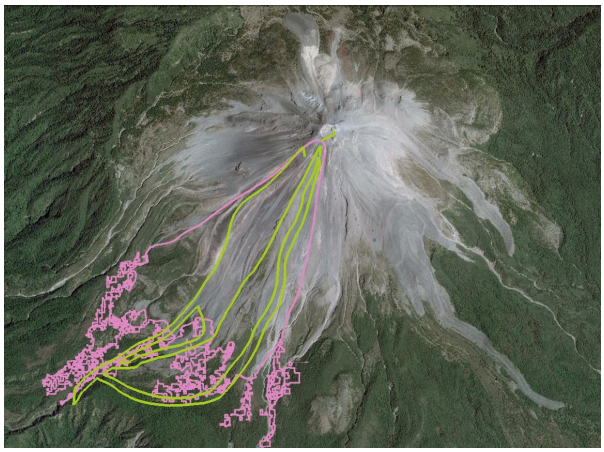


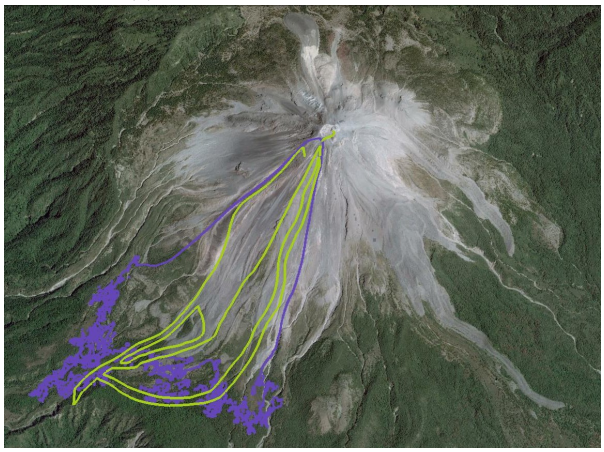
Figure 11: Outline of flow for Colima volcano for eruption 16-17 April 1991.



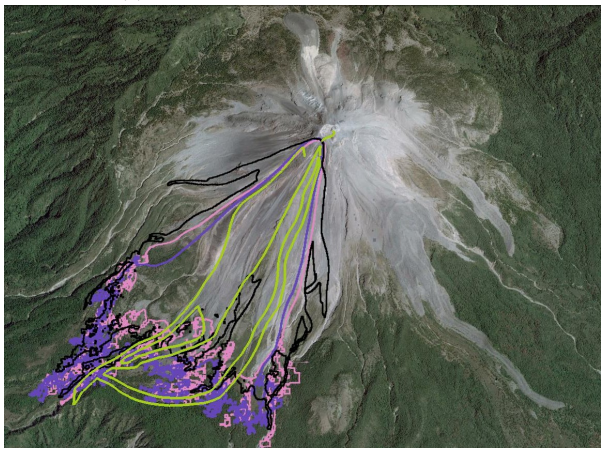
(a) Outline from Heuristic Method.



(c) Outline from Phase field Method.



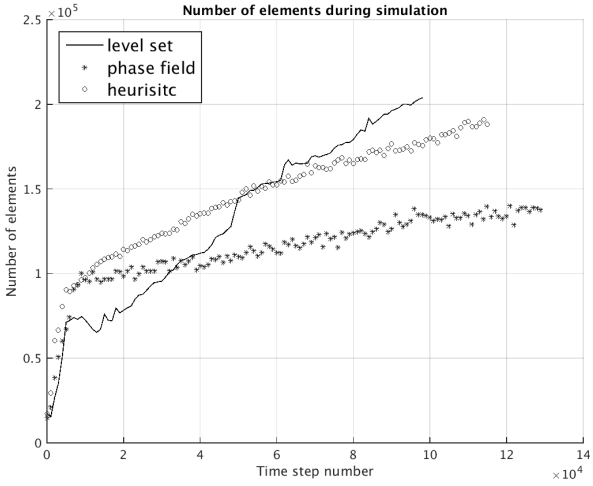
(b) Outline from Level Set Method.



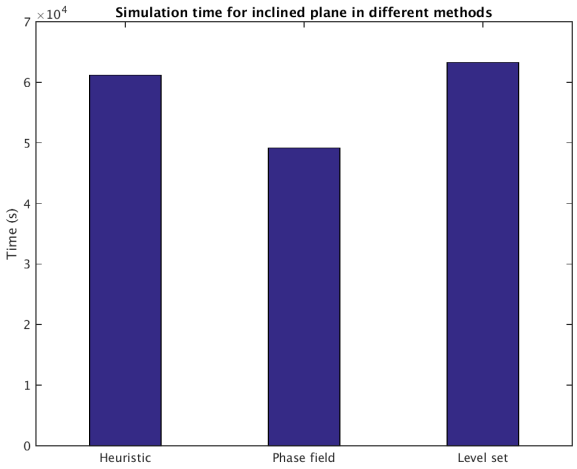
(d) Comparison of all three methods.

Figure 12: Comparison of the obtained outline of flow from Heuristic, Level Set, and Phase field methods for Colima Volcano.

Results of figure 12 shows that all of the methods are successful in controlling the thin layer problem. The length of the flow has been captured in all of the methods very well. This is very important for generating hazard maps. The width of the flow is also reasonable in all of the three methods, although the Eulerian methods are slightly more accurate. The heuristic method captures the separation of the flow into two channels that can be seen in the field deposit data. The heuristic method also indicates the presence of two small channels that are not present in the field outline. Note that we are comparing deposits to field data.



(a) Number of the elements for different method of interface capturing during the simulation.



(b) Time spent for each simulation.

Figure 13: Comparison of the methods based on computational cost.

The computational cost comparison of the methods for the Colima volcano simulations are shown in figure 13. We used 16 x 2.27GHz Intel(R) Xeon(R) L5520 for this case. Figure 13a shows the number of the elements during the simulation for the three methods. This figures shows the heuristic method and level set have a higher number of elements than the phase field method, and both require a higher number of iterations to perform 1 hr physical time simulation. The right figure (figure 13b) shows the required time for the simulation. In this figure, we can see all of the methods take almost the same amount of computational time, which means that although the Level Set and the Phase Field methods are more computationally expensive per time step, the lower number of elements and lower number of iterations help them to have the same computational cost as the heuristic method.

6 Conclusions

The numerical solution of the Savage-Hutter (and similar "shallow-water") equations have historically been plagued by several interrelated numerical difficulties which are collectively characterized by a non-physically thin-layer extending large distances from the realistic main body of the flow. In the best case, this "thin-layer problem" means a "no flow" boundary line must be arbitrarily drawn at some given depth contour. In the worst case, it can cause severe numerical instability that prevents any simulation of a particular event. In this paper, we have described some features of the thin-layer problem, some underlying causes that are common to virtually all numerical solution methodologies for SW equations. We presented a heuristic method and compared it to two Eulerian interface capturing approaches, namely phase filed and level set based methods, that mitigate this problem. In the heuristic method we used a threshold to distinguish between the wet and dry areas, and in the level set and phase field methods we solved a transport equation that implicitly represents the interface of the flow. Then we used the result of a previous step for mesh refinement in all of the three solvers to control the thin layer problem and other related problems. Moreover, in the heuristic approach we also used this result for interface reconstruction and the adjustment of the flux. We implemented these thin-layer control strategies in TITAN2D, our high performance finite volume solver of the depth-averaged granular flow equations. Numerical simulations were performed for two different geophysical mass flows. To verify the solvers, numerical experiments were conducted for an inclined plate and the Colima volcano. For the first case we verified the numerical results with experimental results, and for the second case

we compared the results with field data. These analyses show that with all of the approaches, we not only prevented the loss of numerical stability but also demonstrated behavior that is consistent with observations of the flows. On the basis of these very positive results, we conclude that our thin-layer control strategy, and interface capturing approach provides sufficient benefit. While all of these approaches to thin-layer mitigation was developed in the context of TITAN2D's capabilities, much of it should be appropriate for use in depth-averaged flow solvers with different numerical schemes and physics of the problem.

Acknowledgements

We acknowledge the Colima deposit and experiment data made available to us by Prof. M. I. Bursik and for helpful discussions on the use of that data.

References

[1] D Adalsteinsson and J.a Sethian. The Fast Construction of Extension Velocities in Level Set Methods. *Journal of Computational Physics*, 148(1):2–22, January 1999.

[2] D M Anderson, G B McFadden, and A A Wheeler. Diffuse-Interface Methods in Fluid Mechanics. *Annual Review of Fluid Mechanics*, 30(1):139–165, 1998.

[3] F. Aureli, a. Maranzoni, P. Mignosa, and C. Ziveri. A weighted surface-depth gradient method for the numerical integration of the 2D shallow water equations with topography. *Advances in Water Resources*, 31(7):962–974, July 2008.

[4] Satish Balay, Shrirang Abhyankar, Mark˜F. Adams, Jed Brown, Peter Brune, Kris Buschelman, Victor Eijkhout, William˜D. Gropp, Dinesh Kaushik, Matthew˜G. Knepley, Lois Curfman McInnes, Karl Rupp, Barry˜F. Smith, and Hong Zhang. {PETS}c Users Manual. Technical Report ANL-95/11 - Revision 3.5, Argonne National Laboratory, 2014.

[5] Andrea Balzano. Evaluation of methods for numerical simulation of wetting and drying in shallow water flow models. *Coastal Engineering*, pages 83–107, 1998.

[6] W. J. Boettinger, J. a. Warren, C. Beckermann, and A. Karma. Phase -Field Simulation of Solidification. *Annual Review of Materials Research*, 32(1):163–194, August 2002.

[7] Lia Bronsard and Robert V Kohn. On the slowness of phase boundary motion in one space dimension. *Communications on Pure and Applied Mathematics*, 43(8):983–997, 1990.

[8] Shintaro Bunya, Ethan J. Kubatko, Joannes J. Westerink, and Clint Dawson. A wetting and drying treatment for the Runge–Kutta discontinuous Galerkin solution to the shallow water equations. *Computer Methods in Applied Mechanics and Engineering*, 198(17-20):1548–1562, April 2009.

[9] John W Cahn. Free Energy of a Nonuniform System. II. Thermodynamic Basis. *The Journal of Chemical Physics*, 30(5), 1959.

[10] John W Cahn and John E Hilliard. Free Energy of a Nonuniform System. I. Interfacial Free Energy. *The Journal of Chemical Physics*, 28(2), 1958.

[11] M.J. Castro, A.M. Ferreiro Ferreiro, J.A. García-Rodríguez, J.M. González-Vida, J. Macías, C. Parés, and M. Elena Vázquez-Cendón. The numerical treatment of wet/dry fronts in shallow flows: application to one-layer and two-layer systems. *Mathematical and Computer Modelling*, 42(3-4):419–439, August 2005.

[12] S J Charbonnier and R Gertisser. Field observations and surface characteristics of pristine block-and-ash flow deposits from the 2006 eruption of Merapi Volcano, Java, Indonesia. *Journal of Volcanology and Geothermal Research*, 177(4):971–982, 2008.

[13] Long-Qing Chen. {PHASE}-{FIELD} {MODELS} {FOR} {MICROSTRUCTURE} {EVOLUTION}. *Annual Review of Materials Research*, 32(1):113–140, August 2002.

[14] Liang Cheng and Steven Armfield. A simplified marker and cell method for unsteady flows on non-staggered grids. *International Journal for Numerical Methods in Fluids*, 21(1):15–34, 1995.

[15] David L. Chopp. Some Improvements of the Fast Marching Method. *SIAM Journal on Scientific Computing*, 23(1):230–244, 2001.

[16] Keith R Dalbey. *Predictive Simulation and Model Based Hazard Maps*. PhD thesis, University at Buffalo, 2009.

[17] Roger P. Denlinger and Richard M. Iverson. Flow of variably fluidized granular masses across three-dimensional terrain: 2. Numerical predictions and experimental tests. *Journal of Geophysical Research*, 106(B1):553, 2001.

[18] L. D'Alpaos and a. Defina. Mathematical modeling of tidal hydrodynamics in shallow lagoons: A review of open issues and applications to the Venice lagoon. *Computers & Geosciences*, 33(4):476–496, May 2007.

[19] M E Eglit and Ye I Sveshnikova. Matematicheskoye modelirovaniye snezhnykh lavin; mathematical modeling of snowavalanches. *Materialy Glyatsiologicheskikh Issledovaniy, Khronika Obsuzhdeniya*, 38:79–84, 1980.

[20] L Fraccarollo and E F Toro. Experimental and numerical assessment of the shallow water model for two dimensional dam-break type problems. *Journal of Hydraulic Research*, 33:843–864, 1995.

[21] D Gerlach, G Tomar, G Biswas, and F Durst. Comparison of volume-of-fluid methods for surface tension-dominant two-phase flows. *International Journal of Heat and Mass Transfer*, 49(3-4):740–754, 2006.

[22] James Glimm, John W Grove, Xiao Lin Li, Keh-Ming Shyue, Yanni Zeng, and Qiang Zhang. Three Dimensional Front Tracking. *SIAM J. Sci. Comp*, 19:703–727, 1995.

[23] V R Gopala and B G M van Wachem. Volume of fluid methods for immiscible-fluid and free-surface flows. *Chemical Engineering Journal*, 2008.

[24] J. M. N. T. Gray, M. Wieland, and K. Hutter. Gravity-driven free surface flow of granular avalanches over complex basal topography. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 455(1985):1841–1874, May 1999.

[25] Francis H Harlow and J Eddie Welch. Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface. *Physics of Fluids*, 8:2182–2189, 1965.

[26] C W HIRT and B D NICHOLS. Volume of fluid/VOF/ method for the dynamics of free boundaries. *Journal of Computational Physics*, 39(1):201–225, 1981.

[27] R.˜M. Iverson. The physics of debris flows. *Reviews of Geophysics*, 35:245–296, 1997.

[28] Georges Kesserwani and Qiuhua Liang. Locally Limited and Fully Conserved RKDG2 Shallow Water Solutions with Wetting and Drying. *Journal of Scientific Computing*, 50(1):120–144, March 2011.

[29] K.Hutter, M Siegel, S B Savage, and Y Nohguchi. Two-dimensional spreading of a granular avalanche down an inclined plane part I. Theory. *Acta Mechanica*, 100:37–68, 1993.

[30] Junseok Kim. Phase-Field Models for Multi-Component Fluid Flows. *Commun. Comput. Phys*, 12(3):613–661, 2012.

[31] Junseok Kim, Seunggyu Lee, and Yongho Choi. A conservative Allen – Cahn equation with a space – time dependent Lagrange multiplier. *International Journal of Engineering Science*, 84:11–17, 2014.

[32] R G Larson. *The Structure and Rheology of Complex Fluids*. Topics in Chemical Engineering. OUP USA, 1999.

[33] Laercio M Namikawa. *Multiple Representations of Elevation for Dynamic Process Modeling*. PhD thesis, Department of Geography, University at Buffalo, July 2006.

[34] Stanley Osher and James A Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79(1):12–49, 1988.

[35] Abani K. Patra, a. C. Bauer, C. C. Nichita, E. B. Pitman, M. F. Sheridan, M. Bursik, B. Rupp, a. Webber, a. J. Stinton, L. M. Namikawa, and C. S. Renschler. Parallel adaptive numerical simulation of dry avalanches over natural terrain. *Journal of Volcanology and Geothermal Research*, 139(1-2):1–21, January 2005.

[36] Abani K. Patra, C Nichita, A Bauer, E Pitman, M Bursik, and M Sheridan. Parallel adaptive discontinuous Galerkin approximation for thin layer avalanche modeling. *Computers & Geosciences*, 32(7):912–926, August 2006.

[37] E Bruce Pitman, C Camil Nichita, Abani Patra, Andy Bauer, Michael Sheridan, and Marcus Bursik. Computing granular avalanches and landslides. *Physics of Fluids*, 15(12):3638, 2003.

[38] S.˜P. Pudasaini and K Hutter. Rapid shear flows of dry granular masses down curved and twisted channels. *Journal of Fluid Mechanics*, 495:193–208, November 2003.

[39] M Quecedo and M Pastor. A reappraisal of Taylor-Galerkin algorithm for drying-wetting areas in shallow water computations. *International Journal for Numerical Methods in Fluids*, 38(6):515–531, 2002.

[40] Byron Rupp, Marcus Bursik, Laercio Namikawa, Amy Webb, Abani K. Patra, Ricardo Saucedo, José Luis Macías, and Chris Renschler. Computational modeling of the 1991 block and ash fows at Colima Volcano, Mexico. 2402(11):223–237, 2006.

[41] S B Savage and K Hutter. The motion of a finite mass of granular material down a rough incline. *Journal of Fluid Mechanics*, 199:177–215, 1989.

[42] S B Savage and R M Iverson. Surge dynamics coupled to pore-pressure evolution in debris flows. *Debris-Flow Hazards Mitigation: Mechanics, Prediction and Assessment, edited by: Rickenmann, D. and Chen, C.-L., Millpress*, 1:503–514, 2003.

[43] Mark Sussman, Peter Smereka, and Stanley Osher. A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow. *Journal of Computational Physics*, 114(1):146–159, 1994.

[44] Y.C. Tai, S. Noelle, J.M.N.T. Gray, and K. Hutter. Shock-Capturing and Front-Tracking Methods for Granular Avalanches. *Journal of Computational Physics*, 175(1):269–301, January 2002.

[45] Tayfun E Tezduyar, Sunil Sathe, Matthew Schwaab, and Brian S Conklin. Arterial fluid mechanics modeling with the stabilized space – time fluid – structure interaction technique. *International Journal for Numerical Methods in Fluids*, (October 2007):601–629, 2008.

[46] Tayfun E Tezduyar, Sunil Sathe, Matthew Schwaab, and Brian S Conklin. Arterial fluid mechanics modeling with the stabilized space – time fluid – structure interaction technique. *International Journal for Numerical Methods in Fluids*, (October 2007):601–629, 2008.

[47] E F Toro. *Shock-Capturing Methods for Free-Surface Shallow Flows*. Wiley, New York, 2001.

[48] E F Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. Springer, 2009.

[49] Salih Ozen Unverdi and Grétar Tryggvason. A Front-tracking Method for Viscous, Incompressible, Multi-fluid Flows. *J. Comput. Phys.*, 100(1):25–37, May 1992.

[50] Amy Webb. Granular flow experiments to validate numerical flow model, Titan2D, 2004.

[51] Luiz C Wrobel and C A Brebbia. *Computational Modelling of Free and Moving Boundary Problems: Fluid flow*, volume 1. Walter de Gruyter, 1991.

[52] Xiaofeng Yang, James J Feng, Chun Liu, and Jie Shen. Numerical simulations of jet pinching-off and drop formation using an energetic variational phase-field method. *Journal of Computational Physics*, 218(1):417–428, 2006.

[53] D L Youngs. Time-dependent multi-material flow with large fluid distortion. In K W Morton and M J Baines, editors, *Numerical Methods for Fluid Dynamics*, pages 273–285. Academic Press, 1982.