TITAN2D User Guide

Release 2.0.0, 2007.07.09

Geophysical Mass Flow Group (GMFG), University at Buffalo, NY, USA

July 27, 2007

Contents

List of Figures

Chapter 1

Introduction to TITAN2D

TITAN2D is a computer program developed for the purpose of simulating dry granular avalanches over digital elevation models of natural terrain. The program is designed for simulating geological mass flows such as debris avalanches and landslides. TITAN2D combines numerical simulations of a flow with digital elevation data of natural terrain supported through a Geographical Information System (GIS) interface.

The TITAN2D program is based upon a depth-averaged model for an incompressible Coulomb continuum, a "shallow-water" granular flow. We will briefly review the functionality here; details of the modeling and numerical methodology may be found in the literature [1] [2]. The conservation equations for mass and momentum are solved with a Coulomb-type friction term for the interactions between the grains of the media and between the granular material and the basal surface. The resulting hyperbolic system of equations is solved using a parallel, adaptive mesh, Godunov scheme. The Message Passing Interface (MPI) [http://www-unix.mcs.anl.gov/mpi/] Application Programmers Interface (API) allows for computing on multiple processors, increasing computational power, decreasing computing time, and allowing for the use of large data sets. Adaptive gridding allows for the concentration of computing power on regions of special interest.

---

[1] A.K. Patra, A.C. Bauer, C.C. Nichita, E.B. Pitman, M.F. Sheridan, M. Bursik, B. Rupp, A. Webber, A. Stinton, L. Namikawa, and C. Renschler, *Parallel Adaptive Numerical Simulation of Dry Avalanches Over Natural Terrain*, Journal of Volcanology and Geophysical Research, **139** (2005) 1-21

[2] E.B. Pitman, C.C. Nichita, A.K. Patra, A.C. Bauer, M.F. Sheridan, and M. Bursik, *Computing Granular Avalanches and Landslides*, Physics of Fluids, Vol. 15, Number 12 (December 2003)

Mesh refinement captures the complex flow features at the leading edge of the flow, as well as locations where the topography changes rapidly. Mesh unrefinement is applied where solution values are relatively constant or small to further improve computational efficiency.

The model used in TITAN2D assumes a pile of granular material, pulled downslope by gravity. Friction between particles and between particles and ground resist this momentum. Governing equations for this model, the conservation of mass and conservation of momentum, are solved using approximate numerical solution methods, e.g. finite volumes etc. The direct outputs of TITAN2D are flow depth and momentum. These may then be used to compute, at different points, field observable variables like run-up height, inundation area and time of flow.

TITAN2D operates via a python scripted Graphical User Interface (GUI). Through this interface the user inputs the parameters needed to successfully run the program such as pile dimensions, starting coordinates, internal and bed friction angles, and simulation time. The simulation is computed on a Digital Elevation Model (DEM) of the desired region and results can be displayed through the TITAN2D viewer utilities, or other visualization software packages. The TITAN2D viewer utilities are designed to present end-users with a clear representation of various properties of the mass flow such as pile height and velocity magnitude. The attributes embedded in the data elements that constitute the polygonal mesh are color-coded and applied as a texture over the terrain.

Chapter 2

New Features in Titan Release 2.0.0, 2007.07.09

In this release of TITAN2D, several improvements have been incorporated that add additional capabilities to the simulator, fix previous numerical difficulties with correctly capturing thin-layer flow, and increase the efficiency of the data repartitioning process. These are summarized as follows:

- Included a restart capability that allows the continuation of a previously run simulation without having to start over from the beginning.

- Added option to include "Flux Sources" - This allows the simulation of material that actively extrudes from the ground at a specific rate over a specified period of time. Previously, only a fixed amount of material could be simulated in the form of a pile placed on a sloped terrain and subjected to gravitational forces. Now, simulations can be run using either piles or flux-sources exclusively, or by using a combination of any number of each.

- The initial layout of the computational mesh and maximum level of grid-cell refinement is now determined from the user's specification of the number of computational cells across the smallest pile/flux-source diameter. Prior to this update, the grid-cell density was determined from the user selecting the number of computational

cells spanning the map in a single direction.

- Added option to implement "Discharge Planes" during a simulation run - A discharge plane is a vertically-oriented 2-dimensional plane such that all the material passing over the line it forms on the map (when viewed top-down) is tallied throughout the course of a simulation. This allows the user to track the volume of material moving into or out of any area of the DEM map (for example, by using 4 discharge planes to box-in a particular region).

- Implemented several methods of controlling the unrealistically fast flow that occurs in thin material layers around the edges of a pile, such as:

  - Modifying the adaptive meshing process to include several rings of maximally refined "buffer" cells placed at/near the flow boundary to suppress the numerical error introduced there. This also guarantees that flow at the piles' boundaries will only flow into *maximally* refined cells.

  - Switching from the standard approach of computing velocity to using chain-rule differentiation (resulting in a form of L'Hopital's Rule) to compute the flow velocity when encountering excessively thin layers (used for statistical reporting purposes only).

- Added new viewer capability - Titan2D can now generate data output files that can be viewed using the freely available *paraview* visualization application (http://www.paraview.org). See section 4.1 for basic usage instructions.

- Updated repartitioning function - We have introduced an initial gridding trick, i.e. an automatic division of the computational domain into an appropriate number of subdomains, that has the effect of decreasing the amount of information that needs to be exchanged between non-sequential processors during a multi-processor run of Titan, saving computation time.

Chapter 3

Getting Started

3.1 System Requirements

The TITAN software is open source and is built using only other open source systems. It is designed for both low end single processor use and high end distributed/shared memory multi-processor use. The installation and use procedure is largely similar but there are small differences on each system. We provide both source code and other open source software such as MPICH 2 for TITAN2D and Coin-2 libraries (Open Inventor Clone) for the GMFG Viewer.

*Note: As of this release, the use of the GMFG Viewer is depreciated and users are encouraged to migrate to *paraview*.

3.2 TITAN2D Program File

The latest edition of the TITAN2D program can be obtained by contacting the GMFG (Geophysical Mass Flow Group) at the University at Buffalo - led by Dr. Abani Patra (Professor, Dept. of Mechanical and Aerospace Engineering), or for users outside of UB, from the GMFG website: **http://www.gmfg.buffalo.edu**.

3.2.1   Compiling the Code

Once the file package is downloaded, unzipped, and untarred, the user must descend into the ".../titan" directory to compile the code.

Prerequisite Steps (all users):

1. First make sure that MPI is installed on your computer (as this is required to build Titan). Use either the version available from the gmfg website, or you can use a previously installed version, already on your machine.

2. Next, if you will be using the paraview viewer to view Titan output, check that HDF5 is installed on your machine. This is also available on the gmfg website.

3. Also make sure that Tkinter is installed on your system. This is necessary to run Titan's python GUI. (To ensure this when installing Linux: choose a *custom* installation and simply elect to *Install Everything.*)

4. Continue with compiling the code (see steps below).

Users at the University at Buffalo, compile the program using the command:

```
> sh ub-compile.sh
```

Users outside of the University at Buffalo:

Method 1 - Using the command sequence "configure", "make", "make install".

- If efficient use of paraview is desired, type:
  ```
  > configure --with-mpi=<mpi main directory> --with-hdf5=<hdf5 directory>
  > make
  > make install
  ```

- If efficient paraview support is not required, then type:
  ```
  > configure --with-mpi=<mpi main directory> --without-hdf5
  ```

```
> make
> make install
```

Method 2 - running an included installation script by typing:

```
> sh install-titan.sh
```

- This script will automatically check for both MPI and HDF5 installation. If MPI is not found in its standard location, then the user will be prompted to enter its absolute path, or quit installation. If this script is unable to locate HDF5, then the user will be prompted to enter its absolute path or continue without installing HDF5.

- If there are multiple copies of MPI on your machine, this script will find and use the first one in your path.

(Note: If the standard makefiles are incompatible with your system, you will need to regenerate them using automake v. 1.6 or higher, aclocal, autoconf v. 2.52 or higher, then use the "configure" command as normal. Refer to the included README.AUTOCONF file for helpful details.)

After successfully compiling TITAN2D, the "bin" directory will appear in the ".../titan" directory. You will need to access this bin directory to run TITAN2D.

3.3   GRASS GIS Data file

*__NOTE:__   TITAN2D runs independently of GRASS. That is, GRASS is *NOT REQUIRED* to run TITAN2D.

TITAN2D performs flow simulations on a DEM of a user-defined region.   The DEM data file, containing elevation data, must be formatted to operate in a GRASS (Geographic Resources Analysis Support System) GIS environment. GRASS GIS is an open source GIS with raster, topological vector, image processing, and graphics production functionality. It operates on various platforms through a graphical user interface

11

and shell in X-Windows. GRASS is available free of charge under GNU General Public License (GPL) [http://grass.ibiblio.org/].

Simulation accuracy is highly dependent on the level of DEM resolution and quality. DEMs with higher resolutions (e.g. 5-30m) render more accurate representations of actual geophysical flow events, especially in situations where channelized flow is involved.

The GRASS GIS interface used in conjunction with the TITAN2D GUI allows the user to adjust the area of the desired initial DEM. This capability decreases output file size thus increasing visualization speed and allows the user to focus attention upon a specific region of interest within a much larger DEM.

3.4   Instructions for Using TITAN2D

After obtaining the latest version of TITAN2D and configuring a GRASS DEM of the desired region, the user is ready to simulate a geologic flow event. First, go to the ".../titan/bin/" directory. Then type "**python titan_gui.py**". This command will open the TITAN2D GUI.

3.4.1   TITAN2D Graphical User Interface (GUI)

Figure 3.1 shows the GUI for TITAN2D. The user inputs information into the boxes prior to running the simulation. The information required falls into one of three types: 1) GIS data specifications, 2) Computational parameters and 3) Pile, flux-source, and material property parameters.

1. Specifying the GIS Information: The first six options ask the user to input information on the GIS data to be used in the simulation:

   Main Data Entry Window

   - The **GIS Information Main Directory** box displays the location of the GIS datasets. For example, this may be "/computer_name/home/username/ directory_name/grass.data/grass5/". The directory "./grass5/" is the top GIS
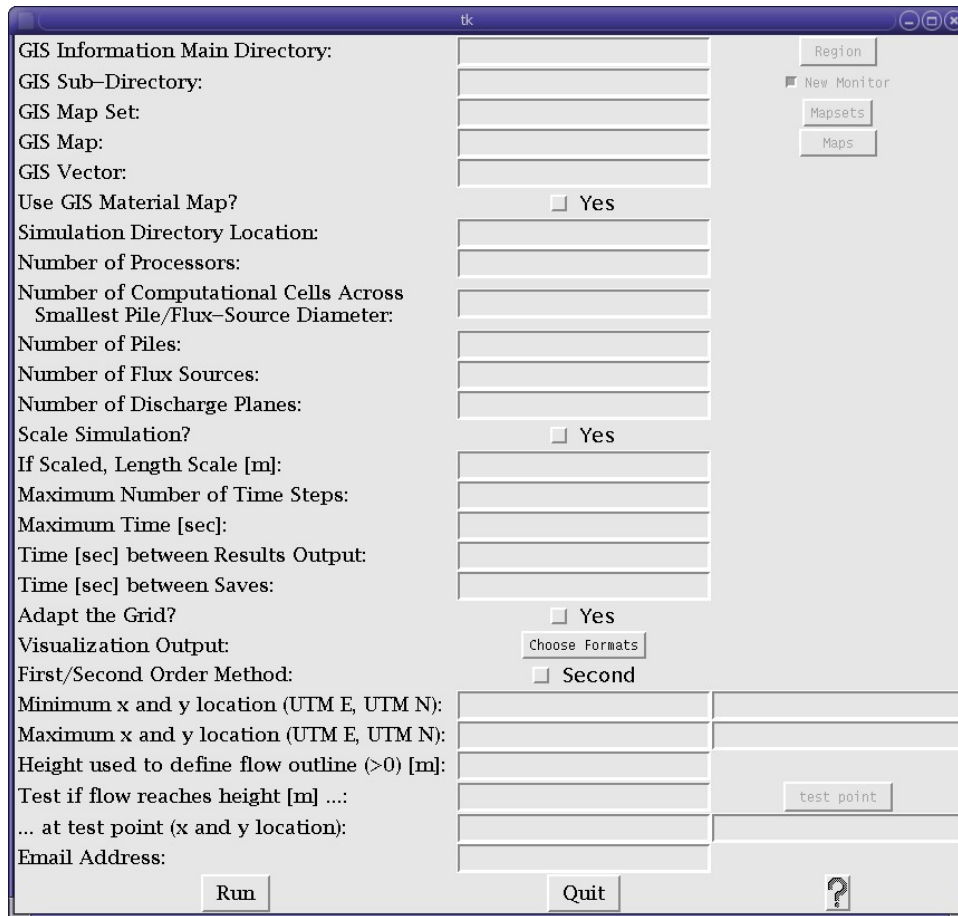
Figure 3.1: Sample Python Graphical User Interface (GUI)

directory in which all the data is stored. Within this directory, subdirectories containing separate datasets are found. When running the simulation, a specific subdirectory is entered into the **GIS Sub-Directory** box.

- The **GIS Map Set** and **GIS Map** point to the DEM dataset that the user chooses to use in the simulation. The "GIS Map" must be found within the "GIS Map Set" Directory. Enter the names of both into the appropriate box on the GUI (see figure 3.2). If using TITAN2D through the GRASS interface, follow the directions in Section 3.4.2.

- *NOTE: The next feature is available only to users who have a specific material map (file ends in _Mat)*. The **Use GIS Material Map?** check box enables the input of a GIS-based surficial material map that matches the area covered

13

by the DEM. This map is used to define the zones in the region where changes in the surface morphology results in a change in the basal friction angle. When this function is enabled, pop-up windows (appearing after the "run" button is clicked) will ask for the internal and basal friction angles for each material represented on the material map.

While the pop-up window asks for both the internal and bed friction angles for each material, TITAN2D uses only the first entered internal friction angle. It is not necessary to enter the values for every material; the angles entered into the first window will be automatically carried over to the next if the values are unchanged. You may enter the internal friction angle in the first window and change only the basal angles in the subsequent windows.
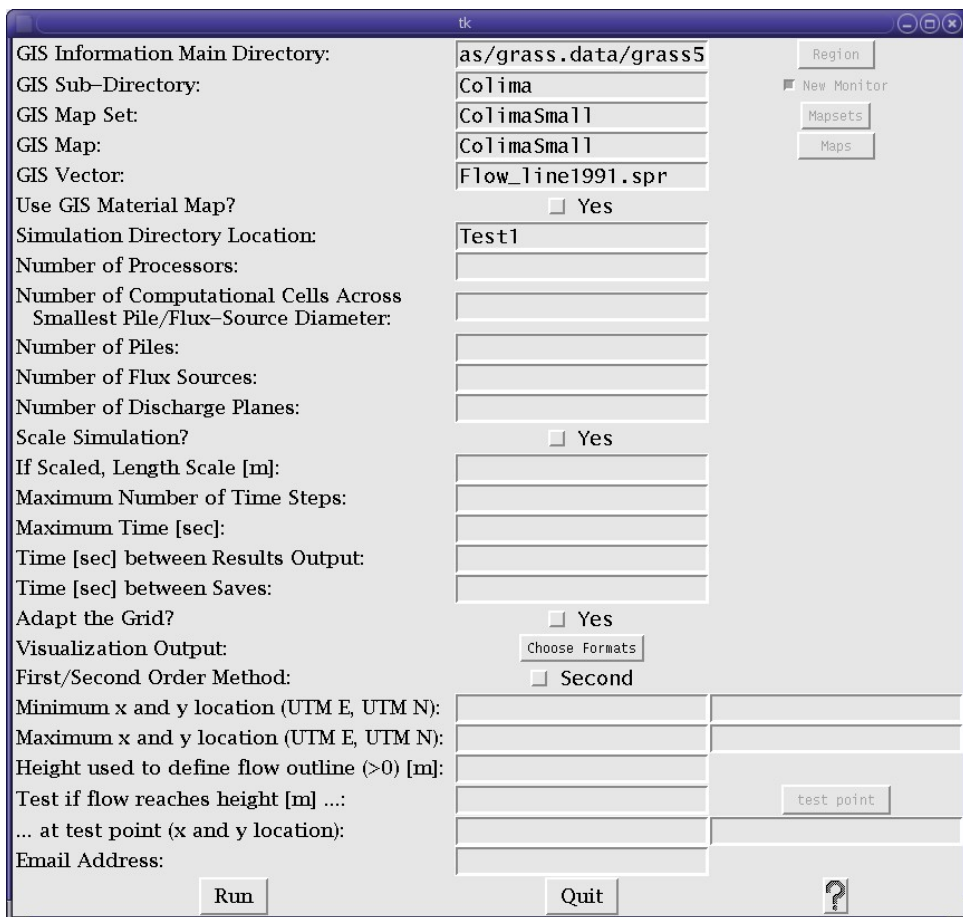
Figure 3.2:   Entering GIS data into Graphical User Interface

2. <u>The Computational Parameters</u>: The next inputs relate to the actual computation and output of the data.

- The **Simulation Directory Location** specifies the directory where the output data files will be stored (see Figure 3.2). A directory must be specified in this box. If the specified directory already exists, it will not be overwritten and data will not be submitted for processing. The specified directories are found relative to the "/computer_name/home/username/directory_name /titan/bin/" directory. For example, the directory path to the simulation directory shown in fig. 3.2 would be ".../titan/bin/Test1/" and the specified directory would be "Test1". Every time the simulation runs, a new simulation directory must be created.

Figure 3.3: Entering Computational Data into the Graphical User Interface

- The number of processors that the user decides to use must be specified in the **Number of Processors** box. If more than one processor is specified, each gets a share of the processing load, thus decreasing simulation run time. However, if too many processors are specified, the data submitted may be queued until the number of chosen processors become available. The user must be aware of the number of available processors on any given machine. An excellent website to check the status of Center for Computational Research (CCR) machines is: http://www.ccr.buffalo.edu/hotpages/content/main.htm. At this site one can find out how many processors a particular machine offers as well as its current workload.

- TITAN2D creates a regular grid/mesh on which the computation takes place. The value specified by the user in the **Number of Computational Cells Across Smallest Pile/Flux-Source Diameter** box is used by Titan to determine the maximum level of grid-cell refinement allowed throughout the simulation. (See figure 3.4 for a basic illustration of this process.) Using the default case as an example, which occurs if the user leaves this field blank, the value of 20 will be used. Then, at the beginning of a simulation, the grid-cell containing the centroid of each pile is successively refined (i.e. split into 4 "son" cells) until it's size is no larger than 1/20th of the smallest pile or flux source diameter. This is the size of the smallest computational cell allowed on the map during the course of the simulation. Although (with an adaptive grid) maximally-refined cells will not occur everywhere, but rather only where they are needed (*only* in close proximity to a pile's boundaries *or* where ever a flow's mass/momentum fluxes become large), it is important to choose a value here that strikes a balance between computation time and calculation accuracy. If there are many debris piles, they will all have maximally refined cells around their edges throughout the simulation. If the size of the maximally refined cells are very small, then there will be more of them and more computation time will be required to process the data along with more disk space to store it.
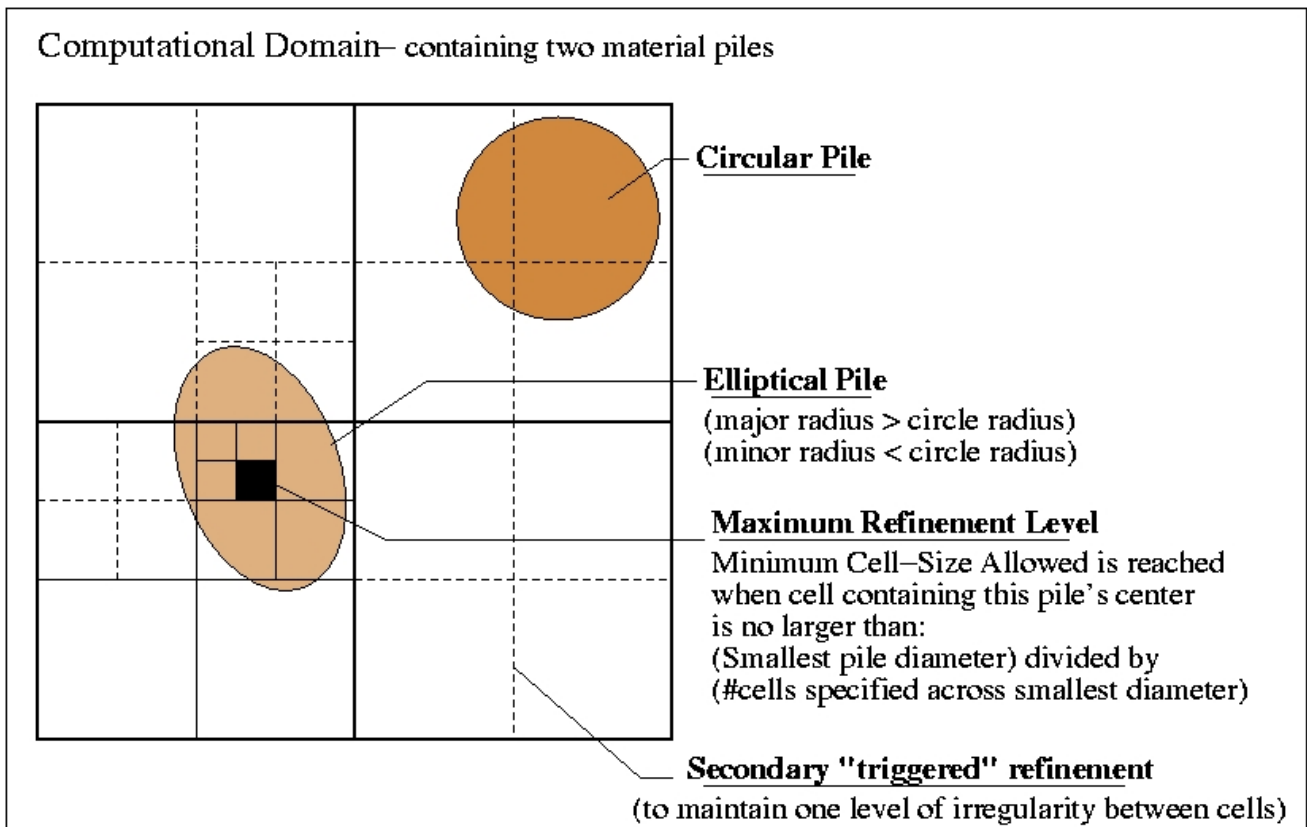
Figure 3.4: Illustration of maximum allowable refinement level as specified by "# of cells across smallest pile/flux-source diam." Here, the minor axis of the ellipse is the smallest pile diameter on the map and is therefore chosen as the diameter that is used in determination of the maximum level of refinement.

- The user must also specify the **Number of Piles** to include in a simulation (figure 3.3). The user may specify any number of piles in a single simulation. Each pile's attributes, such as its size, orientation and location, will be specified separately following the completion of this first form. If more than one pile is included in a simulation, they may be placed at any location on the map - even overlapped. If several piles are overlapped, then the height of material at that particular location will be defined as the largest of the component pile heights at that spot rather than summing them together.

- In addition to modeling the progression of an initially-specified fixed amount of material (i.e. debris pile) as it traverses down a sloped terrain, Titan2D now

17

has the capability to simulate material that actively extrudes from the ground. These sources of material are called flux-sources, any number of which can be implemented using the **<u>Number of Flux Sources</u>** box. For each flux-source specified, an additional window will later open (see figure 3.8), enabling the user to define its specific properties.

- The next data-entry field on the form is where the user specifies the **<u>Number of Discharge Planes</u>**. This feature has been added to give Titan the capability to calculate the amount of material that crosses any line (i.e. vertically-oriented plane) on the DEM map by specifying its two end-points. For each discharge-plane that is specified, another window is opened (see figure 3.9) for the user to enter the coordinates of these points. As the simulation progresses, the amount (cubic meters) of material crossing the plane is recorded in the simulation directory (.../bin/Test1 for example) under the file named *discharge.out*

  Note: The user may specify any number of discharge planes and may connect them in any fashion necessary to capture the volume of flow crossing a specific boundary. One suggestion is to form a box around a region on the map using 4 discharge planes. This will allow you to track the volume of flow entering or leaving that area regardless of its direction.

- TITAN2D allows for several properties of the simulation to be scaled. Clicking the **<u>Scale Simulation?</u>** button (button turns red when selected) allows the governing equations to be non-dimensionalized using pile height, gravity and length scaling factors (among others, such as velocity and time scales, that are secondarily derived from them). The pile height scale is taken as the cubed root of the total volume of material that will appear on the map - whether it is contained in fixed-mass piles, is actively extruded from flux-sources, or exists as some combination of the two. The gravity scaling factor is simply taken as 9.80 ms^{-2}. The length scale (unlike the pile height and gravity scales that are automatically computed by the code) must be specified by the user in the **<u>If Scaled, Length Scale [m]</u>** box. This scale factor refers to the expected runout length of the flows and need only be specified if the simulation is to be

scaled, however for all real-terrain calculations it is highly recommended that the simulation be scaled.

*Note: Scaling will help reduce the occurrence of round-off error and is critical for the proper control of propagating thin layers (i.e. the boundaries of a flow), which otherwise may have a tendency to develop unrealistically high velocities.

- The next two input parameters concern the run time of the simulation. The user must specify the **Maximum Number of Time Steps** (on the order of several thousand) and **Maximum Time** (in seconds). Fractions of seconds may be used if desired (e.g. 2.5 seconds). When the job is submitted for processing, the simulation will stop when it has reached the specified maximum allowed number of time steps or has simulated the specified amount of time (whichever comes first).

  * A Note on setting the maximum time and maximum number of time steps: Determining an acceptable stopping criteria for the simulation is difficult when considering general cases. Because of this, the stopping criteria for the simulation must be set by the user. The two criteria that are used are *Maximum Time* and *Maximum Number of Time Steps*. Both of these values should be set high enough such that the geologic event being simulated has ended (e.g. the material has come to rest). If either of these values are set too low, the simulation will end before the material has reached static equilibrium. If both of these values are set excessively high, wasted computation will be performed dynamically simulating a pile that has already come to rest. The simulation will end when either the number of time steps computed is equal to the *Maximum Number of Time Steps* or the simulation time has reached the *Maximum Time*. The number of computed time steps needed to simulate a geologic event will vary depending on the amount of computational mesh points used, the friction parameters, the use of grid adaptation, the simulation order, and initial pile geometry and location.

- In the **Time [sec] Between Results Output** box, the user can specify how often an output file is to be created. For example, the user may wish to generate an output file every 5 seconds of simulated time. The user may also

enter fractions of seconds if desired.

- The user can also specify the **Time [sec] Between Saves**. Titan now has the capability to restart from a saved run. This may be necessary if the maximum number of iterations has been reached or simulation time has expired or the computer simply crashes before you would like the simulation to end. Instead of running the entire simulation over again, you can continue from the point where a previous run has left off. The value entered in this field specifies the frequency at which the configuration data for the current run is saved to file. If run on a single processor, this data is alternately written to files named restart0000.0 and restart0000.1. To continue a previous run, the user must:

  (a) Find the most recently written restart0000.x file (or restart0000.x *and* restart0001.x files for a two processor run, and so on) by looking at the value of the "savefile" parameter in the output_summary.-00001 file. The extension of the most recently written file (i.e. extension 0 or 1) is specified by the last recorded value of "savefile".

  (b) The user then must rename the appropriate restart0000.x to restart0000.this and run Titan again as usual. (Noting that with two processors, the user must rename both restart0000.x and restart0001.x to a *.this* extension and run Titan again in its multi-processor mode)

  (Note: if the simulation that you would like to continue has ended because either the maximum number of iterations or runtime has been reached, you must first find and increase these values appropriately in the *simulation.data* file [to allow for an extended run] prior to restarting the run).

- The next box in this section is the **Adapt the Grid?** option. This refers to the computational grid and reduces the computational cost while maintaining the simulation's accuracy. However, this can also introduce some instability into the computation. Also note that if an adaptive grid is not selected, then the entire computational domain will be a uniform maximally refined grid with a cell-size determined by the user's specification in the *Number of Computational Cells Across Smallest Pile/Flux-Source Diameter* box above. Due to the

major savings in computation time, it is recommended that this be selected unless instability is detected in the output.

- The **Visualization Output** box allows the user to choose the formats of the output files (tecplotxxxx.plt, mshplotxxxx.plt, GMFG Viz, XDMF/Paraview, Web Viz, or grass_sites) - Activate the buttons corresponding to the visualization outputs that are desired. tecplotxxxx.plt and mshplotxxxx.plt are tecplot files [www.amtec.com]. GMFG Viz is for the TITAN2D visualizer. XDMF/Paraview generates xdmfxxxx.h5 and xdmfxxxx.xmf files used with the freely available *paraview* visualization application [http://www.paraview.org/HTML/Index.html]. Web Viz is for the Quickview Viewer output. The user can have multiple visualization output formats with each simulation run.

- **First/Second Order Method**: Clicking on the **Second** button for this option allows you to select the 2nd order method for calculating the values in a computational grid cell (the 1st order method is the default). Under the first order method, the values for pile height, momentum etc. that are calculated by the model are approximated as constant across the entire cell. This may mean that there is a jump up or down to the value of the same parameter in the neighboring cell. Under the 2nd order method, the values of the parameters are assumed to vary linearly across the cell. The 2nd order method takes into account the value of the neighboring cells and uses those values to calculate the value for the cell in question. For example, if the neighboring cells up-flow of the cell in question have values lower than the cells down-flow of the cell in question, the value of the cell is going to increase (have a slope) in the down-flow direction. If there is no difference in values between the neighboring cells, then the cell in question will also stay constant. Selecting the 2nd order method will produce slightly more accurate results, but may also increase the computation time as the code has to perform more calculations.

- The **Minimum (and Maximum) x and y location (UTM E, UTM N)** boxes can be used if a computational region smaller than the entire GIS region is desired, the user can input the minimum and maximum x and y location of the desired computational region in UTM coordinates.

*Note on entering coordinates: The user can look up the boundaries of a particular map by looking in it's corresponding header folder. For example, if using the DEM "ColimaSmall", the map boundaries are located in the file by that name in the following directory: /Colima/ColimaSmall/cellhd. (Coordinates are also required for specifying the locations of pile/flux-source centers and the end-points of discharge planes.)

- The **Height used to define flow outline** (> 0) **[m]** box sets the "boundary" of the mass flow. This was *necessary* in previous versions of Titan because, due to numerics, the flow height continued to thin with increasing distance from the pile centroid, taking on incredibly small numbers, without ever actually reaching "zero". In this version of Titan, although thin-layer flow is now controlled, the user is still able to define a pile boundary (i.e. ignore material less than a specified pile height) by entering a value greater than zero here. This will be used to, among other things, compute the spread of the pile in the x and y directions. Spread in each direction is defined as the maximum minus minimum coordinate where the pile height is greater than the value entered here. If no value is entered it defaults to $\frac{1}{50}$th of the maximum initial pile height. The x and y spread are 8th and 9th entries on each line of files named "statout_lhs.<##>" (where <##> is a 2 digit number). See the section on probabilistic simulations for more information.

- The **Test if flow reaches height [m] ...:** and **... at test point (x and y location):** boxes set the criteria to determine if flow reaches a particular point, namely did flow of this depth reach this point at any time during the calculation. The time (in seconds) that the flow reaches the point is the 3rd entry on each line of files named "statout_lhs.<##>" (where <##> is a 2 digit number). A value of -1 indicates that the flow did not reach the location during the simulated period of time, it is possible that not enough time was simulated. See the section on probabilistic simulations for more information.

- When the simulation has finished running, the user will be notified via the email address specified in the **Email Address** box. If an address is not set, notification will be sent to user@buffalo.edu. An email notification is not sent

if TITAN2D is being used on a LINUX PC.

- The **Run** button produces the starting files found within the simulation directory. When this button is selected, a new window appears (figure 3.5), in which the user specifies the two friction angles (see section below). The **Quit** button exits from the GUI, while the **?** button opens a new window displaying help files for this particular GUI.

3. <u>Pile, Flux-Source, and Material Property Parameters</u>: this user-input information is used to characterize the nature of the material in the simulation.

<u>Friction Angles - Data Entry Window</u>

- The next two parameters, the **Internal Friction Angle** and the **Bed Friction Angle** set the resistive frictional forces that occur within a material being simulated and between that material and the basal surface. The *internal friction angle* corresponds to friction arising from particle-particle interactions within the flowing material and is equivalent to the natural slope of the free-surface that would form if a cylindrical pile of the granular material were placed on a flat plane and allowed to collapse under its own weight. The *bed friction angle* corresponds to the friction that develops due to particle-ground interactions. This value is equivalent to the minimum slope that an inclined surface must obtain before the material placed on it begins to slide from its static position. Enter the appropriate values into the pop-up window that appears (see figure 3.5). When finished, click on "done" then "quit".

- When the **Done** button is selected, the friction angles data is stored in the simulation directory under the file named *frict.data* which will be used by Titan during a run. The **Quit** button closes this window.

*Note: A typical range of internal friction angles that occur in debris flows with a fluid volume fraction of up to 60 percent is: 25-45 deg. [1] The bed friction angle, however, is highly dependent on the character of the basal surface.
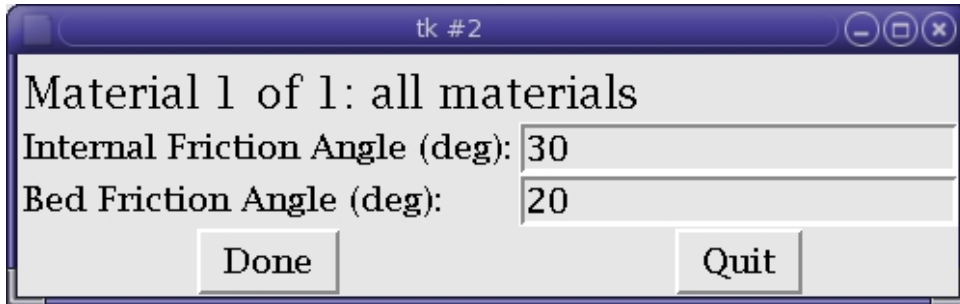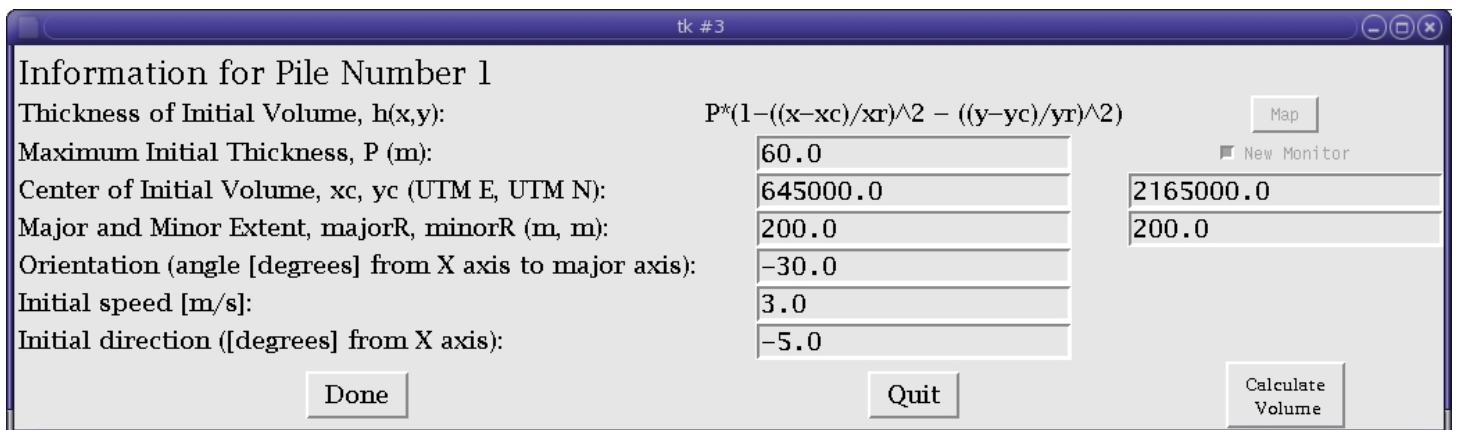
Figure 3.5: Pop-up window for entering friction angles

Figure 3.6: Pop-up window for entering initial pile conditions

---

[1] *Iverson*, Table 3, p.253, *The Physics of Debris Flows*, Reviews of Geophysics, 35, 3 / August 1997

<u>Pile Parameters - Data Entry Window</u>

- Figure 3.6 shows the pop-up window used for specifying the pile dimensions and locations. This window appears if a non-zero number of piles was specified in the main data-entry window. The first line in the window states the pile number for which the dimensions are being entered. In order for the user to specify an initial pile geometry that can vary, the pile has a defined shape of a paraboloid given by the equation $P * (1 - ((x - xc)/xr)^2 - ((y - yc)/yr)^2)$ (assuming an orientation angle of zero). The data to be entered by the user are: **Maximum Initial Thickness, P** (in meters), **Center of Initial Volume,xc,yc** (in UTM coordinates), the **Major and Minor Extent, majorR, minorR** of the initial pile (in meters), the **Orientation (angle [degrees] from X axis to major axis)**, and its **Initial speed [m/s]** and **Initial direction ([degrees] from X axis)**. Both angles are measured counterclockwise from the x-axis. The angle needs to be entered if you desire a non-zero value.

  (**Note:** The x-axis is defined as UTM E; y-axis is defined as UTM N).

  If two or more piles are being used and they overlap, the larger of the two pile heights will be taken as the pile height at the grid point. There are three buttons on the bottom of this window. When the **<u>Done</u>** button is selected, the pile parameters are entered into the GUI. The **Quit** button will close the pile dimensions window. If more than one pile needs to be specified, a new pile dimension window will appear when the *Quit* button is selected. Once the parameters for all the piles have been specified, then the data will be submitted. The **<u>Calculate Volume</u>** button (figure 3.6) will calculate the volume for the individual piles using the pile height and the X and Y extents. The volume is given in cubic meters and assumes there are no overlapping piles.

<u>Flux Sources - Data Entry Window</u>

- Similar to the pile information window, the flux source data entry window allows the user to specify certain parameters that will characterize the nature of one or more flux sources. Each flux source requires the following infor-
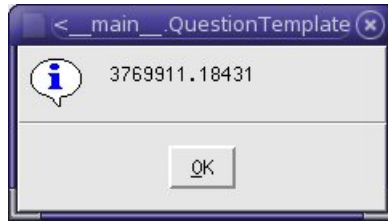
Figure 3.7:   Calculated Volume Pop-Up Window
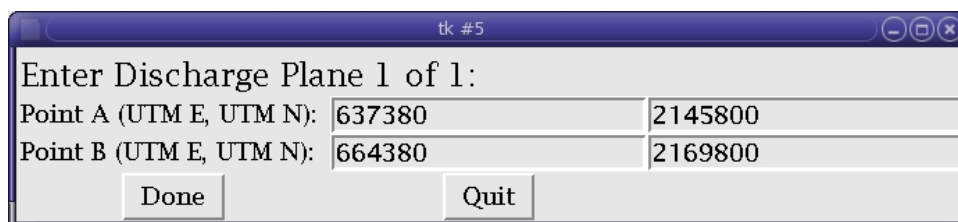
Figure 3.8:   Flux-Source Data Entry Window

mation: **Extrusion flux rate [m/s]** - the average rate at which material extrudes vertically from the ground (the material initially extrudes at twice the average flux rate, then decreases linearly to zero at the end of the flux source's duration), **Active Time [s], start, end** - where the user can specify a starting and ending time for the flux source that can encompass either part or all of the simulation, **Center of the source, xc, yc** - given in UTM coordinates, **Major and Minor Extent, majorR, minorR** - of the elliptically shaped flux source, **Orientation (angle [degrees] from X axis to major axis)**, **Initial speed [m/s]** - referring to the initial horizontal speed (i.e. tangential to the terrain) of the material as it leaves the flux-source (the flux-source itself, remains stationary), and finally the **Initial direction** of material - measured in degrees counter-clockwise from the x-axis (UTM E).

<u>Discharge Plane Coordinates - Data Entry Window</u>

- This window lets the user enter the coordinates of the two endpoints of a vertically-oriented discharge-plane. The user may enter the values or, if running Titan through GRASS, select the two points directly using the mouse cursor on the map. Each point is identified by a UTM E and UTM N coordinate. During the simulation Titan will calculate the amount (cubic meters) of material passing between these points and write the data in *discharge.out*

  *Note: Discharge planes have an orientation - meaning that flow passing through them in one direction may be recorded as having a positive volume, whereas flow passing through in the opposite direction will have a negative volume contribution. The orientation of discharge planes is such that if each point of 4 planes are successively laid out in a counter-clockwise manner (where the result is that the planes form a closed box), then any flow leaving the box will have a positive volume contribution. (That is, the material flux through a discharge plane obeys a right-hand rule sign convention.) See Figure 3.10 for further explanation.

Figure 3.9: Discharge Plane Data Entry Window
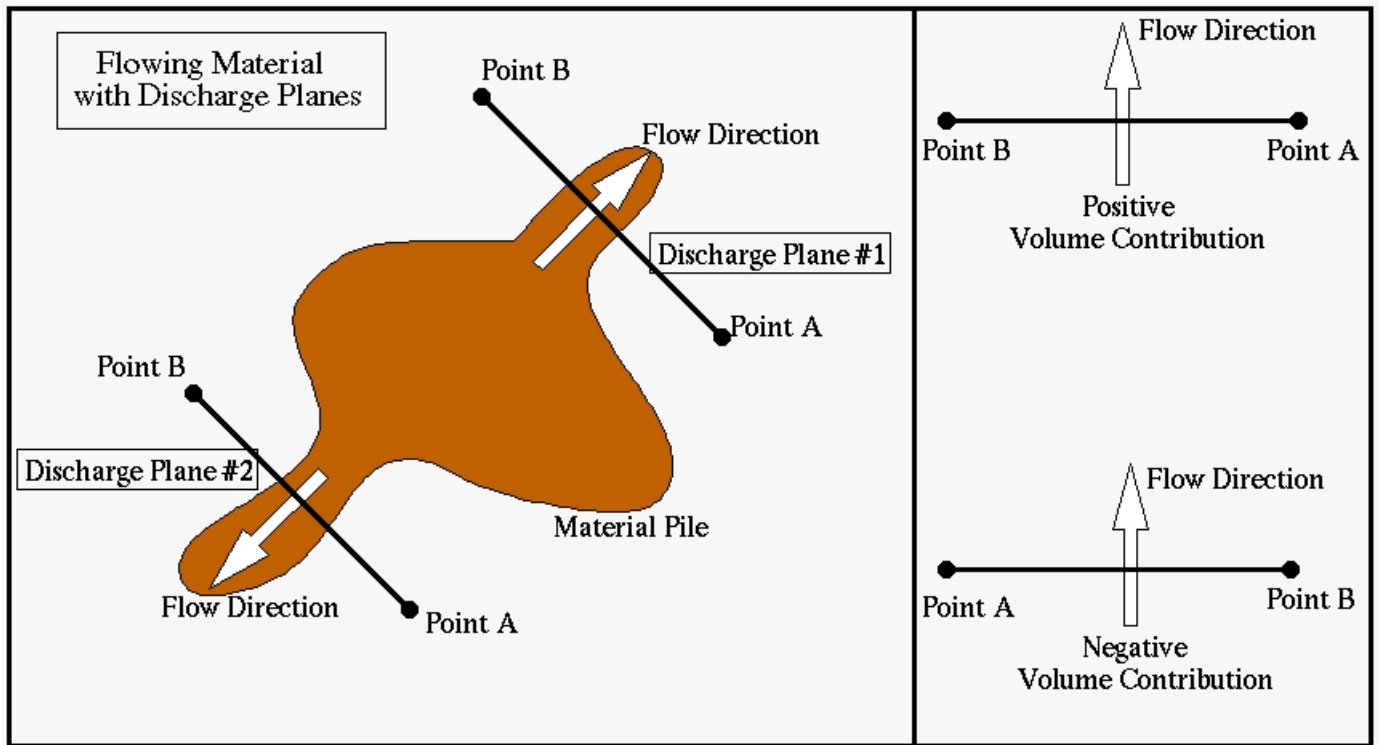
Figure 3.10: Discharge Planes and Orientation - this figure shows two discharge planes oriented in different directions (*with respect to the flow passing through them*). The volume contribution of a flow passing through a discharge plane follows a **right-hand rule** sign convention. As such, the flow passing through the first discharge plane [as shown] will have a positive (+) volume contribution. The second discharge plane is laid out in a manner such that the flow shown passing through it will have a negative (-) volume contribution.

3.4.2 Using TITAN2D GUI Through The GRASS GIS Interface

The TITAN2D GUI has the unique capability of running within a GRASS GIS environment. Within a GRASS environment the user is able to graphically access available DEMs, choose a specific region of interest within a DEM, and choose a starting point for the initial pile. GRASS can be operated by either interactive or command-line operations. This feature is available to the user as long as a copy of Grass5 is installed on your computer.

1. Before starting GRASS, open the .grassrc5 file in your root directory. It should

28

look something like this:

GISDBASE: /home/user/grass.data/grass5

LOCATION_NAME: DEM_location_ name

MAPSET: DEM name

GRASS_GUI: tcltk

If the text after GRASS_GUI: says "**text**", change it to "**tcltk**". This will allow you to access GIS files through a GUI rather than with text.

2. Start GRASS GIS at the prompt with the command: "**grass5**"

3. Specify the location of the GRASS mapset by clicking on the desired DEM file name in the pop-up window (figure 3.11).

4. Change your working directory to titan/bin

5. Ensure the following files are present in the titan/bin directory:

   (a) pilehelper

   (b) regionhelper

   (c) r.gmfg.titan2D

   (d) titan_gui.py

GRASS Command-Line Operation

1. Specify simulation parameters on the command line for ./r.gmfg.titan2D, to get a list of available options and parameters, type: `run r.gmfg.titan2D -h`

2. After the python GUI pops up (figure 3.12), input simulation parameters as usual.

3. Example of the model command line:

```
./r.gmfg.titan2D -a -s map=Tahoma30 dir=tempe mp=1 mesh=100 iang=30 bang=15
length=8000 maxts=1000 maxtime=100 outts=100 outfmt=tecplot, mshplot,
HDF piles=1 pileh=50
```
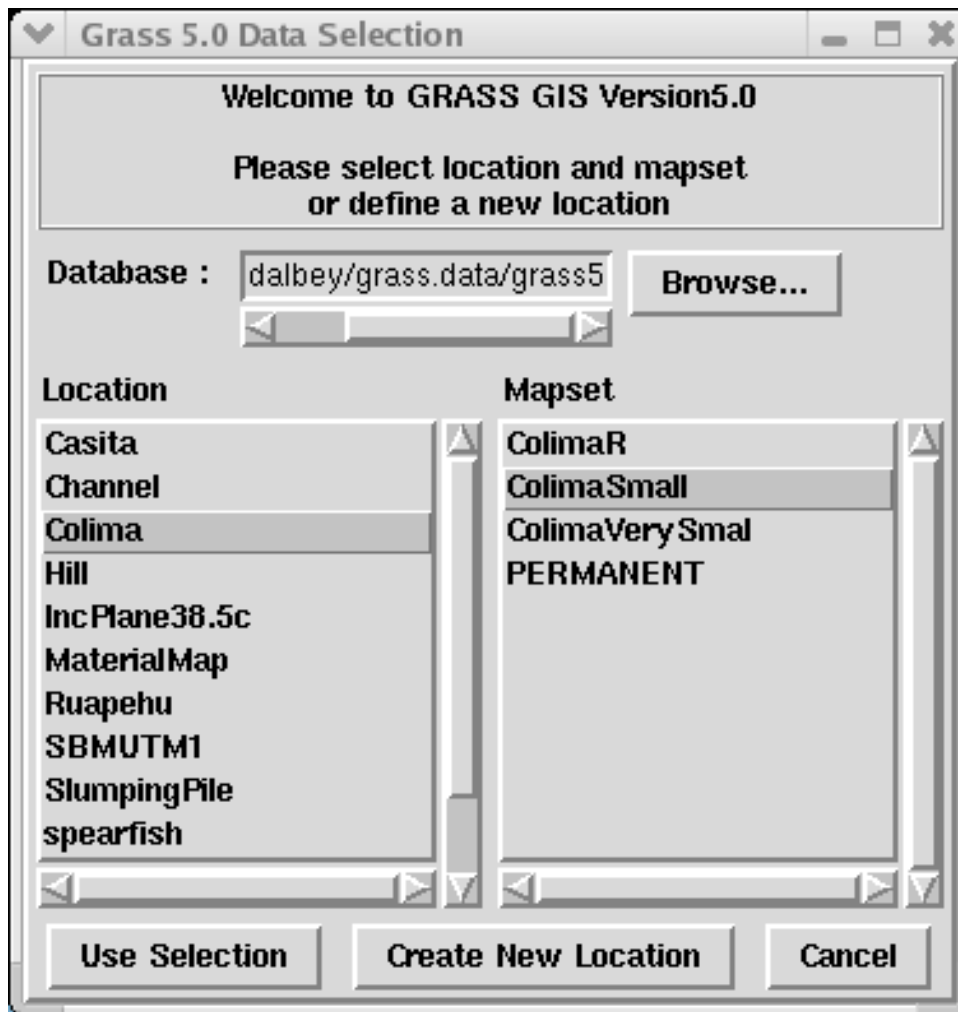
Figure 3.11: GRASS GIS map-set selector window

4. **Specifying the region for the simulation:**

- Start GRASS monitor either from GRASS GUI or using GRASS command:
  `d.mon start=x0`

- Display any map on the monitor with the command: "d.rast map_name"

- Specify the region interactively using command: "d.zoom"
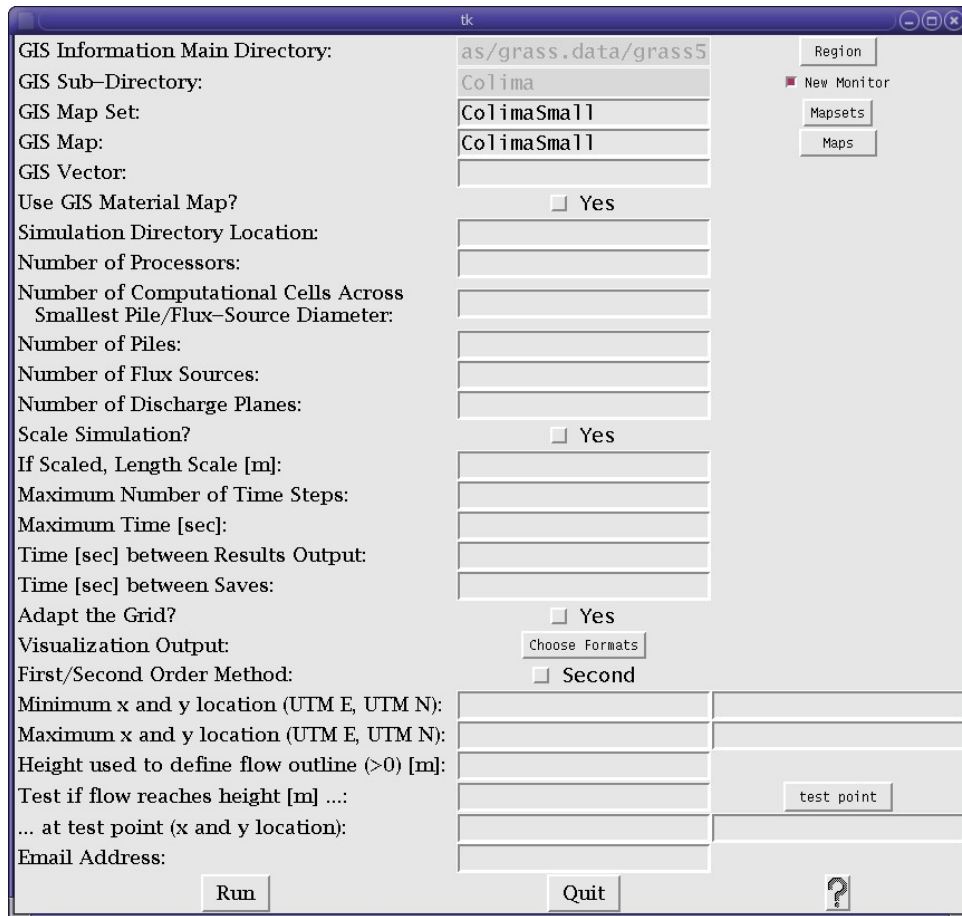
Figure 3.12: Python Interface with GRASS functions activated

5. **Interactive Operation of The Model:**

- Start the model from within GRASS using the following command:
  "**./r.gmfg.titan2D**"

  Optionally, you can specify any command line options for the model and they
  will be reflected in the GUI.

- Specify a mapset either by typing its name into the text field on the GUI or
  by clicking on the button <u>Mapsets</u> and selecting from the drop down menu.

- Specify a DEM either by typing its name into the text field on the GUI or by
  clicking on the button <u>Maps</u> and selecting from the drop down menu.

- Press the <u>**Region**</u> button on the on the GUI. You can untoggle the
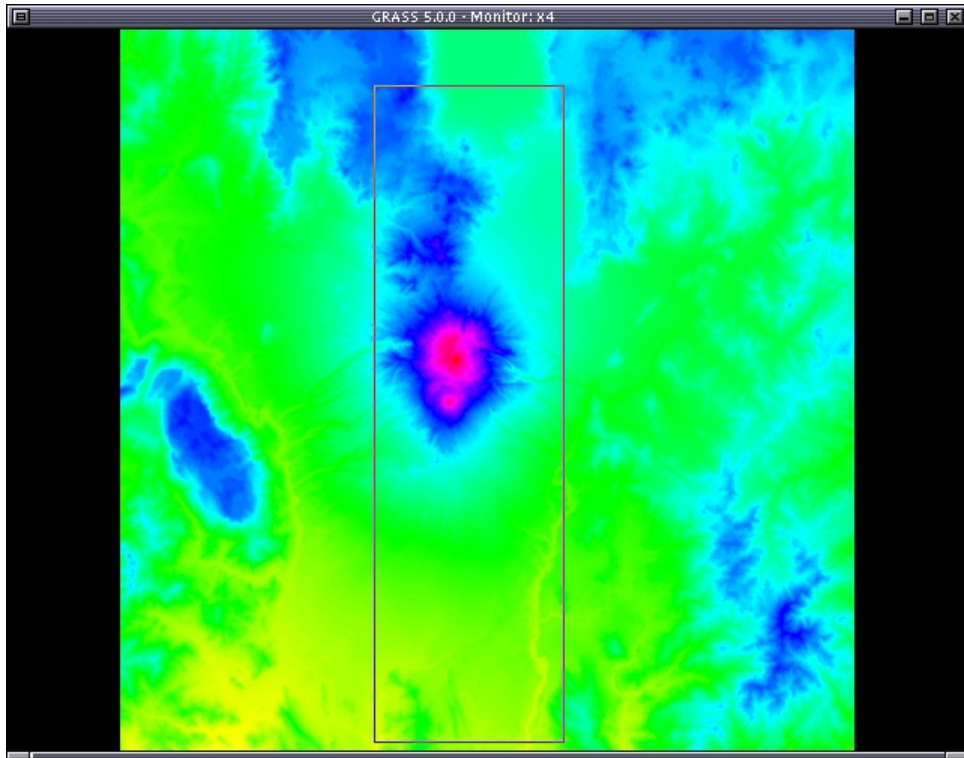
Figure 3.13: Original DEM

New Monitor button if you want to reuse your active GRASS monitor.
Specify the region of interest from the original DEM (figure 3.13). Figure 3.14
exhibits "zooming in"on a region of interest by using the following sequence
of mouse clicks to draw a box around the region:

1) **left button:** 1st region corner

2) **middle button:** 2nd region corner

3) **right button:** accept the region

- To specify the test point location interactively, press the **test point** button on
  lower right of the GUI. After a map screen appears, click on the best estimated
  test point location using the left mouse button. The coordinates of the point
  clicked will appear in the GUI.

- Enter the model parameters into the GUI (as explained in section 3) and press

32
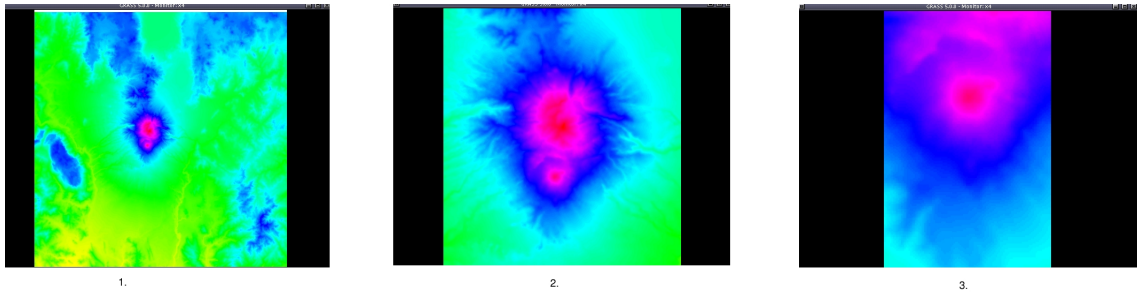
Figure 3.14: Zooming in on a desired region within a DEM

the **Run** button. After entering the friction angle data you will be presented with a pile information pop-up window (fig 3.6).

- To specify pile location interactively, press the **Map** button on the dialog. You can untoggle the **New Monitor** button if you want to reuse your active GRASS monitor. After a map screen appears, click on the best estimated pile location using the left mouse button. The coordinate of the point clicked will appear in the dialog box.

3.4.3 Running TITAN2D

After clicking the "done" and "quit" buttons found on the final information pop-up window, change your directory to the working simulation directory (for example: /titan/bin/Test1). At this point, you will need to start processing the job by typing:

- For single processor runs on a LINUX PC: "./titan"

- For all other cases: "mpirun -np X ./titan" (where X equals the number of processors used)

Runtime information will be printed to the screen at each iteration. This information consists of:

1. Time at the end of each timestep in hrs:min:sec

2. Volume of the flow

3. Maximum pile height [m]

33

4. Maximum velocity [m/s]

5. Average Velocity [m/s]

<u>Note:</u> **(Segmentation Violation)** - If you are running Titan and receiving a Segmentation Violation/Fault message, change your stack settings to allow unlimited stack size.

- If using csh/tcsh shells, type:
  > `limit stacksize unlimited`
  > `(Then run titan again as usual)`

- If using sh/bash/ksh, type:
  > `ulimit -s unlimited`
  > `(Then run titan again as usual)`

* Warning! If the flow has not stopped moving by the end of the simulation, TITAN2D may APPEAR to give unphysical results. The cause for this is that the size of the timesteps is dependent on the velocity. Large values for the bed and internal friction angles result in low velocities which translates to larger timesteps. This means that a simulation run with larger friction angles may have more time to flow and thus could travel further in the same number of iterations than an otherwise identical run with smaller friction angles. The run ending time should be checked. A warning is printed to the screen and to a file **(sim_end_warning.readme)** at the end of each run saying how much physical time was simulated and gives the maximum final velocity and its location in UTM coordinates. The **output_summary** file is used to show how far along in the simulation a given output is. The stopping criteria is based on time. This file relates simulated time to iteration number.

3.4.4   Probabilistic Mass Flow Simulations

What is LHS? LHS stands for Latin Hypercube Sampling. It is a constrained sampling method that can converge in far fewer samples than Monte Carlo. It works by dividing each random dimension into N equally probable bins. A sample point within each bin is randomly chosen. Each bin is divided into 2 equally probable parts, and a sample point is

generated in each of the new bins that doesn't already have one. This refinement should be repeated until the desired level of accuracy is obtained.

Titan2D has the capability to perform LHS simulations with uncertain bed friction (either normally or uniformly distributed) or uncertain volumes (uniformly distributed).

To perform a Titan2D LHS stochastic simulation follow the instructions below. Remember each Titan2D run in an LHS simulation run must run on a single processor (you must have entered "1" for the number of processors through the python gui "titan_gui.py".)

Step 1: Generating the LHS sample points
decide whether you want to perform a simulation with an uncertain/random/stochastic bed friction angle OR (not both) an uncertain/random/stochastic initial pile volume. If you choose the former type "./lhsbed" (no quotes) and answer the questions it asks you. If you choose the later type "./lhsvol" and answer the questions it asks you. These will produce a file named "stat_ctl.bed" or "stat_ctl.vol" respectively.

Step 2: Starting the LHS Simulation
Follow the instructions here in place of what you find in section 3.4.3 when running a stochastic simulation.
A) if submitting a batch job edit the pbs script "pbslhs" don't forget to pass the proper "stat_ctl." file (generated in Step 1) to the "dist-stats.pl" script. then type "qsub pbslhs" at the command prompt.
B) it is also possible to run the "dist-stats.pl" script from the command line on a single computer, however, this will take quite a long time. To launch an lhs simulation without using pbs, type "perl dist-stats.pl –ctlfile=stat_ctl.<extension>" at the command line. The perl script auto detects the number of CPU's (on computers running linux) and performs that many runs simultaneously.

Step 3: Computing Statistics

After the LHS simulation is completed type "./lhstitanstats" at the command line. This will produce a file named "statout.plot" which can be used to make convergence plots. There are 20 entries on each line, the first entry is the number of sample points used to generate the statistics on this line, the second is the estimated probability (0-1) that the flow reached the test height at the test point (which are entered through the python gui "titan_gui.py"), the rest are the mean, standard deviation, and skewness of the end state properties

a) volume averaged velocity

b) maximum height

c) x coordinate of the centroid

d) y coordinate of the centroid

e) x direction spread

f) y direction spread

The spread in each dimension is defined as the difference between the maximum and minimum coordinates at which the pile height is greater than or equal to the edge height. The edge height is set through the python gui "titan_gui.py".

Step 4: Plotting Convergence Studies of Statistics

We have provided a perl script "plot_stats.pl" that uses gnuplot to make convergence plots from the data in "statout.plot". Simply type perl plot_stats.pl from within the simulation directory to generate 3 ".ps" files suitable for printing on printers setup for UNIX, or viewing through programs such as ghostview or gimp. If you prefer not to use the "plot_stats.pl" script, the data file "statout.plot" (see Step 3) is straight forward enough that you can easily use any other plotting tool, such as a spreadsheet package or matlab, that you are familiar with.

Chapter 4

TITAN2D Viewers

4.1 *Paraview* visualization application

Paraview is an open-source data visualization software package. It is available from *(http://www.paraview.org/New/index.html)* for no cost. Titan can output data in Paraview readable e**X**tensible **D**ata **M**odel and **F**ormat (XDMF *http://www.arl.hpc.mil/ice/*). Although not required, Xdmf format utilizes HDF5 library to store actual data (computational grid, pile heights etc). We strongly recommend building titan with HDF5 support. HDF5 is a free-software, downloadable from HDF website *(http://hdf.ncsa.uiuc.edu/HDF5/)*.

When selected, titan will generate data files named *xdmf*.xmf* and *xdmf*.h5*, if titan was compiled with HDF5 support. Otherwise only *xdmf*.xmf* files will be created, all the data being in ASCII format. These files can take considerably larger disc space in comparison to HDF5 files. We have tested only Paraview-2.6 for titan. But we don't foresee any problem with other versions. Following are some quick steps to:

View titan output in Paraview:

1. Open Paraview using paraview command on linux

2. Open xdmfxxxxxxx.xmf file, using **Open Data** in **File** menu

3. Click on **Accept** button on the left-hand panel (Map should appear in the main

window)

4. Adjust view using mouse buttons

   - hold down left–button and move mouse to rotate the view

   - hold down middle–button and move mouse to pan the view

   - hold down right–button and move mouse to zoom-in or zoom-out

5. Change to **Display** tab on left-panel

6. Select or deselect flow properties in the color section

7. You can change the contour levels using **Edit Color Map**

Create animations:

1. Open xdmfxxxxx.xmf file representing $1^{s}t$ time-step using steps 1-7

2. Add files for subsequent time-steps using **Timesteps** under **Parameters** tab

3. From **View** menu, select **Keyframe Animation**, left panel will start showing animation controls

4. Change no. of frames to no. of files added in step 2

5. Select **Source** to 1st filename (e.g. xdmf0000000000.xmf)

6. Click **Add Frame** once, select **Value** as 1st filename

7. Click **Add Frame** again and select **Value** as last filename

8. Adjust time to simulation time

9. Click on VCR style play ($\triangleright$) button to view animation

10. To save animation click on right-most button on VCR controls (the one with film symbol on it)

4.2   GMFG Viewer for Linux

To install GMFG viewer on your system, download Install_GMFGViewer.tar.gz and uncompress it with tar -xzvf Install_GMFGViewer.tar.gz Change into the Install_GMFGViewer directory and follow the instructions.

1. Run the installViewer.sh script

   ./installViewer.sh <name of GMFG Viewer tarball>

2. This script will first check if Qt (a free graphical user toolkit) is installed on your system and then proceed to install all the software needed to compile GMFGViewer on your system.

3. This script will also compile GMFGViewer and the executable file will be placed in GMFGViewer/bin folder.

4. Copy the executable file to the folder with *.bin files and python_input.data file generated by Titan2D.

5. To run the viewer, type: ./GMFGViewer

6. Enter the number of processors (same as the number of *.bin files).

7. Enter degree of resolution of the mesh to be displayed.
   1 → very fine resolution (GIS resolution; need large RAM or fast graphics card)
   2 → medium resolution
   3 → coarse resolution

8. The viewer will look for a texture image with the name "GIS_Texture.rgb" in the same folder as the *.bin files and apply it to the terrain. If not found, the terrain will appear plain white.

9. A viewer such as that shown in 4.1 will appear.

10. When the viewer is activated, by default it starts with sequential display of color-coded pile height contours of flow regions of all time steps. There are two radio buttons and four push buttons in the top panel of the viewer (see Fig. 4.2)
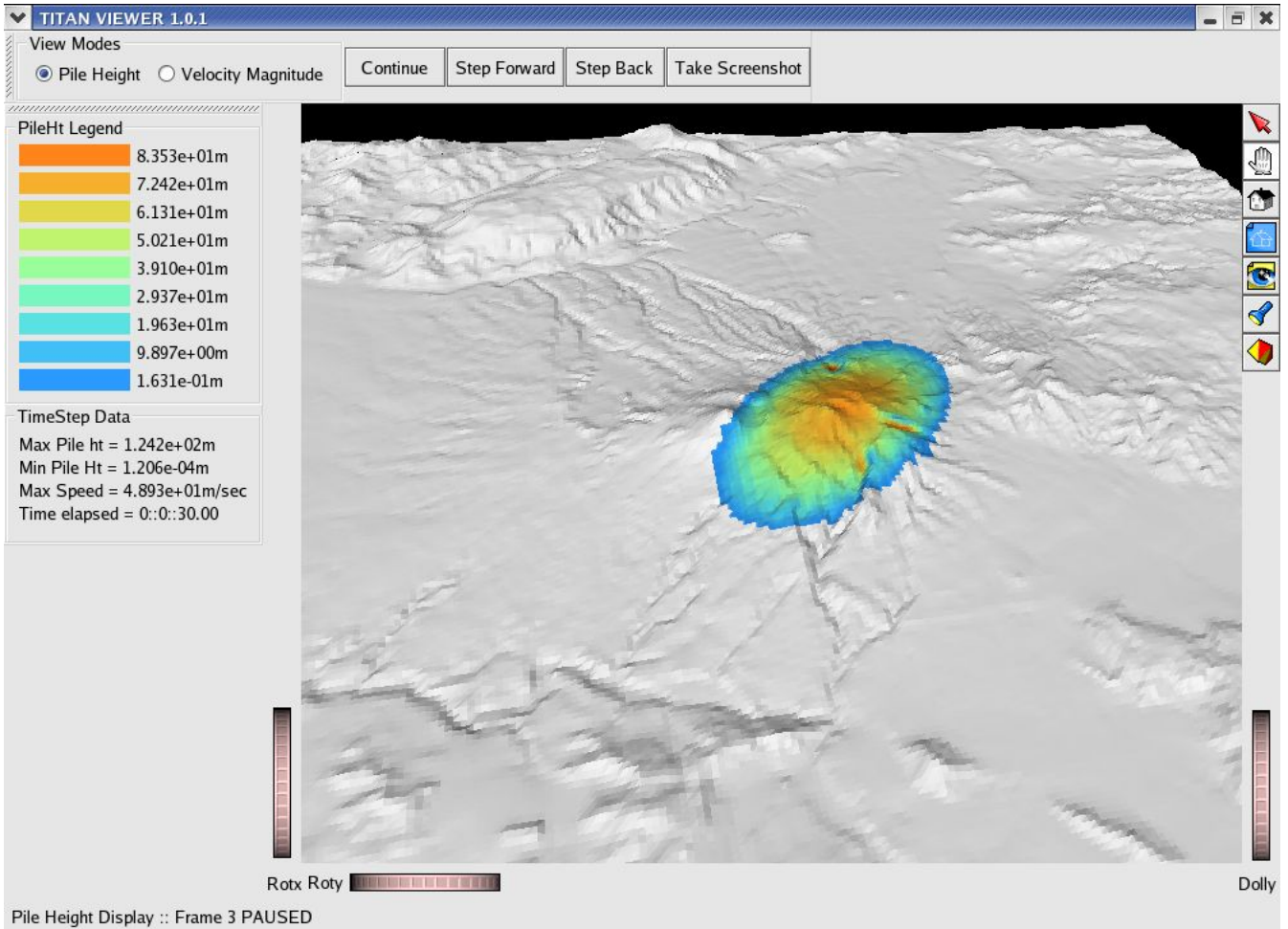
Figure 4.1: Viewer screenshot

Figure 4.2: Viewer screenshot2

Figure 4.3: Viewer screenshot3

- The pair of radio buttons allows the user to toggle between pile height contours and velocity magnitude contours.

- The first push button allows the user to pause and resume the animation. When the "Pause" button is hit, the remaining three push buttons get activated (4.2) and the user can step through the paused animation. Clicking the "Step Forward" button repeatedly, advances the animation by one time step and loops back to the first time step when the last one is reached. "Step Back" button behaves similarly but in the reverse direction. Hitting the first push button again will resume the animation.

- When the animation is paused, a snapshot of the flow can be taken by pushing the "Take Screenshot" button. This saves the currently displayed time step as a .ps image in the current folder.

- In the paused state, information related to the current time step such as maximum pile height, maximum velocity reached and time elapsed is displayed. Legends depicting absolute values of pile heights and velocity magnitudes associated with the colors are also displayed. For the pile height legend, values equal to or greater than 1/512th of the initial pile height are color-coded and displayed.

11. The three dials on the bottom edges of the viewer are provided for easy navigation (rotation, zooming) through the data.

12. The buttons on the right edge of the viewer are (from top to bottom) (see Fig. 4.3)

    (a) To allow the user to pick geometry (disabled)

(b) Track ball to rotate and move through the geometry.

(c) Reset to home viewing position.

(d) Setting the home viewing position.

(e) View all - View all the geometry in the data.

(f) Cross-hair - To zoom into a selected region in the terrain. Click this button and click the area in the window that needs to be magnified.

(g) Toggle between orthographic and perspective viewing.

13. Keyboard shortcuts for navigating through the data:

(a) hold down left mousebutton and move mouse pointer to rotate the camera around it's current focal point

(b) hold middle mousebutton to pan (or a CTRL-key plus left mousebutton, or a SHIFT-key plus left mousebutton)

(c) hold down left + middle mousebutton to zoom / dolly, or CTRL + middle mousebutton, or CTRL + SHIFT + the left mousebutton

Chapter 5

Quick Reference Guide

* **PLEASE NOTE:** This Quick Reference Guide is a copy of the **README** file available in your downloaded copy of TITAN2D. For the most recent updates and changes to TITAN2D please consult the **README** file.

- The code currently simulates granular flows. It runs in parallel with mesh refinement and unrefinement. The maximum refinement level is set by the user when specifying the *Number of Computational Cells Across Smallest Pile/Flux-Source.* Mesh repartitioning is also used to maintain a good load-balance. A python script is included to organize the preprocessing and launching of jobs on different computers, including all available computers at the Center for Computational Research (CCR) at UB. GRASS is now used as the GIS. The python script can also be run directly from GRASS. Instructions for this are in the README.GRASS file.

- The python script is located in the bin directory and is run by the command: "**python titan_gui.py**

- A GUI will come up and ask for certain info. There are several pop-up windows that will open, depending on the information you provide. The information that the GUI asks for is:

Main Data Entry Window

- **GIS Information Main Directory** – The main directory where the GIS information is stored. The *Region* button can be used to view elevation contours for the region specified if the GRASS interface is used. All GIS input data must be in GRASS format to be used.

- **GIS Sub-Directory** – The sub-directory where the GIS information is stored.

- **GIS Map Set** – The name of the GIS map set. The *Mapsets* button can be used to find the desired mapset through the GRASS interface.

- **GIS Map** – The name of the GIS map. The *Maps* button can be used to find the desired map through the GRASS interface.

- **Use GIS Material Map?** – *NOTE: The next feature is available only to users who have a specific material map (file ends in _Mat)*. The *Use GIS Material Map?* check box enables the input of a GIS-based surficial material map that matches the area covered by the DEM. This map is used to define the zones in the region where changes in the surface morphology results in a change in the basal friction angle. When this function is enabled, pop-up windows (appearing after the "run" button is clicked) will ask for the internal and basal friction angles for each material represented on the material map.

- **Simulation Directory Location** – The location from where the job will be submitted. All of the information (except for the GIS information) needed to run the simulation will be stored in this directory. If this specified directory already exists, then the pre-existing directory will not be modified and no job will be submitted.

- **Number of Processors** – The number of processors that will be used during the simulation. The code only allows a power of 2 (i.e. 2^n) amounts of processors and the amount of processors must be less than or equal to 2056.

- **Number of Computational Cells Across Smallest Pile/Flux-Source Diameter** – The number entered in this box is used by Titan to determine the

44

maximum level of grid-cell refinement allowed throughout the simulation. The default value is 20, meaning that the dimensions of the smallest cell size on the map will correspond to a length that is at most 1/20th that of the smallest pile or flux-source diameter - that is, 1/20th of the smallest value entered in the *"Major and Minor Extent, majorR, minorR (m,m)"* field on the Pile or Flux-Source Information Forms. When an adaptive grid is selected, only a small portion of the total map area will be maximally refined; without adaptation however, the entire map will consist of maximally refined cells - which, if small, can be computationally expensive.

- **Number of Piles** – Insert a value corresponding to the number of fixed-mass debris piles that you would like to simulate at a given time. Pile attributes, such as dimensions, orientation and location are specified when this first stage is completed.

- **Number of Flux Sources** – Indicate the number of locations where you would like to simulate material that actively extrudes from the ground. For each flux-source specified, an additional window will later open, enabling the user to define specific properties of each source.

- **Number of Discharge Planes** – This feature has been added to give Titan the capability to calculate the amount of material that crosses any line (i.e. vertically-oriented plane) on the DEM map by specifying its two end-points. For each discharge-plane that is specified, another window is opened for the user to enter the coordinates of these points. As the simulation progresses, the amount (cubic meters) of material passing through the plane is recorded in the simulation directory (.../bin/Test1 for example) under the file named *discharge.out*

- **Scale Simulation ?** – Activate the "Yes" button to scale the governing equations by the pile height, a length scale and a gravity scale (activated button appears as a red color). The pile height scale is taken as the cubed root of the total volume of material to appear on the map. Thus, the scaled simulation calculates the pile height as a fraction of this value. The gravity scale is 9.80 m s^{-2}. The length scale is a user-specified number.

- **If Scaled, Length Scale [m]** – A scale that will usually correspond to the

expected runout length of the flow. This is only used if the simulation is scaled.

- **Maximum Number of Time Steps** – A maximum amount of time steps that the simulation should run. For most simulations, this should be in the 1,000s range.

- **Maximum Time [sec]** – The maximum amount of time that the simulation will approximate.

- **Time [sec] Between Results Output** – This corresponds to how often results will be saved to file for later analysis. These files can become very large and the user may not need to see results for every time step since some time steps may have little change in the results from the previous time step. For example, a user may wish to only save a timestep every 5 seconds of simulated time. If desired, the user may choose to use a fraction of a second.

- **Time [sec] Between Saves** – specify the frequency at which the configuration of the simulation is saved to a restart0000.x file. To continue a given simulation, rename the most recent restart0000.x file to restart0000.this and run titan again as usual. (Note that additional restart0001.x, restart0002.x, etc. files will exist - and will also have to be renamed - for multi-processor runs)

- **Adapt the Grid ?** – Activate the "Yes" button to adapt the grid during the simulation (activated button appears as a red color). Adapting the grid should result in reduced computational cost while maintaining a high simulation accuracy, but can also introduce instabilities into the computation.

- **Visualization Output** – Choose Formats (tecplotxxxx.plt, mshplotxxxx.plt, GMFG Viz, XDMF/Paraview, Web Viz, or grass_sites) - Activate the buttons corresponding to the visualization outputs that are desired. tecplotxxxx.plt and mshplotxxxx.plt are tecplot files. GMFG Viz is for the TITAN2D visualizer. XDMF/Paraview generates xdmfxxxx.h5 and xdmfxxxx.xmf files used with the *paraview* visualization application. Web Viz is for the web visualization output. Note that the user can have multiple visualization output formats with each simulation run.

46

- **First/Second Order Method** – Activate the <u>**Second**</u> button to assume a linear variation (instead of constant values) of the conserved quantities (pile height, momentum) across each cell. The second order method should give more accurate results but increases computation time.

- **Minimum x and y location (UTM E, UTM N)** – If a computational region that is smaller than the GIS region is desired, the user can input the minimum x and y location of the desired computational region.

- **Maximum x and y location (UTM E, UTM N)** – If a computational region that is smaller than the GIS region is desired, the user can input the maximum x and y location of the desired computational region.

- **Height used to define flow outline** (> 0) **[m]** – Used to set the "boundary" of the mass flow. The user is required to enter a greater than zero height, which will be used to, among other things, compute the spread of the pile in the x and y directions. If no value is entered it defaults to $\frac{1}{50}$th of the maximum initial pile height.

- **Test if flow reaches height [m] ...:** and **... at test point (x and y location):** – Set the criteria to determine if the flow reaches a particular point, namely did flow of this depth reach this point at any time during the calculation.

- <u>**Email Address**</u> – The email address to send the notification of completion for the run to. Email will be sent to user@buffalo.edu if not set. An email address is not needed if you are doing the run on a PC.

- <u>**RUN**</u> button – Sets up and runs the simulation.

- <u>**QUIT**</u> button – exits the script.

- <u>**?**</u> button – Help button which displays the README file.

<u>Friction Angles - Data Entry Window</u>

- **Internal Friction Angle (deg), Bed Friction Angle (deg)** – These two friction angles must be entered into the pop-up window that appears after clicking the "run" button. Both angles are to be input in degrees. They represent the internal resistive forces that occur within a pile and the resistive force between a pile and the basal surface respectively.

<u>Pile Parameters - Data Entry Window</u>

- After entering the two friction angles, and clicking on "done" then "quit", a new window will appear to input the geometry and coordinates of each pile. The first line will show which pile number the user is inputting data for. The geometry for each pile is a paraboloid given by: $P * (1 - ((x - xc)/xr)^2 - ((y - yc)/yr)^2)$ (assuming an orientation angle of zero). The data to be entered is: **Maximum Initial Thickness, P** (in meters), **Center of Initial Volume,xc,yc** (in UTM coordinates), the **Major and Minor Extent, majorR, minorR** of the initial pile (in meters), the **Orientation (angle [degrees] from X axis to major axis)**, and its **Initial speed [m/s]** and **Initial direction ([degrees] from X axis)**. (**Note:** The x-axis is defined as UTM E; y-axis is defined as UTM N). In order to allow anyone to easily specify an initial pile geometry that can vary, the pile is assumed to have a shape of a paraboloid. The equation for the pile height is listed (a negative pile height calculated from this equation will be set to zero). These equations are for the non-scaled pile. If two or more piles overlap, the highest height of the pile is used as the pile height at that point.

The buttons in this window are:

- **DONE** button - After the values for the pile geometry are entered, hit the *Done* button to enter them into the script.

- **QUIT** button - After hitting the *Done* button, hit the *Quit* button to exit out of this window. If there are more pile geometries to input, another window will pop

up. Otherwise, the script will run the job.

- **Calculate Volume** button - From the Maximum Initial Thickness and X and Y Extent of the initial volume, this will calculate and display the actual volume (in cubic meters) resulting from the inputted values. This volume is only for this pile geometry and assumes that no other piles are overlapping this region.

- **Map** button – If you are using the Grass interface, the pile center can be specified from the elevation contour map by using the Map button.

- In order to figure out the proper values to input for the pile equation, run the simulation once for a couple of time steps in order to get some results files. Examine the results files for the proper location and values for the pile height equation.

Flux Sources - Data Entry Window

- The following information is required for each flux source that is specified: **Extrusion flux rate** – the rate at which material extrudes vertically from the ground, **Active Time** – the user can specify a starting and ending time for the flux source that can encompass either part or all of the simulation, **Center of Source, xc, yc** – given in UTM coordinates, **Major and Minor Extent, majorR, minorR** – required because the flux source is elliptically shaped, **Orientation (angle [degrees] from X axis to major axis)**, **Initial Speed** – referring to the initial horizontal speed (i.e. tangential to the terrain) of the material as it leaves the flux-source (the flux-source itself, remains stationary), and the **Initial Direction** of material – measured in degrees from the x-axis.

Discharge Planes - Data Entry Window

- This window lets the user enter the coordinates of the two endpoints of a vertically-oriented discharge-plane. The user may enter or, if running Titan through GRASS, select the two points directly using the mouse cursor on the map. Each point is identified by a UTM E and UTM N coordinate.

Notes on Python Software

- The latest version of Python can be downloaded for free from:

  http://www.python.org/download/

Appendix A

A.1 Governing *"Shallow Water"* Equations

The shallow-water model conservation equations[1] solved by Titan are given as:

$$\frac{\partial}{\partial t}\big(\vec{U}\big) + \frac{\partial}{\partial x}\big(\vec{F}(\vec{U})\big) + \frac{\partial}{\partial y}\big(\vec{G}(\vec{U})\big) = \vec{S}(\vec{U})$$

where:

$$\vec{U} = \begin{bmatrix} h \\ hV_x \\ hV_y \end{bmatrix}$$

is the vector of conserved state variables

(with h = flow depth, hV_x = x-momentum, hV_y = y-momentum).

$$\vec{F}(\vec{U}) = \begin{bmatrix} hV_x \\ hV_x^2 + \frac{1}{2}k_{ap}g_z h^2 \\ hV_x V_y \end{bmatrix}$$

is the mass and momentum fluxes in the x-direction

(with hV_x = mass flux in x-direction, $hV_x^2 + \frac{1}{2}k_{ap}g_z h^2$ = x-momentum flux in x-direction, $hV_x V_y$ = y-momentum flux in x-direction).

---

[1]For complete derivation, see: Denlinger, R.P., and Iverson, R.M. (2004), *Granular avalanches across irregular three-dimensional terrain: 1. Theory and computation*, J. Geophys. Res., 109, F01014, doi:10.1029/2003JF000085.

$$\vec{G}(\vec{U}) = \begin{bmatrix} hV_y \\ hV_xV_y \\ hV_y^2 + \frac{1}{2}k_{ap}g_zh^2 \end{bmatrix}$$

is the mass and momentum fluxes in the y-direction

(with hV_y = mass flux in y-direction, hV_xV_y = x-momentum flux in y-direction, $hV_y^2 + \frac{1}{2}k_{ap}g_zh^2$ = y-momentum flux in y-direction).

$$\vec{S}(\vec{U}) = \begin{bmatrix} 0 \\ g_xh - hk_{ap}\text{sign}\left(\frac{\partial V_x}{\partial y}\right)\frac{\partial}{\partial y}(g_zh)\sin\phi_{int} - \frac{V_x}{\sqrt{V_x^2+V_y^2}}\max\left(g_z + \frac{V_x^2}{r_x},0\right)h\tan\phi_{bed} \\ g_yh - hk_{ap}\text{sign}\left(\frac{\partial V_y}{\partial x}\right)\frac{\partial}{\partial x}(g_zh)\sin\phi_{int} - \frac{V_y}{\sqrt{V_x^2+V_y^2}}\max\left(g_z + \frac{V_y^2}{r_y},0\right)h\tan\phi_{bed} \end{bmatrix}$$

is the vector of driving and dissipative source terms

(with g_xh = driving gravitational force in x-direction, $-hk_{ap}\text{sign}\left(\frac{\partial V_x}{\partial y}\right)\frac{\partial}{\partial y}(g_zh)\sin\phi_{int}$ = dissipative internal frictional force in x-direction, $-\frac{V_x}{\sqrt{V_x^2+V_y^2}}\max\left(g_z + \frac{V_x^2}{r_x},0\right)h\tan\phi_{bed}$ = dissipative basal frictional force in x-direction; similar terms for y-direction).

Note Also:

k_{ap} = active/passive lateral stress coefficient term, where "active" k_{ap} assumes a smaller value in a diverging flow, while k_{ap} "passive" takes on a larger value and means that the flow is converging.

A.2   GIS Header File

The header file contains information about the projection, elevation data bounding box coordinates, number of columns and rows, and resolution. Additional information are stored to allow the correct interpretation of the binary data file. For a GIS Map defined in the TITAN2D GUI, the corresponding header file is named GIS Map and should be located at: GIS Information Main Directory/GIS Sub-Directory/GIS Map Set/cellhd/GIS Map. This information can be used with the GIS of your choosing,

including GRASS.

The header-file is an ASCII file, with the following lines:

proj: projection code
zone: UTM projection zone
north: upper Y-direction coordinate
south: lower Y-direction coordinate
east: right X-direction coordinate
west: left X-direction coordinate
cols: number of columns
rows: number of rows
e-w resol: resolution in X-direction
n-s resol: resolution in Y-direction
format: binary data format
compressed: compression flag

Projection code and the UTM projection zone are not used, therefore one can use any value, such as 1 for projection code (in GRASS, 1 corresponds to UTM projection). The UTM projection zone can also be any number. If desired, the correct UTM zone number should be used in UTM projection zone.

The number of columns multiplied by resolution in X-direction must be equal to the difference between right X-direction coordinate and left X-direction coordinate. The number of rows multiplied by resolution in Y-direction must be equal to the difference between the upper Y-direction coordinate and the lower Y-direction coordinate.

Binary data format must be -1 indicating that the data are IEEE float values. Compression flag must be 1 if is compressed, 0 otherwise.
More information can be found in the GRASS 5.0 Programmer's Manual: **http://mpa.itc.it/ markus/grass50progman/node37.html**

Example header file:

proj: 1

zone: 13

north: 2210030

south: 2109950

east: 700050

west: 599970

cols: 1667

rows: 1667

e-w resol: 60.0359928

n-s resol: 60.0359928

format: -1

compressed: 1

A.3   GIS Data File

Elevation data is stored in the data file, a binary file stored at the fcell directory. For a GIS Map defined in the TITAN2D GUI, the corresponding data file is named GIS Map and should be located at: GIS Information Main Directory/GIS Sub-Directory/GIS Map Set/fcell/GIS Map The data in the file are IEEE float values, stored in Big-Endian byte order. Data can be compressed using zlib. In a uncompressed file, elevation values are stored sequentially from the first column of the first row to the last column of the last row.

An example uncompressed file for a 4 row by 5 column data would be:

row[0]column[0] row[0]column[1] row[0]column[2] row[0]column[3] row[0]column[4]

row[1]column[0] row[1]column[1] row[1]column[2] row[1]column[3] row[1]column[4]

row[2]column[0] row[2]column[2] row[2]column[2] row[2]column[3] row[2]column[4]

row[3]column[0] row[3]column[3] row[3]column[2] row[3]column[3] row[3]column[4]

row[ ]column[ ] are float values in IEEE format:

If the file is compressed, the first byte of the file indicates the number of bytes used for each elevation data and must be 4 (0x04). The next four bytes indicates the initial position of the first row of data, and the sequence continues with the initial position of each row for all rows. A compressed row is flagged by the first byte, which must be 0x31. The difference between the row to be read and the next row is used to read the set of compressed values, that are compressed using zlib compress function. The uncompressed data will correspond to the elevation float values.

Further information can be found at:
http://mpa.itc.it/markus/grass50progman/node35.html
http://www.gzip.org/zlib

A.4 GIS Material Header File

The header file contains information about the projection, material data bounding box coordinates, number of columns and rows, and resolution. Note that this information should be the same as the elevation data header file. Additional information is stored to allow the correct interpretation of the binary data file. For a GISMap defined in the TITAN2D GUI, the corresponding header file is named GISMap_Mat and should be located at: GIS Information Main Directory/GIS Sub-Directory/GIS Map Set/cellhd/. This information can be used with the GIS of your choosing, including GRASS. The header-file is an ASCII file, with the following lines:

    proj: projection code
zone: UTM projection zone
north: upper Y-direction coordinate
south: lower Y-direction coordinate
east: right X-direction coordinate

west: left X-direction coordinate

cols: number of columns

rows: number of rows

e-w resol: resolution in X-direction

n-s resol: resolution in Y-direction

format: binary data format

compressed: compression flag

Projection code and the UTM projection zone are not used, therefore one can use any value, such as 1 for projection code (in GRASS, 1 corresponds to UTM projection). The UTM projection zone can also be any number. If desired, the correct UTM zone number should be used in UTM projection zone.

The number of columns multiplied by resolution in X-direction must be equal to the difference between right X-direction coordinate and left X-direction coordinate. The number of rows multiplied by resolution in Y-direction must be equal to the difference between the upper Y-direction coordinate and the lower Y-direction coordinate. Binary data format must be 0 indicating that the data are byte (8 bits) values. Compression flag must be 1 if is compressed, 0 otherwise.

More information can be found in the GRASS 5.0 Programmers Manual: **http://mpa.itc.it/ markus/grass50progman/node37.html**

Example header file:

    proj: 1

zone: 13

north: 2210030

south: 2109950

east: 700050

west: 599970

cols: 1667

rows: 1667

e-w resol: 60.0359928

n-s resol: 60.0359928

format: 0

compressed: 1

A.4.1 GIS Material File

Material data is stored in the data file, a binary file stored at the cell directory. For a GISMap defined in the TITAN2D GUI, the corresponding material data file is named GISMap_Mat and should be located at: GIS Information Main Directory/GIS Sub-Directory/GIS Map Set/cell/GISMap_Mat. The data in the file are byte values (8 bits). Data can be compressed using zlib. In a uncompressed file, material index values are stored sequentially from the first column of the first row to the last column of the last row. An example uncompressed file for a 4 row by 5 column data would be:

row[0]column[0] row[0]column[1] row[0]column[2] row[0]column[3] row[0]column[4] row[1]column[0] row[1]column[1] row[1]column[2] row[1]column[3] row[1]column[4] row[2]column[0] row[2]column[2] row[2]column[2] row[2]column[3] row[2]column[4] row[3]column[0] row[3]column[3] row[3]column[2] row[3]column[3] row[3]column[4]

row[ ]column[ ] are byte (8 bits) index values: If the file is compressed, the first byte of the file indicates the number of bytes used for each elevation data and must be 1 (0x01). The next four bytes indicates the initial position of the first row of data, and the sequence continues with the initial position of each row for all rows. A compressed row is flagged by the first byte, which must be 0x31. The difference between the row to be read and the next row is used to read the set of compressed values, which are compressed using zlib compress function. The uncompressed data will correspond to the material index values.

A.5 GIS Material Categories File

Material categories file describes the material name associated to the index stored in the data file. For each integer value present in the data file, the name of the material is defined. The categories file is stored at the cats directory. For a GISMap defined in the TITAN2D GUI, the corresponding material categories data file is named GISMap_Mat

and should be located at: GIS Information Main Directory/GIS Sub-Directory/GIS Map Set/cats/GISMap_Mat. The categories file is an ASCII file, with the following lines:

<max_cat> categories

Vector map: <some_name>

0.00 0.00 0.00 0.00

<ind_a>:<Mat_a>

<ind_b>:<Mat_b>

.

<ind_n>:<Mat_n>

<ind_max_cat>:<Mat_max_cat>

<max_cat> is the biggest index of material; <some_name> is a name (not relevant here); <ind_a> is the index of material A; <Mat_a> is the name of material A; <ind_b> is the index of material B; <Mat_b> is the name of material B;); <ind_n> is the index of material N; <Mat_n> is the name of material N;); <ind_max_cat> is the biggest index and corresponding to material M; and < Mat_max_cat> is the name of material M. An example categories file with 5 categories and biggest index 30 data would be: # 30 categories

Vector map: FrictionMaterial

0.00 0.00 0.00 0.00

4:MaterialA

15:MaterialB

18:MaterialC

20:MaterialD

30:MaterialE