WORKSHOP SESSION 2A: Extracting landscape metrics for tectonic interpretation (Hilley & Arrowsmith)

The following is a cookbook that will lead you through some of the functionality that you can use in the Stanford Standard-Option Landscape Modeling and Analysis Environment ((SO)-LAME). You are welcome to use the binaries or source for your own use. It is free. However, as with so many things in life, you generally get what you pay for. Thus, you can't get mad at me or hold me responsible for problems in the code or issues with compilation on exotic platforms. My hope is that the tools provided in LAME are general enough to be used (and improved upon) broadly. Thus, if you make any improvements to the code, please send me a copy so that I can bundle it up with the newest distribution of LAME.

LAME was written (and continues to be written) by myself and Eitan Shelef. If you use it and like it, please thank us by buying us a beer at one of the national meetings.

Section 1: Preprocessing steps to get ALSM data ready for LAME

Steps in preparing DEM for LAME analysis (These have been done for you for expediency):

1) Import gridded ALSM data into ArcGIS (preferably Arc/INFO), and georeference to a distance-based coordinate system such as UTM.
2) Fill any NODATA values in interior of analysis area if necessary.
3) Smooth data if necessary.
4) Export data as a floating point grid.
5) Import floating point grid into RiverTools.
6) Calculate D8-based flow directions using iterative linking method in RiverTools.
7) Calculate upstream areas and Horton-Strahler order using RiverTools.

Step 1: Import gridded ALSM data into ArcGIS (preferably Arc/INFO), and georeference to a distance-based coordinate system such as UTM.

The procedure used to complete this step will vary depending on the specifics of the data that you are using. For example, if filtered point cloud data are used as your initial input, you will need to interpolate these to a regular grid using the algorithm of your choice. Kriging, as implemented in GSLIB, provides a means of interpolating irregular data to a regular grid – this package is still free the last time I checked and provides a variety of variogram models and visualization tools that may be used to interpolate data in many dimensions. Once a gridded dataset has been produced, LAME requires it to be projected using units consistent with the elevation units of your dataset. For example, if your interpolated Digital Elevation Model (DEM)'s original horizontal increments are in degrees (a geographic projection) and the elevation units are in feet, you must project the DEM into a distance-based coordinate system (such as

UTM in which the horizontal coordinates are Easting and Northing in units of meters), and adjust the vertical units appropriately.

Step 2: Fill any NODATA values in the interior of the analysis area if necessary.

In many cases, DEMs produced by commercial vendors have NODATA in the interior of the DEM that reflect the fact that no ground-return points were identified within a particular pixel. However, when such values are encountered by flow routing algorithms, they are often treated as sinks in the topography, and so flow routing and estimates of basin area at all points downstream of these NODATA values may be impacted by their presence. Thus, to ensure continuity of flow across the grid, values for these NODATA points must be interpolated from surrounding points.

The most straightforward way to do this is using the GRID module of Arc/INFO. To do this, open a command prompt, and navigate to the directory containing your DEM (which should be stored as an Arc/INFO GRID to do this part of the processing) using the "cd" command. Then, use the "arc" command at the prompt to start Arc/INFO, and then type "DISPLAY 9999". Then, start the GRID module by typing "grid" at the command line. Finally, you can fill the NODATA values in your grid by using the following command at the "Grid:" prompt:

Grid: interpdem = con(isnull(dem),focalmean(dem,rectangle,3,3,data),dem)

where "dem" is the name of your GRID-format DEM. This command finds all of the NODATA values in the DEM, and then fills them with the mean of the surrounding values that actually have data. However, you can imagine that there may be positions in the DEM where there NODATA values are surrounding a point on all sides, which would register a NODATA value as a result of this operation. Thus, it may be necessary to repeat this command several times to progressively fill in large NODATA patches in your dataset.

To check if the operation indeed filled all NODATA values within the DEM, you can use the following commands at the "Grid:" prompt:

Grid: mapextent interpdem

Grid: test = isnull(interpdem)

Grid: image test

You will see a map displayed that has only one of two colors: black for points in the DEM that have valid data, and white for points that still contain NODATA. If you see NODATA points present in the area you wish to analyze, repeat the interpolation operation as follows:

Grid: interpdem2 = con(isnull(interpdem),focalmean(interpdem,rectangle,3,3,data),interpdem)

Grid: kill interpdem ALL

Grid: rename interpdem2 interpdem

Grid: kill test ALL

Then, repeat the check described above to see if NODATA values still exist within the area of the DEM that you would like to analyze.

Step 3: Smooth data if necessary.

In some cases, ALSM-derived DEMs will have a significant amount of noise on the topographic surface, which can be seen most clearly in a hillshade rendering of the elevation model. Sometimes, it is desirable to damp some of this noise by computing averaged values at each point based on surrounding points. Additionally, you could use a low-pass filter with an assigned cut-off frequency, but it is easiest (and often adequate) to simply use neighbor averaging of elevation values to damp much of the noise. To do this, enter the GRID module of Arc/INFO as described above, and use the following command to perform a neighborhood averaging of the elevation values:

Grid: filtdem = focalmean(interpdem,rectangle,3,3,data)

Where the two 3's in the above statement give the extent of the rectangle used to average values across the grid. If you wish to smooth by averaging points from farther distances than the surrounding cells, you can increase these values accordingly.

Step 4: Export data as a floating point grid.

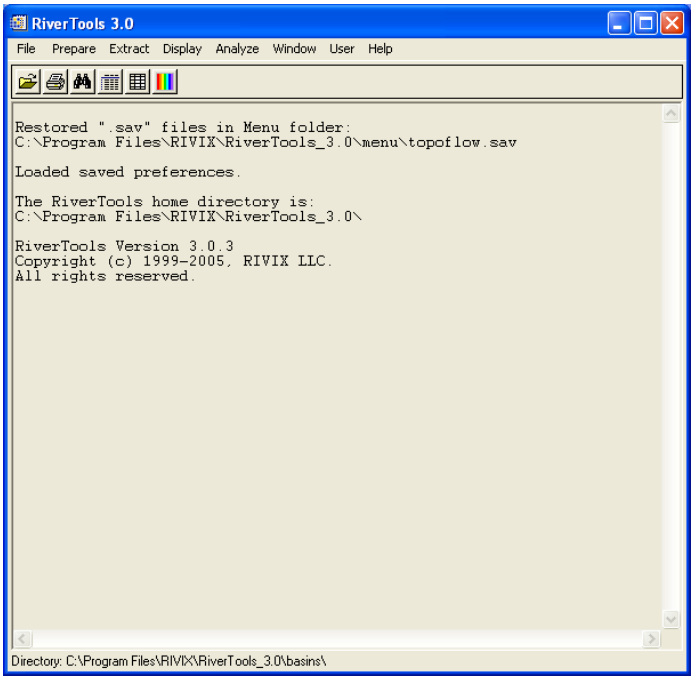
Once the DEM has been pre-processed in Arc/INFO, you can export it as a floating point grid that you can then read into RiverTools for further processing. To do this, type the following command at the Arc: prompt:

Arc: gridfloat filtdem filtdem.flt

Where filtdem is the name of the processed DEM in GRID format. NOTE: If you find yourself stuck in the GRID module of Arc/INFO, you can always exit by typing "quit" at the "Grid:" prompt.

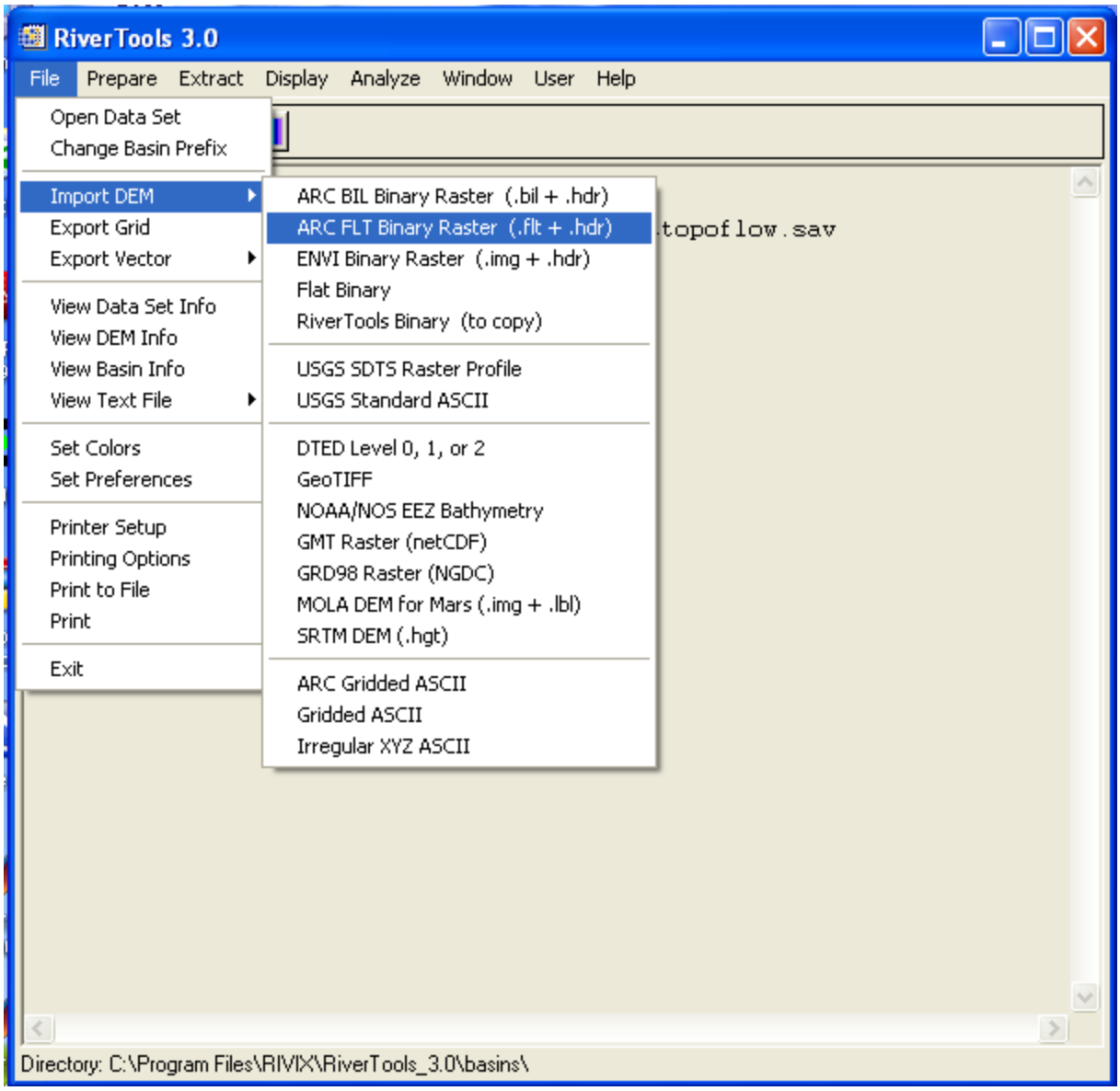
Step 5: Import floating point grid into RiverTools.

Next, you will need to import the floating point grid that you exported from Arc/INFO into RiverTools. To do this, first start RiverTools, where you will be met by the following dialog:

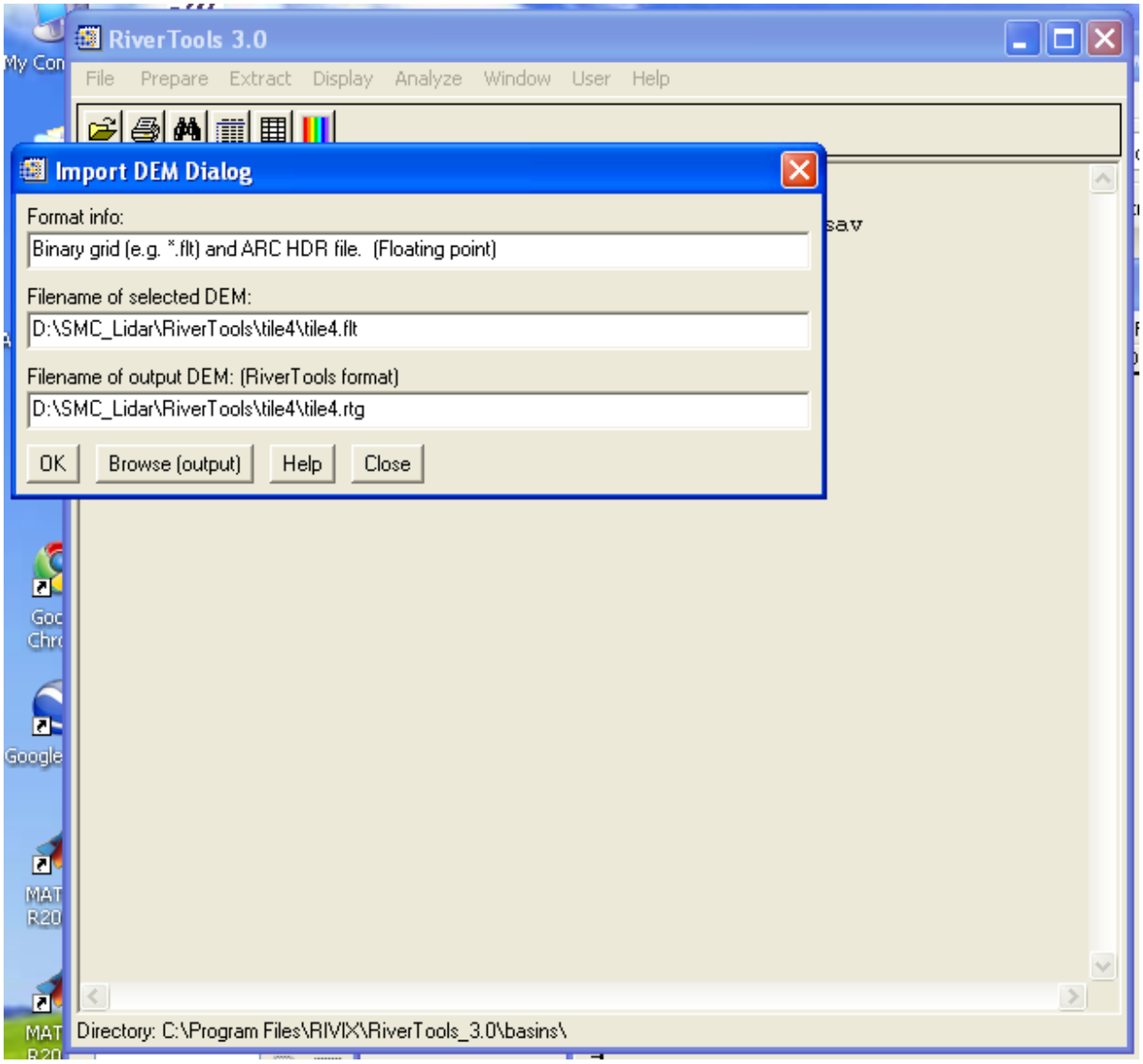


This is the main RiverTools dialog screen, which provides you with a set of menus that I will refer to hereafter as the “main menus”.

Next, import the floating point dataset by selecting the “File→Import DEM→ARC FLT Binary Raster (.flt+.hdr) as follows:



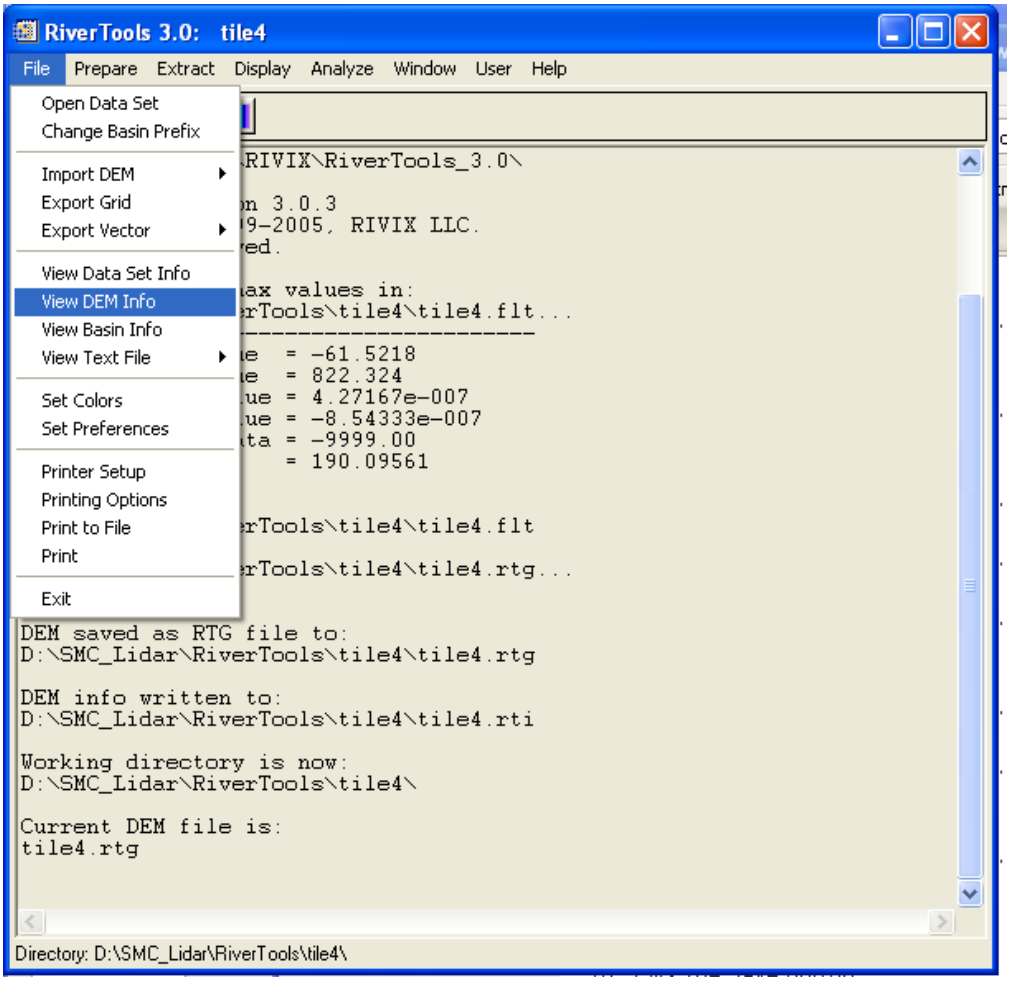
You will then need to navigate through the directory structure to find the file that you exported from Arc/INFO. Once you select the *.flt file, you will be greeted by the following dialog:



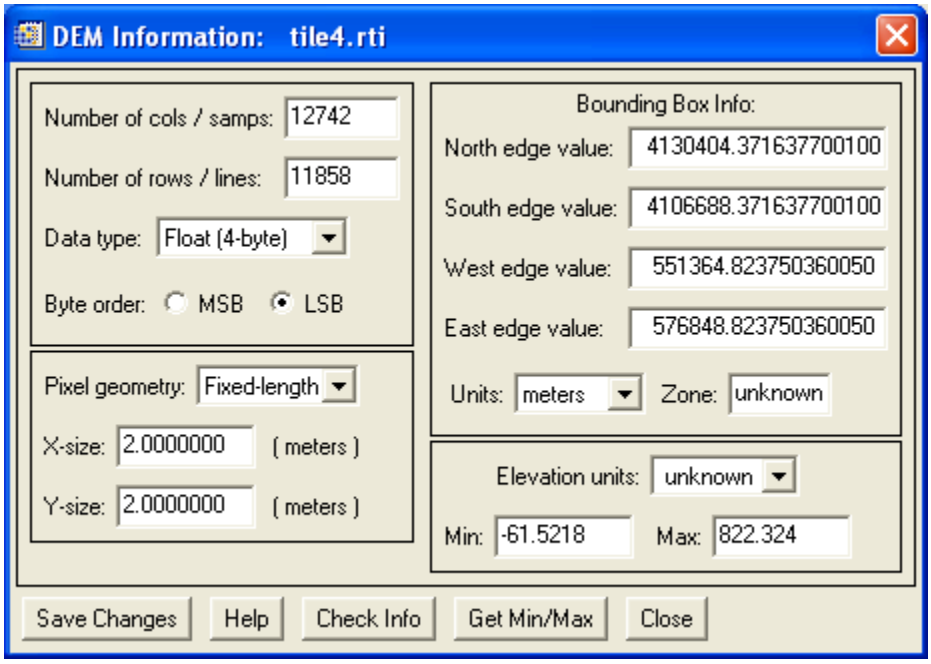
I suggest that when importing a .flt file, you should change the suffix on the Filename of output DEM to .rtg, since this is the standard suffix for a RiverTools Grid. Note that I have changed this suffix in the dialog above, although by default, RiverTools sets this filename's suffix to .flt.

Sometimes, at fairly unpredictable intervals, RiverTools presents an error when importing a DEM. This error can safely be ignored, so go ahead a click "OK" to proceed. Also, you might receive the following warning from RiverTools when the import is completed:

If this occurs, you will need to define the vertical units and UTM zone manually. To do this, click "OK" and then select "View DEM Info" from the "File" menu:

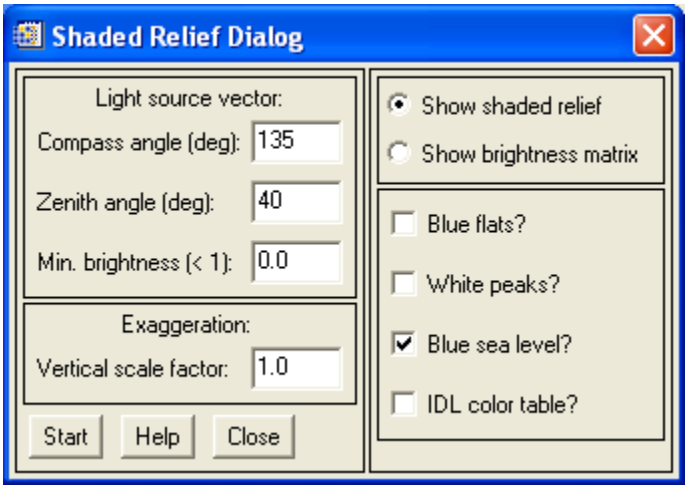


This will open a dialog that looks something like this:

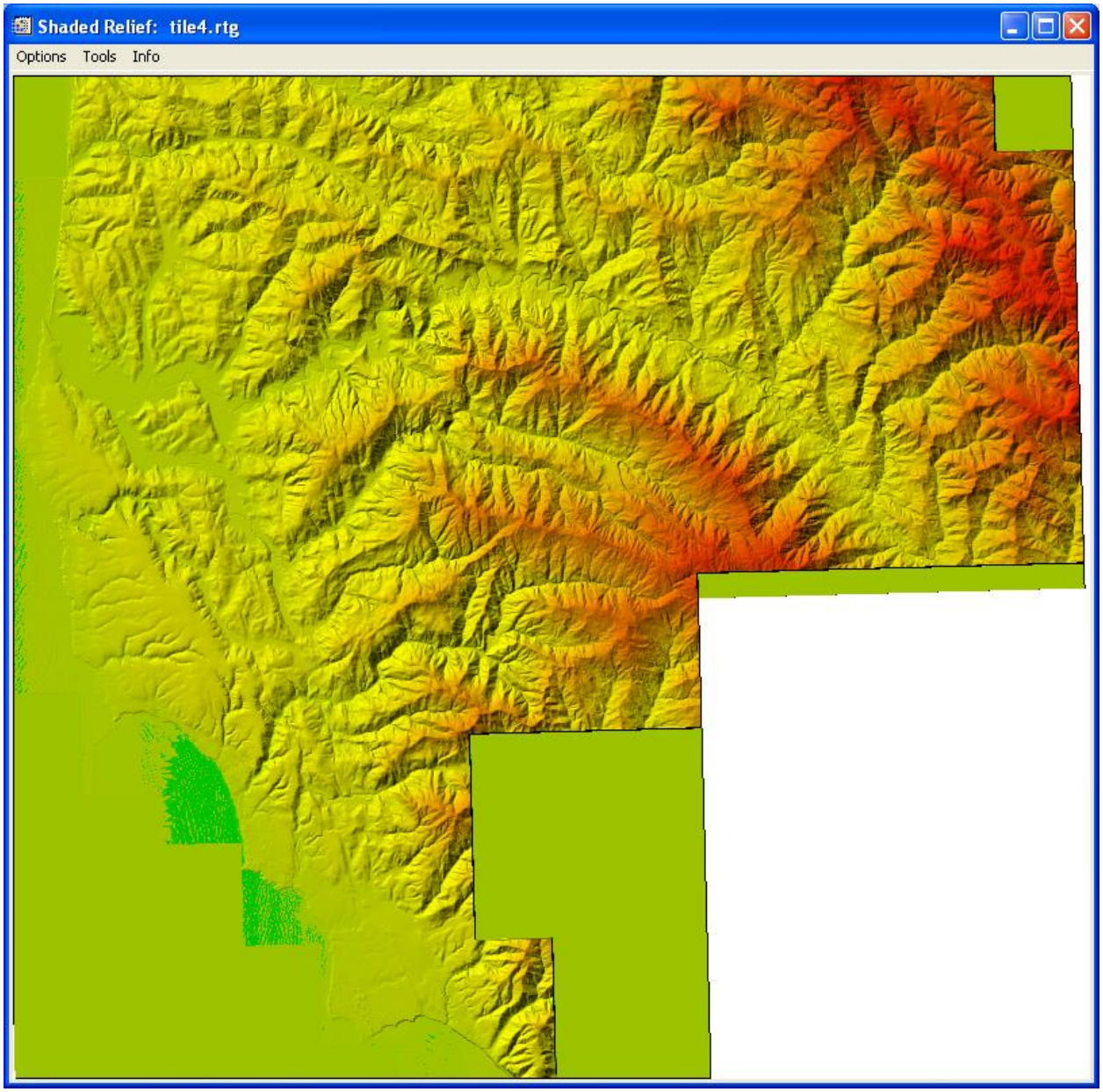


Make sure to define the "Elevation units:" field appropriately, and fill in the "Zone" if you are using a UTM projection. Then, go ahead and save the changes to the DEM information, allowing the program to overwrite the existing file.

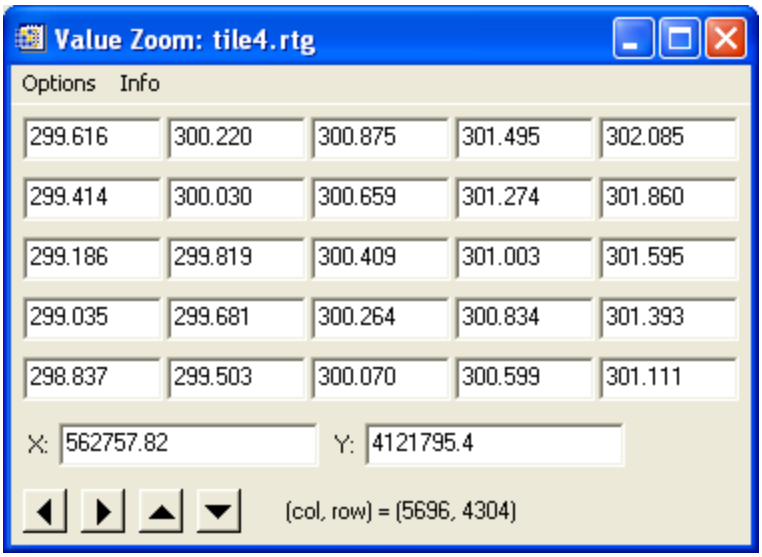
While it is tempting to jump right into enforcing flow across the grid, I suggest always checking the imported grid to make sure that the import was successful. You can do this by first plotting the DEM using the "Display→ Shaded Relief Plot" feature of RiverTools, and selecting your imported dataset:



Go ahead and "Start" the operation, which will open up a window that will display your data that should look something like this:



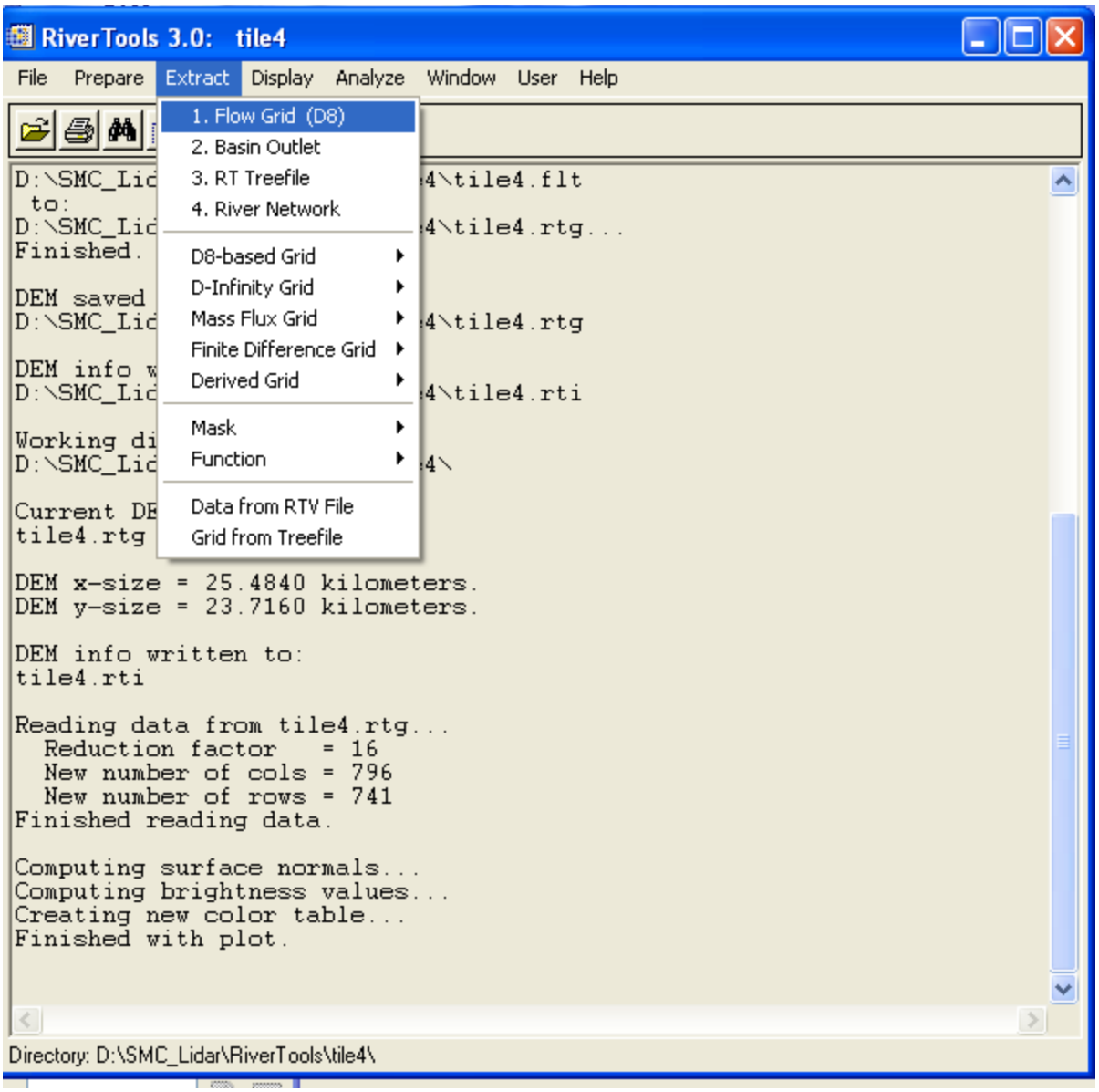
You can then use the “Tools→Value Zoom” feature of this window to select a particular point in the DEM to see if the values are indeed realistic:



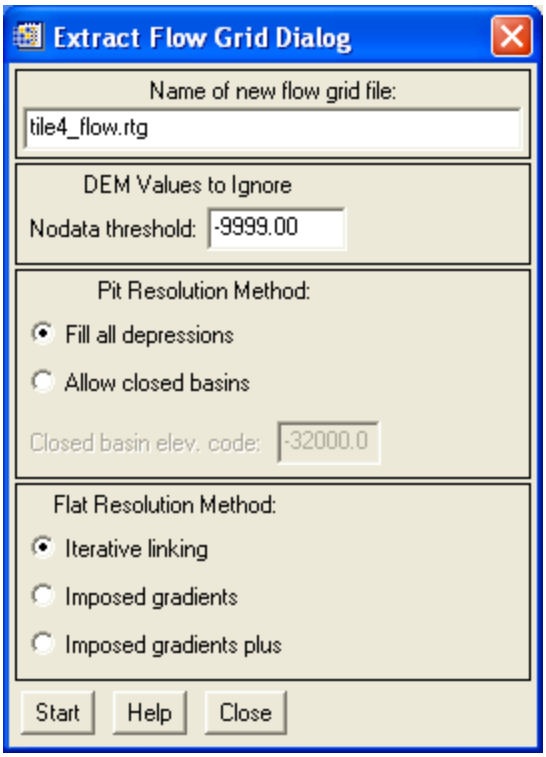
You can use the X and Y positions to check the RiverTools grid against the grid that was used to create the floating point file that you imported into RiverTools from Arc/INFO. If the values match, the import was successful. In my experience, if the import is unsuccessful, the values of the DEM will be unrealistic and so you can quickly tell if something went wrong by simply inspecting the values of the DEM using the "Value Zoom" dialog.

Step 6: Calculate D8-based flow directions using iterative linking method in RiverTools.

Once imported into RiverTools, you will need to correct the DEM for local sinks that may be produced by the real topography, interpolation, and/or incomplete sampling of the topographic surface. This will ensure that flow continuity is guaranteed across your DEM. It is important to point out that in many cases, internal sinks may indeed be present in the real topography, and if you are interested in capturing these features, you should identify and label them appropriately prior to carrying out this part of the DEM correction. To fill pits in the DEM and calculate the D8-based flow direction (flow is routed to the steepest of the 8 downslope neighboring cells (4 adjacent + 4 diagonal), select "Extract→1. Flow Grid" from the main RiverTools menu:



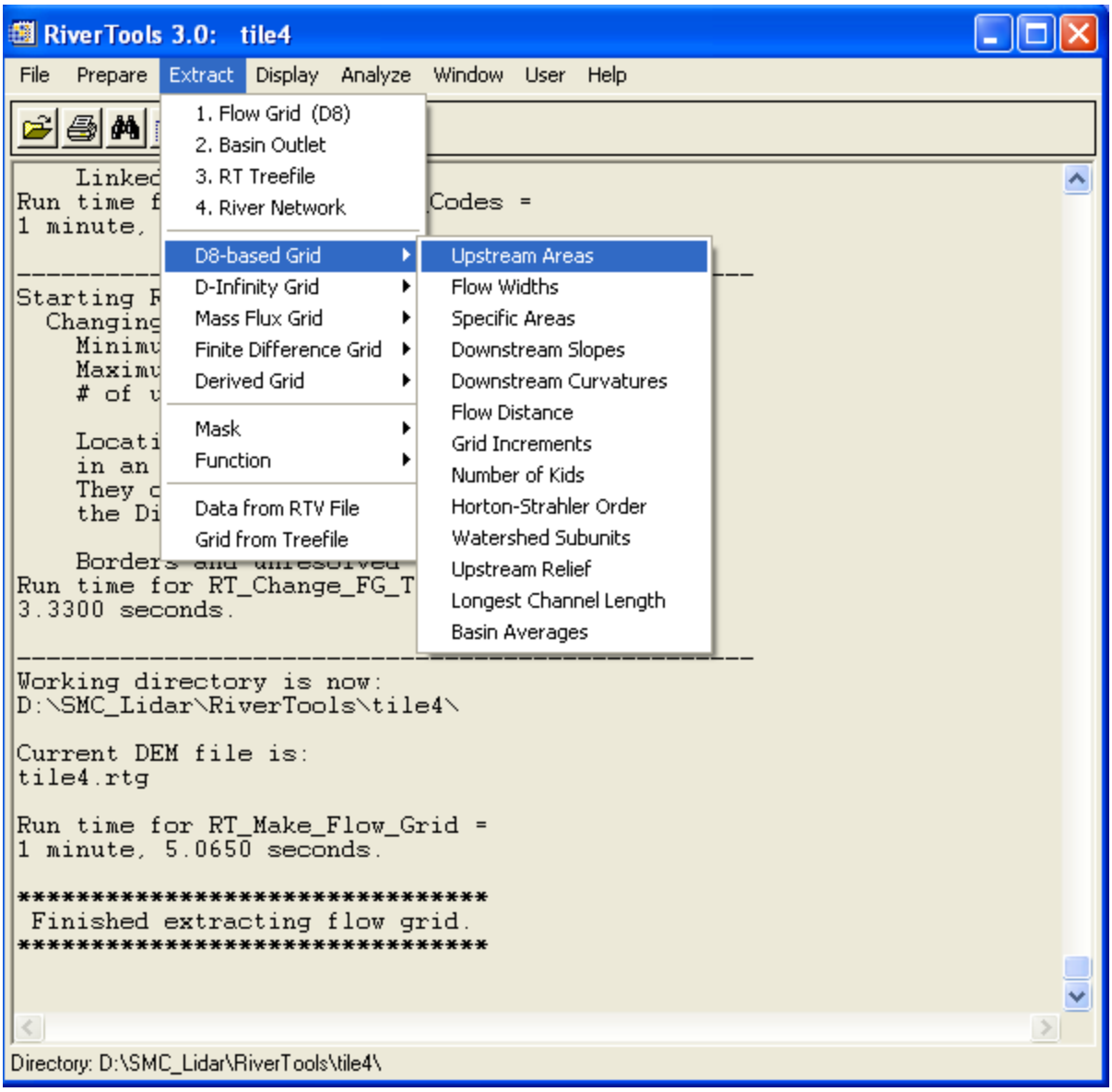
At this point, you may be alerted that a Raw DEM grid is missing. If this is the case, you can go ahead and click “Yes” to copy the imported version of the DEM into another file to make sure that the original data are preserved in this raw DEM file. Next, the “Extract Flow Grid Dialog” will emerge:



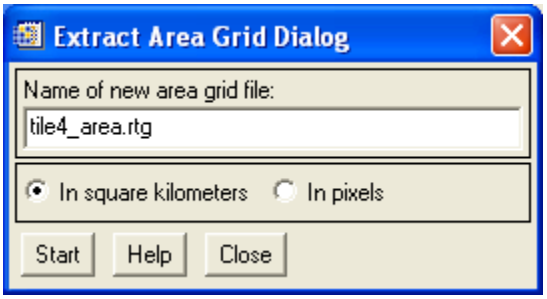
Here, you can specify the presence of closed basins identified by an internal sink point (which you must tag in the original DEM using a particular, unique elevation value to identify each sinks' presence). If you are interested in ensuring that flow is connected throughout the DEM, simply select "Fill all depressions". Also, while RiverTools has several algorithms for routing flow across the filled portions of the DEM, LAME can only handle the "Iterative Linking" method, so go ahead and select that flat resolution method. Finally, it will make your life much easier if you leave RiverTools to assign the name of each grid in this process, so don't change the field that assigns the name of the new flow grid file. Go ahead and click "Start". This will begin the DEM filling process, which can take quite some time depending on the size of the DEM.

Step 7: Calculate upstream areas and Horton-Strahler order using RiverTools.

Finally, you will need to extract grids from the flow direction grid that represent the upstream contributing area and strahler order at each point in the ALSM DEM. First, use the "Extract→D8-based grid→Upstream Areas" function to calculate upstream area in the DEM:

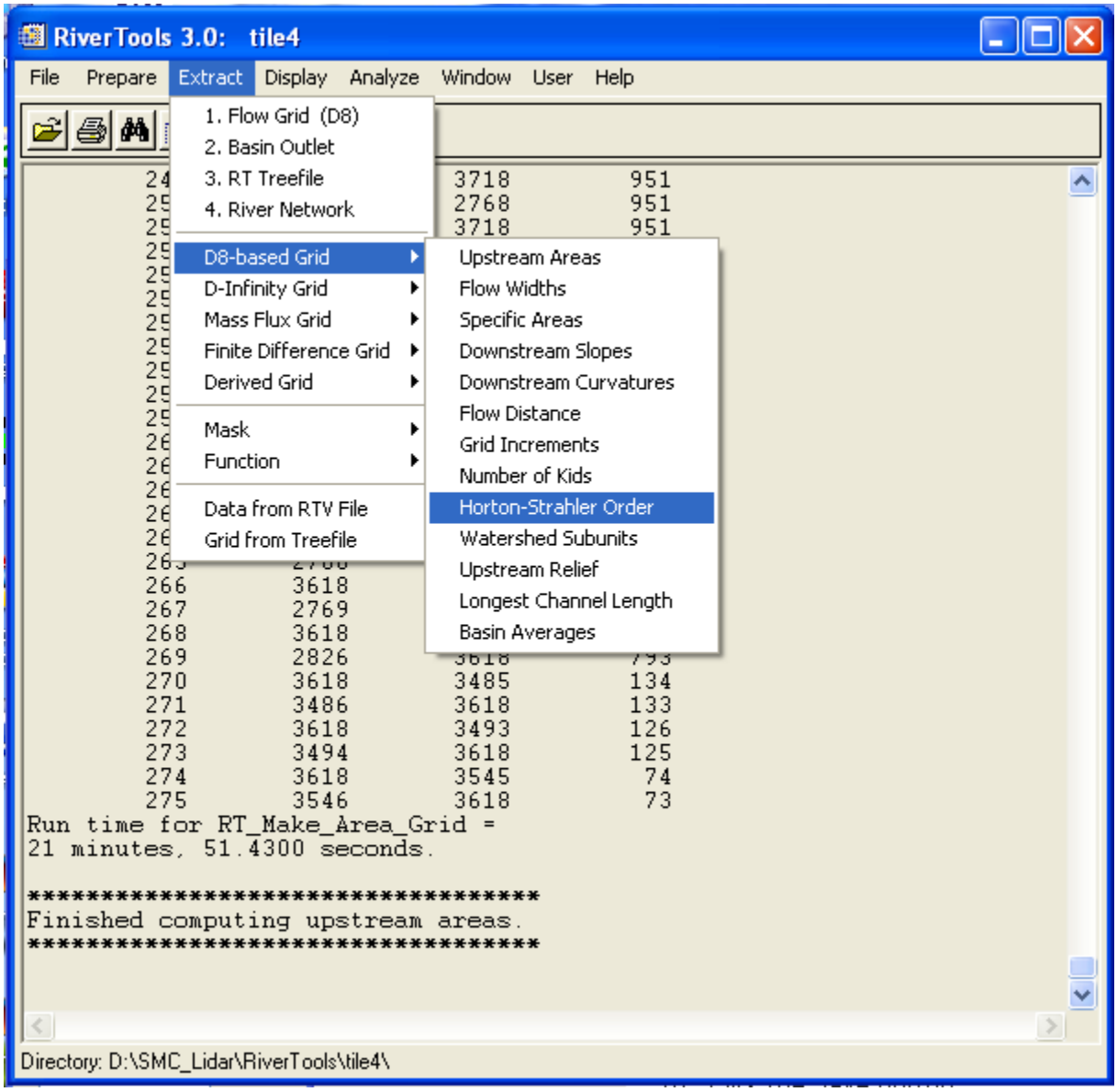


This will produce a dialog box that looks like this:



Make sure to calculate the upstream area in units of square kilometers, and then click “Start”. This will produce a grid whose values record the upstream basin area at each point in the DEM.

Finally, to compute the Strahler order for each point in the grid, use the "Extract→D8-based Grid→Horton-Strahler Order" function available from the main menu of RiverTools:



This will initiate a dialog that lets you change the name of this grid. For ease, just leave this field untouched, and select "Start" to compute the stream order of each point in the grid. Note that this stream order does not consider a threshold of channelization, and so each point contributes to the stream order, whether it is along a hillslope or within a channel.

This concludes the data preprocessing steps that are required to perform the topographic analysis in LAME.

Section 2: Importing DEMs into Matlab and Displaying Data

There are two steps in importing data into matlab:

1) Read each of the floating point files you created in RiverTools into Matlab using the command line (files include the DEM, flow directions, basin areas, and stream orders).
2) Display the DEM as a shaded relief map.

Step 1: Read binary files into Matlab

I have created a series of matlab scripts that are designed to interface RiverTools data with Matlab, and can also import and export results to Arc/INFO. When combined with the MEX interfaces to LAME described below, this can be used to carry out the different topographic analyses described and re-export the results to ArcGIS for plotting in a pleasant, easy-to-use environment. Alternatively, I have provided a number of functions in LAME that can create semi-transparent shaded relief maps atop which can be plotted various continuous or point values derived from the topographic analysis. It is your choice as to how you wish to display the values of the topographic metrics and archive the results for further analysis.

To read the binary files produced by RiverTools, start Matlab, and navigate to the working directory in which your binary files are contained. Next, read them into Matlab using the following commands:

```
>> dem = rt_importrawdem('binaryfileprefix');

>> area = rt_importarea('binaryfileprefix');

>> order = rt_importorder('binaryfileprefix');

>> fd = rt_importflow('binaryfileprefix');
```

Note that in each of the above commands issued at the Matlab prompt, the "binaryfileprefix" is just the prefix of the basin names. These scripts automatically put on the appropriate suffix to the filename (such as "_rawDEM.rtg", "_area.rtg", "_order.rtg", and "_flow.rtg"), so you need only include the original name of the basin that you used in RiverTools. Make sure that the prefix is bounded by two apostrophes, as this is the format that Matlab requires for character strings. You can then save these grids if you want as follows:

```
>> save mygrids dem area order fd
```

This command creates a MAT file in which it stores the different data structures that represent the RiverTools' processed data.

➔ Note that I have done all of these steps for you in the workshop exercises to save time and to allow us to focus on the analysis. Thus, the preceding steps are for your reference when processing your own data. We will start the exercise with Step 2, described below ←

Step 2: Display the DEM as a shaded relief map.

LAME has a MEX module that will compute the RGB values of a colored, shaded relief map. As a side note, there is similar function in the LAME MEX interface that will allow any continuous grid to be overlain transparently over the shaded relief DEM in Matlab. Each of the Matlab grid structures that you created by importing the binary files from RiverTools contains the coordinate system information (although not the projection information). Thus, the units of the map will represent the easting and northing coordinates of each point on the ground.

To display the shaded relief map, issue the following command at the Matlab command line:

>>plotgridandhs(<dem>,<azimuth>,<elevation_angle>,<minimum_data_value>,<maximum_data_value>);

Where <dem> is the name of the structure containing the DEM

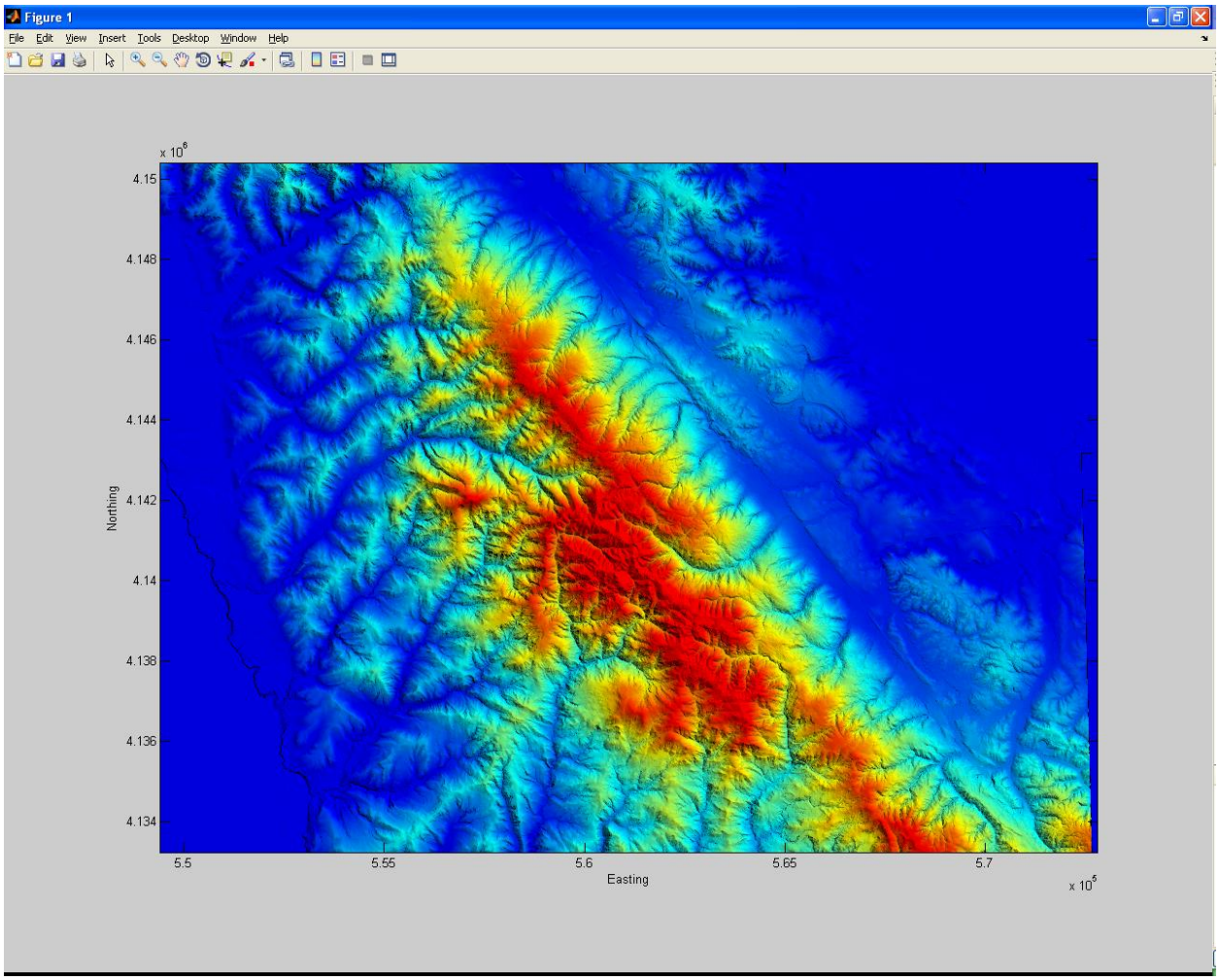
<azimuth> is the azimuthal angle from which the sun is shining to create the shaded relief part of the map.

<elevation_angle> is the angle between the horizontal and the sun's position, again for the shaded relief part of the map.

<minimum_data_value> is the lowest data value considered in the linear color ramp. Values of the DEM less than this value will be colored blue.

<maximum_data_value> is the largest data value considered in the linear color ramp. Values of the DEM greater than this value will be colored red.

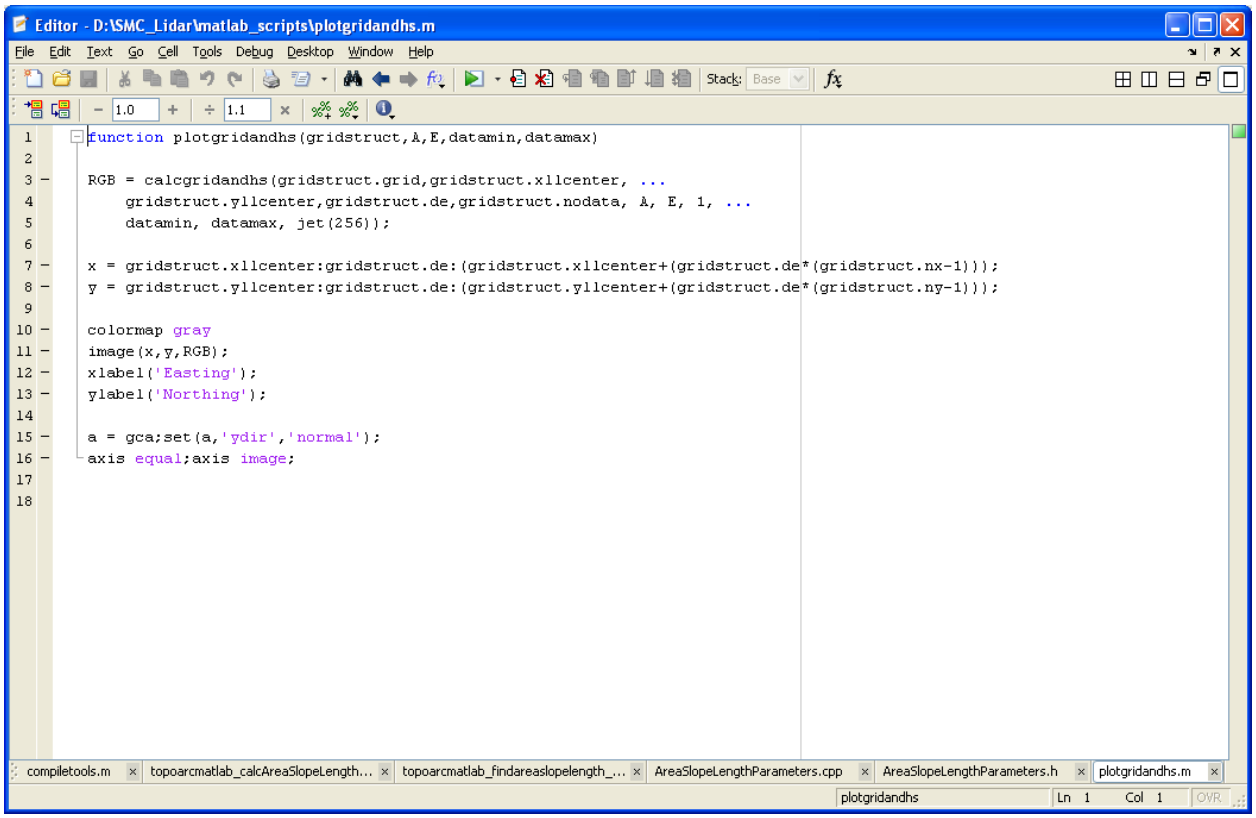
An example of the output produced by this command is shown below:



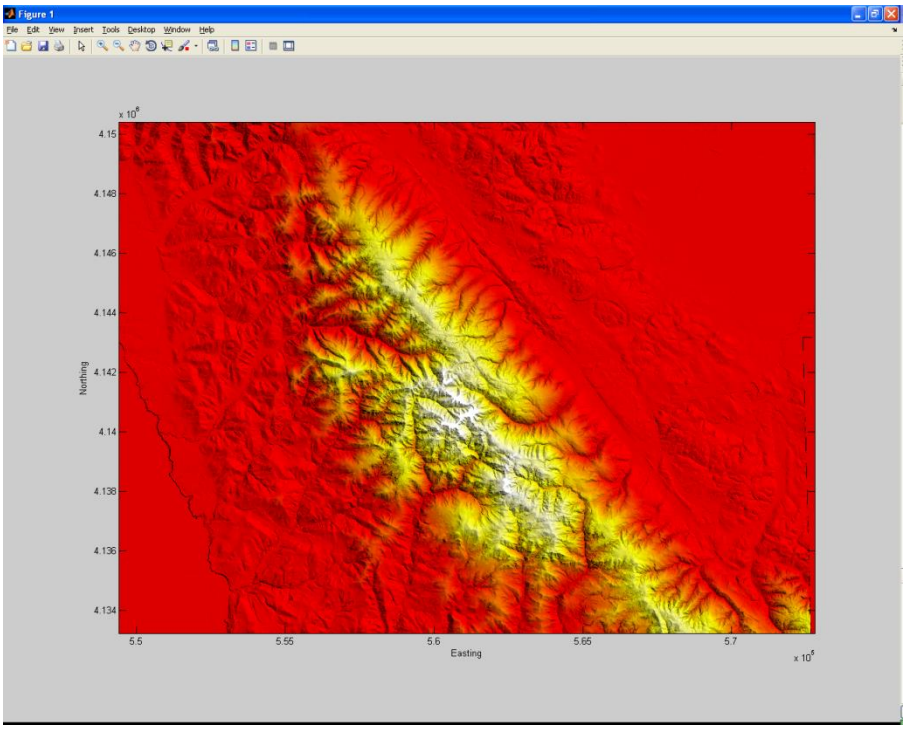
Now, go ahead and familiarize yourselves with the topography using this image. If you would like to color code the map according to a linear color ramp with a different range, you can clear the graphics window by typing:

>> clf

At the Matlab prompt. Then, you can repeat the above command with a different maximum and minimum data range. Additionally, you can manually change the color ramp itself in the plotgridandhs.m script. Go ahead and open the script (which is located in the "matlab_scripts" directory) – it should look like this:

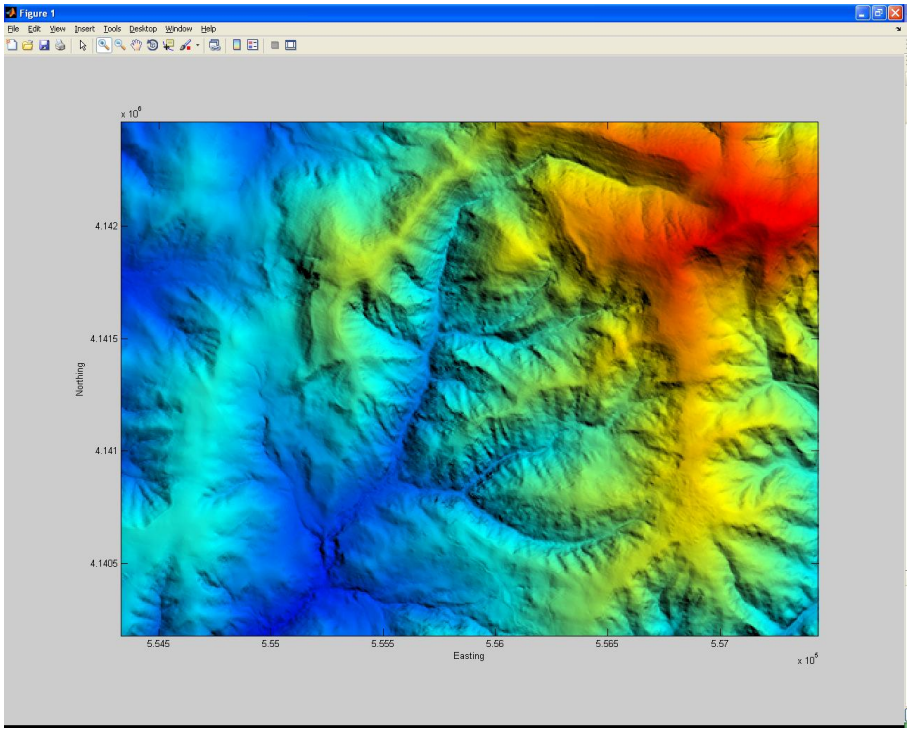


Notice that line 5 of the script has the command “jet(256)”. Jet is the default Matlab blue-to-red color ramp. There are others built into Matlab as well. Most of them are quite obnoxious, but you might try changing “jet” to “hot” to see how the color ramp will change. You can then “clf” and then replot the DEM, and you will see something like this:



So, you can probably see why the script defaults to jet.

Also, you can use the zoom tools to investigate the topography in more detail, so go ahead and explore the features of the topography for several minutes:



Section 3: Performing the topographic analysis

The topographic analysis we will perform today will compute several metrics across the landscape, and in the next section, we will combine the results of individual groups into a single database that we will compare with various other geologic and geochronologic information I have provided for this area. There are five parts to this section:

1) Define basin outlets.
2) Make area-slope plots to define channel concavities for individual basins and then compute a reference concavity for all basins.
3) Map channel steepness values back onto the landscape and export these values to ArcGIS for further plotting and analysis in the next section.
4) Calculate mean channel steepness values in each basin and use this to create a basin map of mean channel steepness.
5) Calculate mean slope angle and convexity for basins whose areas are < 1 km^2, but whose maximum basin area in total is > 0.5 km^2 that are nested within selected basins. Slope angles and convexity values will be computed for planar, concave, and convex portions of the landscape separately, in addition to mean values within these basins regardless of convergence/divergence along hillslopes.

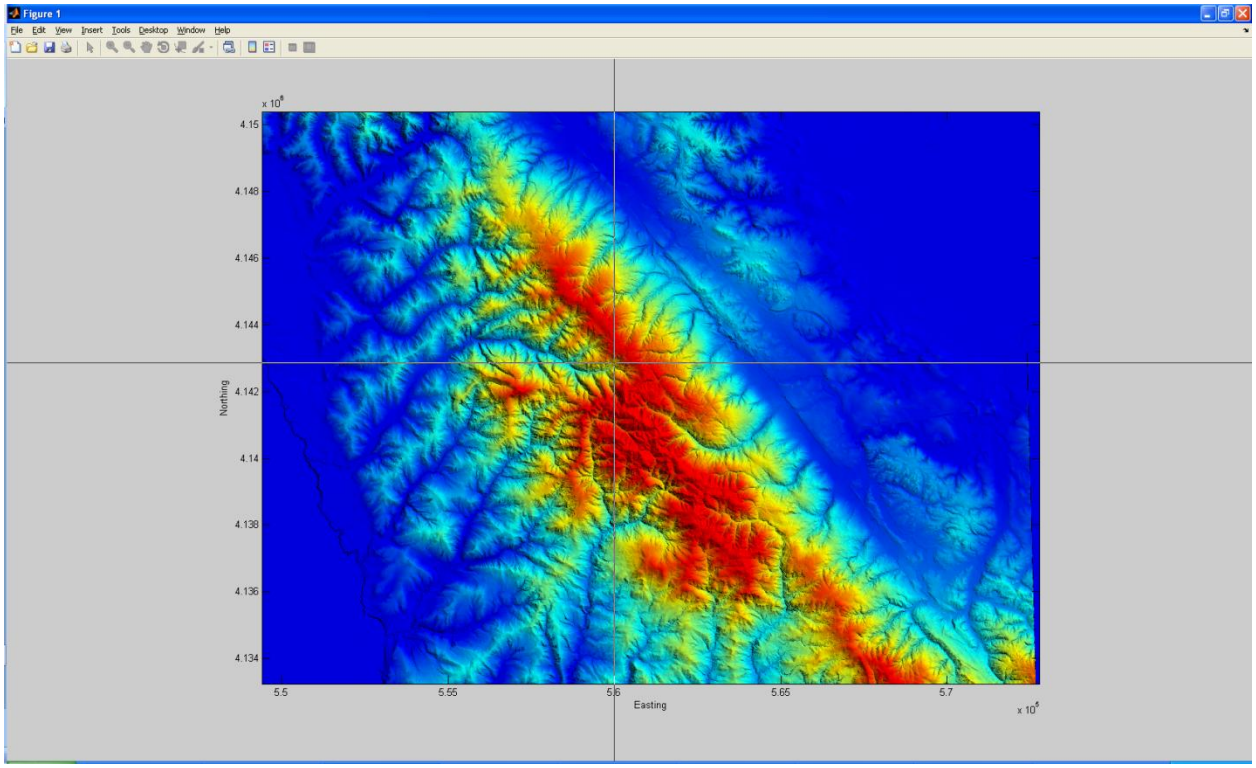
Step 1: Define the basin outlets.

Each group has been provided with one of four tiles of ALSM data from San Mateo County in the Bay Area of California. The first step in analyzing the topography is to identify those portions of the landscape that you wish to analyze. To do this, LAME contains some tools for both tracking flow downslope from a particular point, as well as recursively searching the channel network upslope of a set of designated points to find all locations draining to each of the outlets. This can be done interactively with the Matlab MEX interface.

I have written a simple wrapper script which will help identify and label the outlet points you may wish to analyze as part of this exercise. First, make sure that you have an adequate view of the shaded relief DEM as created by the plotgridandhs command described above. Next, type the following at the Matlab command prompt to begin interactive selection of outlet points:

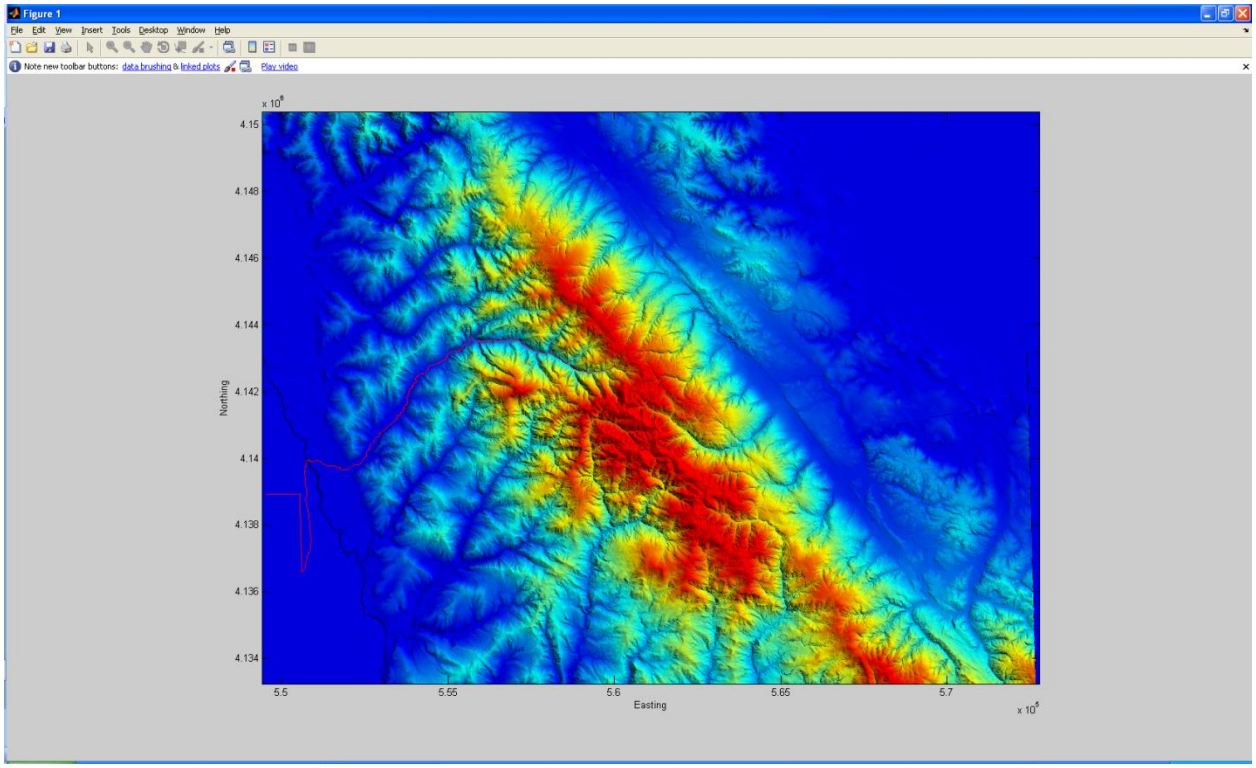
```
>> [xo, yo] = selectoutletpoints(fd);
```

This will begin an outlet selection process within the active figure, which should be your ALSM map of the area.

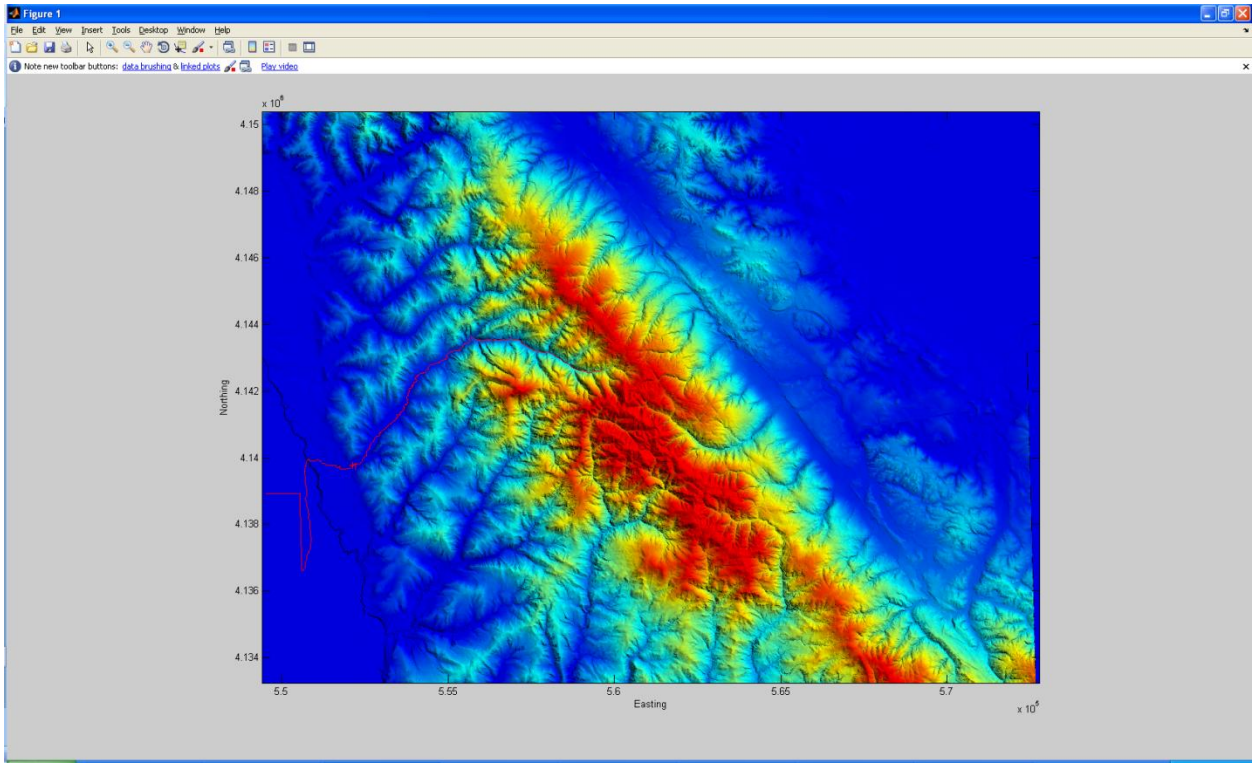


The outlet selection process consists of two steps: first, you outline a flow path down the basin, and then you click on the point along that flow path that you wish to define the outlet. This makes sure that the outlet you ultimately select lies along the basin's flow path and not on a close, but much more locally draining point.

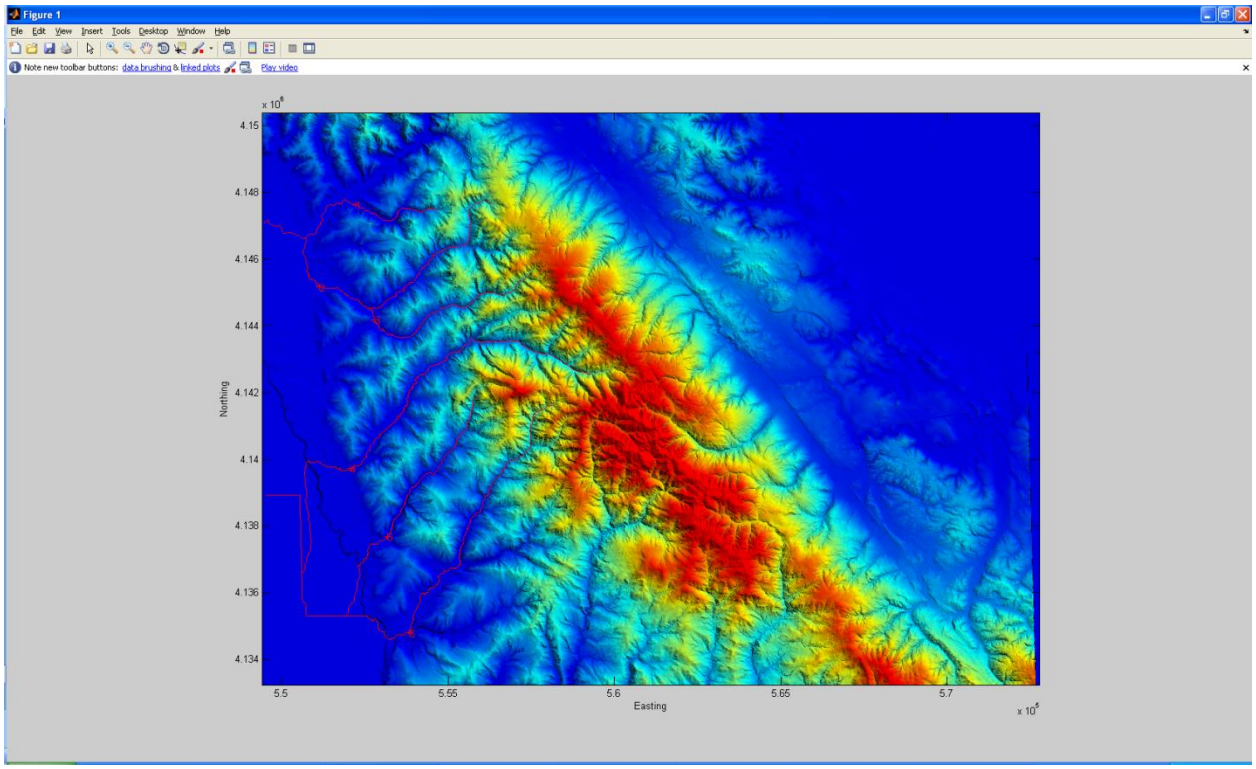
First, use the left button on your mouse to define the flow path. To do this, go to the headwaters of a basin for which you would like to define an outlet. This first point doesn't have to be at the basin divide—the recursive search algorithm will find the basin divide in the next step of this exercise. Just click on a point that is close to define a flow path down the topography. That flow path will look something like this:



Next, you can select the outlet point along that flow path with a second click of the left mouse button. This will label the selected outlet with a red “+” and place a number corresponding to the basin outlet number adjacent to the cross:



If you wish to select additional outlets, keep using the left button to first define the flow path and then select the outlet. If instead you wish to stop the procedure, just use one of the other two buttons on the mouse and the process will stop. Once you are done, the program will return the (x,y) coordinates of each of the outlets that you selected in the variables you used to execute the command, in this case, xo and yo. Here is an example of the above DEM in which I have successfully selected all of the basin outlets I wish to analyze:



Looking at the values of xo and yo by typing their names at the Matlab command line, we can see that the outlet easting and northing values are indeed present:

```
>> xo

xo =

  1.0e+005 *

    5.5223    5.5114    5.5279    5.5207    5.5318    5.5384

>> yo

yo =

  1.0e+006 *

    4.1476    4.1452    4.1442    4.1397    4.1377    4.1348

>>
```

NOTE: This is common sense, but it is always prudent to save your work, especially your outlet points, which will be used throughout the rest of the exercise. You can do this by typing "save outlets xo yo" at the Matlab command line to save the values of xo and yo in a file called outlets.mat.

Step 2: Make area-slope plots to define channel concavities for individual basins and then compute a reference concavity for all basins.

We will now use the locations of each of the basin outlets to calculate channel steepness values across the basins in which we are interested. Channel steepness is basically a metric that normalizes channel slope at a particular location for the systematic changes that correlate with upstream basin area. This allows us to reference some measure of the slope of a channel at each location in the basin to the value that each channel slope might assume if it were located at a particular reference basin area. For a complete discussion of how this works, see Wobus et al. (2004). Basically, for portions of the channel network that are devoid of the effects of debris flows, it is empirically observed that channel slopes decrease as a power function of drainage area such that when the two are plotted in logarithmic space, they can be regarded as somewhat linearly related to one another. The slope of this relationship in log-log space is referred to as the channel concavity, while the y-intercept of the regressed data reveal the channel steepness—a measure of the channel slope that might be expected at $x = \log(A) = 0$. By using all of the area-slope measurements within a drainage basin (or perhaps an entire DEM), the overall best-fitting channel concavity can be determined. If this concavity is assumed constant across the basin or DEM, it can be used to calculate the steepness value (y-intercept of the log-transformed area-slope pairs) for each point in the landscape for which we measure basin area and channel slope to identify spatial deviations in the relative steepness of channel links that cannot be explained by differences in basin area. We might intuit that such changes might reflect spatial variations in the rates of tectonic rock uplift, communication of changing baselevels throughout a landscape, changes in the underlying bedrock substrate or flow/sediment properties, variations in factors such as climate, and/or changes in the nature of channel erosion processes.

Typically, even in ALSM data, deviations of DEM elevations from true elevations of the land surface can cause fairly substantial mis-estimates of channel slopes, which can be quite small and thus susceptible to noise. We follow the method of Wobus et al. (2004) in implementing a means of ameliorating this effect (with some modifications of their original method to allow recursive calculation of channel steepness throughout an entire drainage network). Briefly, channel segments within the selected basin are isolated according to Strahler order and elevation values sampled at a prescribed interval are used to calculate a second-order finite-difference approximation of slopes between these equal intervals. This unevenly samples channel slopes across the landscape in that steep channel segments will be sampled more frequently than those with low slopes. Once channel slope is calculated for a particular point, the basin area draining to that point is interpolated along the channel from the Upstream Area grid calculated in RiverTools. This process is repeated for each sampled point along each channel segment, and the process is repeated recursively up the drainage network until a prescribed, minimum Strahler order is encountered. This prevents the recursive algorithm from searching far onto the

hillslope of the landscape, thus saving significant computation time because of the deep recursions required to sample these portions of the surface.

Luckily for you, all of this process is implemented in a single command in the LAME package that has an easy-to-use MEX interface with Matlab. There are two separate steps for calculating channel steepness across the DEM tile for the basins you selected. The first is the extraction of area and slope values for each of the basins, and the second is segregation of those points in the DEM that likely represent fluvial channels from those that may be derived from other processes such as debris-flow scour.

First, you must calculate the area and slope values for points at which channel slope can reliably be calculated. As discussed above, to do this, you will need to specify a sampling interval over which channel slope will be calculated. Additionally, you will need to specify a minimum stream order at which the recursive algorithm will no longer search the channel network. Again, this later parameter is introduced to save both compute time and memory, as considering every point in a particular basin as a channel introduces a heavy burden on computation. As an example, for this particular dataset, you might set your sampling interval to five meter intervals, and consider points whose stream order is greater than three when calculating area and slope points across the DEM. To do this, first define the sampling interval and minimum stream order as follows using the Matlab command line:

```
>> si = 5; minorder = 3;
```

Then, you can calculate the location, area, and slope values at each sampled point within each basin (defined by the outlets as described above) by issuing the following command at the Matlab prompt:

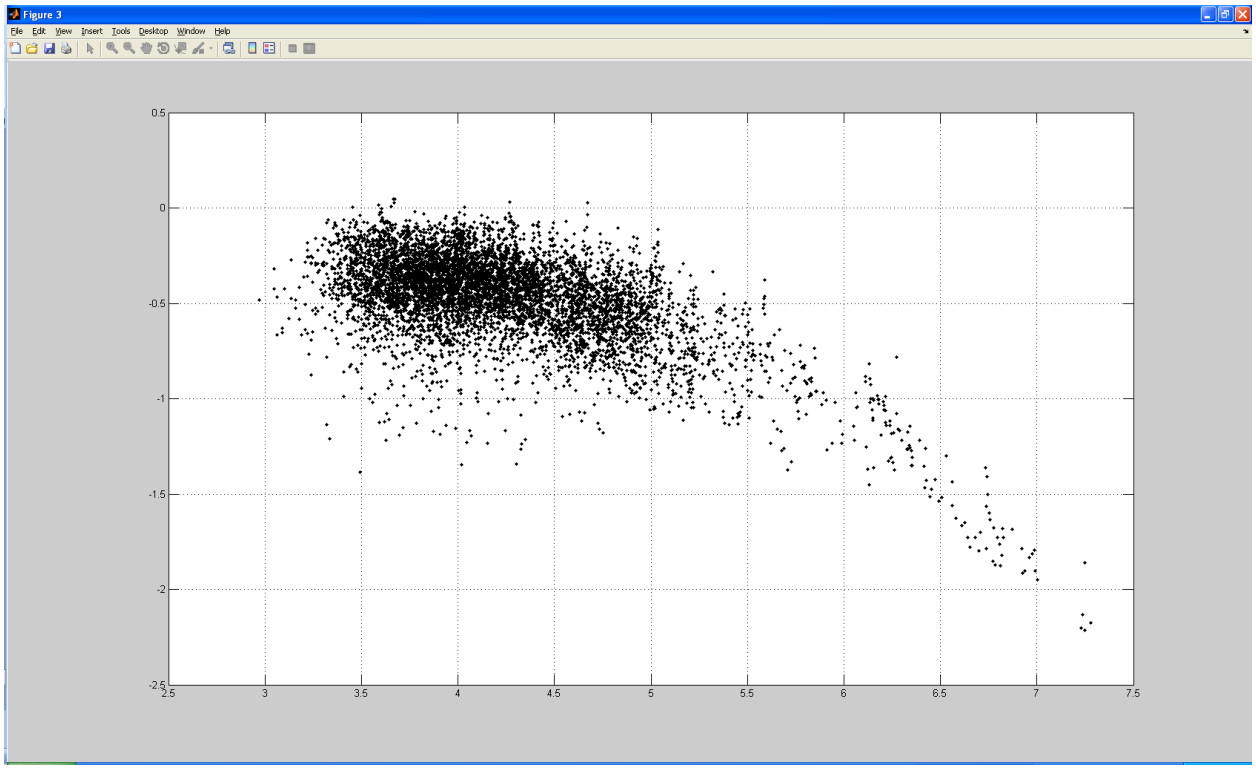
```
>> [b, a, s, x, y, I, j, n] = topoarcmatlab_calcareaslope(dem,area,fd,order,xo,yo,si,minorder);
```

This will execute the LAME C++ module for determining area at slope (returned in variables a and s, respectively) for each of the basins. The location of each of the points at which area and slope are calculated are stored in x and y, and the indecies of the DEM matrix that correspond to the row and column of those points' locations are stored in vectors I and j, respectively. The basin number is stored in the vector b. This vector contains a value corresponding to each area and slope value calculated at a point (x,y), which denotes the basin number in which each point is contained. For example, the first 400 values of x and y may have been derived from the first basin whose outlet you defined above, and so the vector "b" will have a value of one for these first 400 elements. Points derived within the second basin you identified above will have values of "2", the third "3", and so on. The final parameter "n" records how many DEM cell values are contained between a particular sampled point and the next sampled point above in the channel network. This value may be used to provide some declustering of channel steepness values, since the algorithm will return these values in more abundance along steep channel segments relative to those that are shallow. Again, you can save the results of this analysis using a save command such as:

```
>> save areaslope b a s x y I j n
```

In the second step of the channel steepness analysis, you will identify the range of areas over which power-law area-slope relationships appear valid, and then use this range to calculate the channel concavity for each basin, followed by calculating the best-fit reference concavity for all basins.

Channel concavity is defined as the slope of the line that best defines the relationship between area and slope in logarithmic space. For example, in the plot below, you can see how slope and area are related in a particular basin:



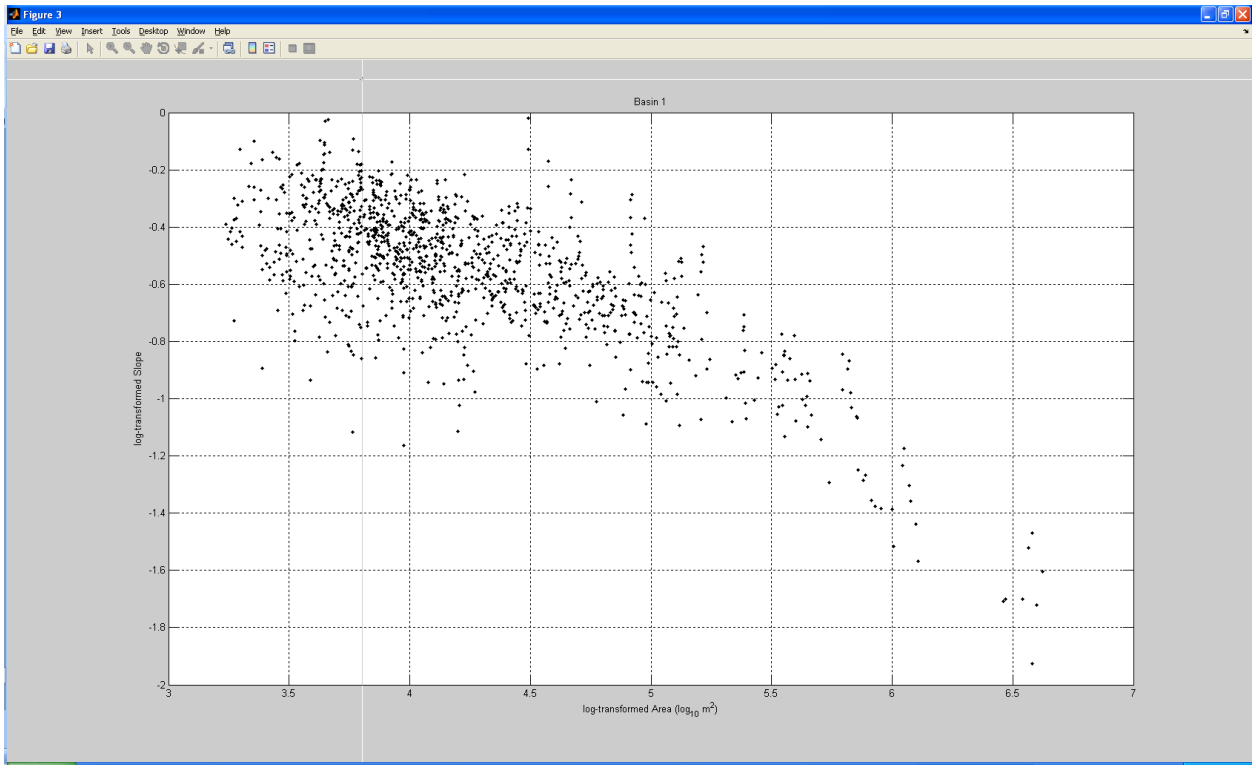
Where the a-axis is the log-transformed basin area at a point, and the y-axis is the log-transformed channel slope at a point. You can see that the points on the right define a relatively linear relationship between area and slope, and those points clustered in the middle and left portion of the figure show a somewhat linear relationship with a lower slope. Stock and Dietrich have interpreted this break in the area-slope plot as the location in the landscape where processes such as debris-flow scour and other hillslope processes transition to fluvially dominated channels. We will follow with this supposition in this exercise. Thus, to define the channel concavity, we first need to isolate those points that define what we feel to be fluvial channels from those dominated by debris flow and hillslope processes. We will assume that this location corresponds to the "roll-over" in this plot and define concavity values for those points dominated by debris-flow and hillslope processes (points to the left of the roll-over) and those points that may be dominated by fluvial processes (points to the right of the roll-over).

I have provided a Matlab function that will interactively allow you to select the roll-over point, and then manually define the concavity for the small basin areas of the landscape, as well as the large-basin area portions. To start the process of defining the channel concavities, type the following command:

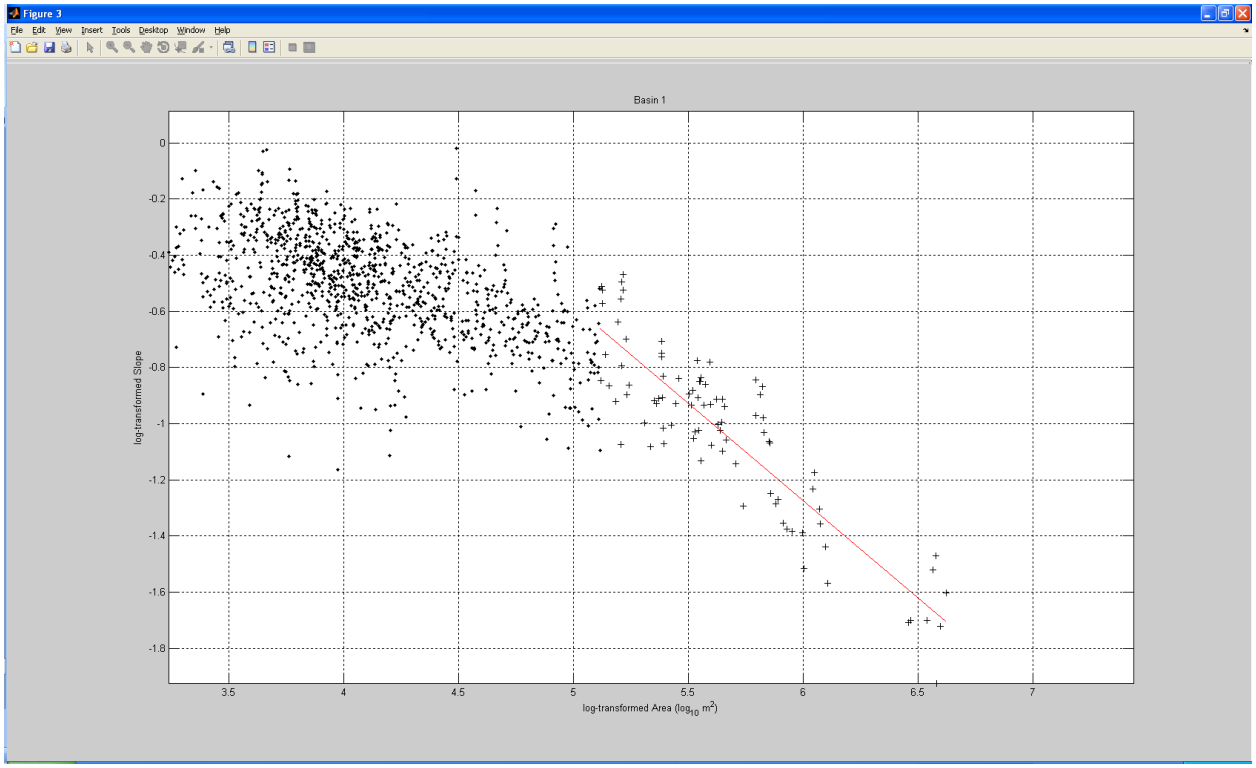
>> [conc_df,conc_chan,refconc_df,refconc_chan,type] = calculate_concavity(b,a,s);

The function will return vectors of values for the concavity selected for the small-basin areas (conc_df) in each of the basins, the concavity selected for the large-basin areas (conc_chan), the mean concavity for the small basin area points for all of the selected basins you defined earlier weighted for the number of points in each basin (refconc_df), the mean concavity for the large basin area points for all of ht eselected basins you defined earlier weighted for the number of points in each basin (refconc_chan), and a vectors of ones and twos corresponding to the points in b, a, and s, which denotes if you defined a particular point to the right of the rollover (channel processes, labeled “2”) or to the right of the rollover (debris-flow processes, labeled “1”).

Once you type this command, and interactive session will be carried out for each of the basins you selected. Each session will require four clicks from you, which will first identify the location of the rollover, and then ask you to locate points that define the concavities of points to the left and right of the rollover. After typing the command, you will be greeted by a screen that looks like this:

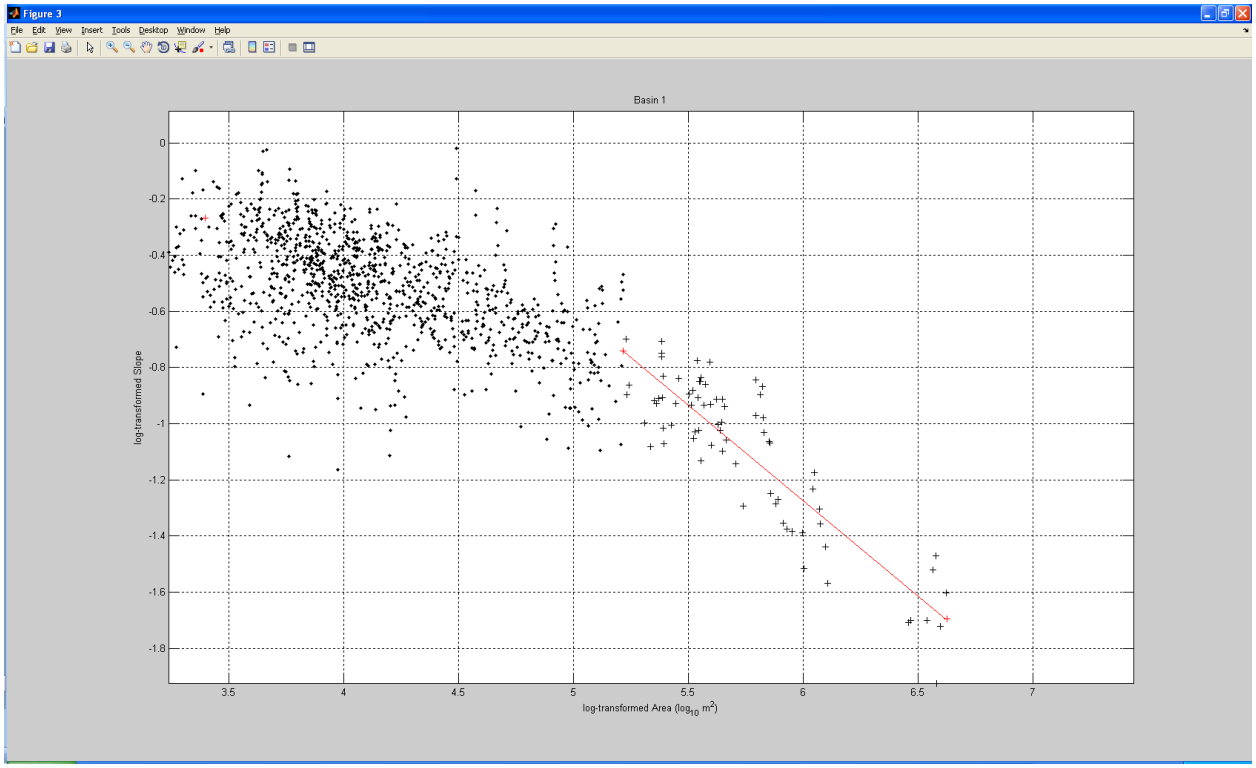


The first click will define the location of the rollover. In this particular basin, we can see that points with log basin areas > ~5 appear to have a slightly different slope than those to the left. We therefore define the location of the rollover by clicking on its location. Once you select this location, another window will be created in place of the old:



You can see in this window that all "channel" points are labeled as crosses, and all points that we might consider to be dominated by debris-flow and hillslope processes are labeled as points. Also, for your convenience, I have provided a best-fit regression line in red that shows the best-fit channel concavity based on the location of the rollover you chose.

Next, you can define the overall kinked trend of the adjoining line segments that appear representative of the area-slope relationships in a particular basin. First, click on the left-most point that defines the trend of the small-basin area/slope relationship. Second, click on the point along the rollover where the channel concavity meets this trend. Finally, click on a third point to define the trend of the channel concavity. This will make a kinked line consisting of two straight segments. For your reference, each time you click, the program will plot the location of your click with a red cross. For example, the three clicks made in the plot above (again from left to right) are shown in the following figure:



Repeat this process for each of the basins you defined earlier. The function will return the concavities calculated for each basin and for all basins in the variables conc_df, conc_chan, refconc_df, refconc_chan, and will return a vector of numbers that denotes if a particular input (a,s) pair was identified as being to the right (2) or left (1) of the rollover:

```
>> conc_df

conc_df =

    0.2449    0.2412    0.2152    0.2930    0.3665    0.2557

>> conc_chan

conc_chan =

    0.6765    0.7483    0.8198    0.6926    0.5928    0.5920

>> refconc_df

refconc_df =

    0.2696

>> refconc_chan

refconc_chan =

    0.6722

>>
```

As discussed by Wobus et al. (2004), a reference concavity must be chosen to compare channel steepness values at points throughout multiple basins. In the next step, use the reference concavity with the calc_chansteepness command described below to calculate a steepness value for each point in the channel network.

To calculate steepness, I have provided you a command that will output a vector of steepness values as follows:

```
>> ks = calc_chansteepness(a,s,type,refconc_df,refconc_chan);
```

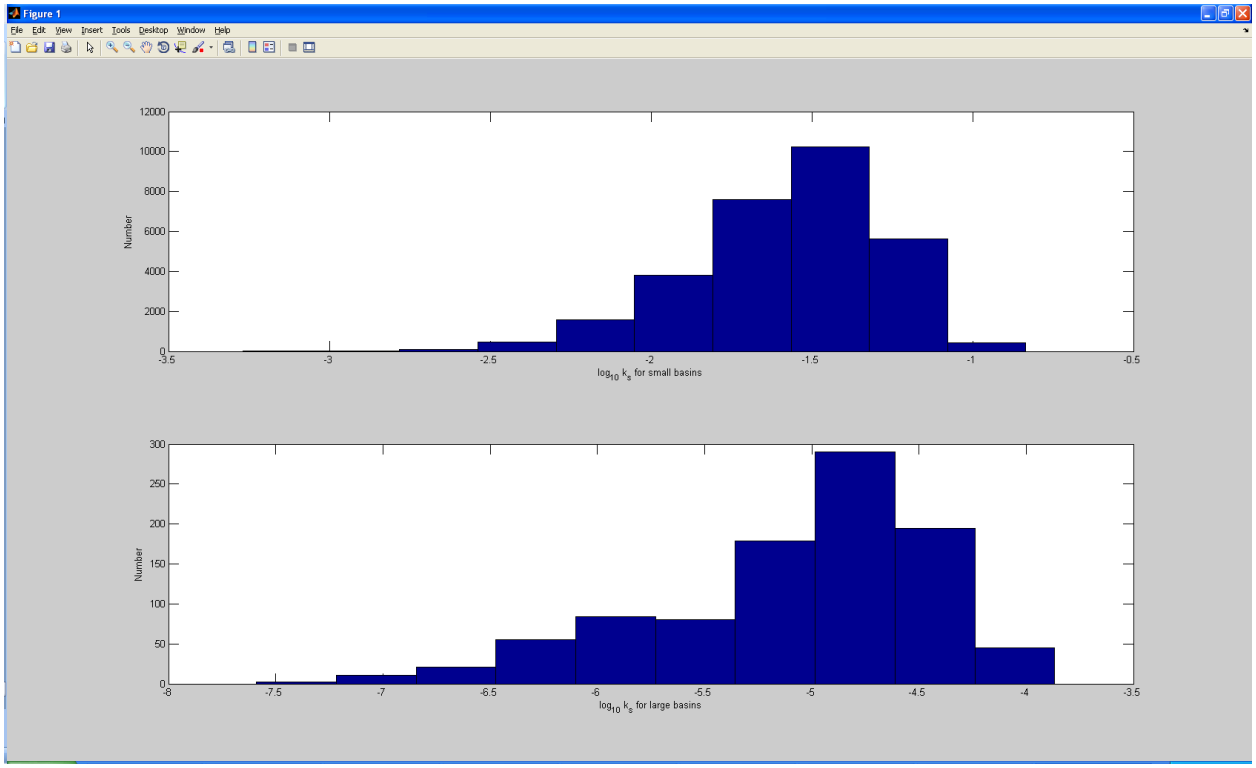
This will calculate the channel steepness at each (a,s) point, segregating steepness calculations by the location of each point relative to the rollover in log a-log s space.

Step 3: Map channel steepness values back onto the landscape and export these values to ArcGIS for further plotting and analysis in the next section.

In this next step, we will map a map of the steepness values across the selected basins. First, to see the relevant variation in the channel steepness values, use the following command:

```
>> plotkshistogram(ks,type)
```

This will produce a histogram of the log-transformed ks values that will allow you to define a range over which you wish to plot a map of the log-transformed values. For example, the above command will produce a window similar to this:

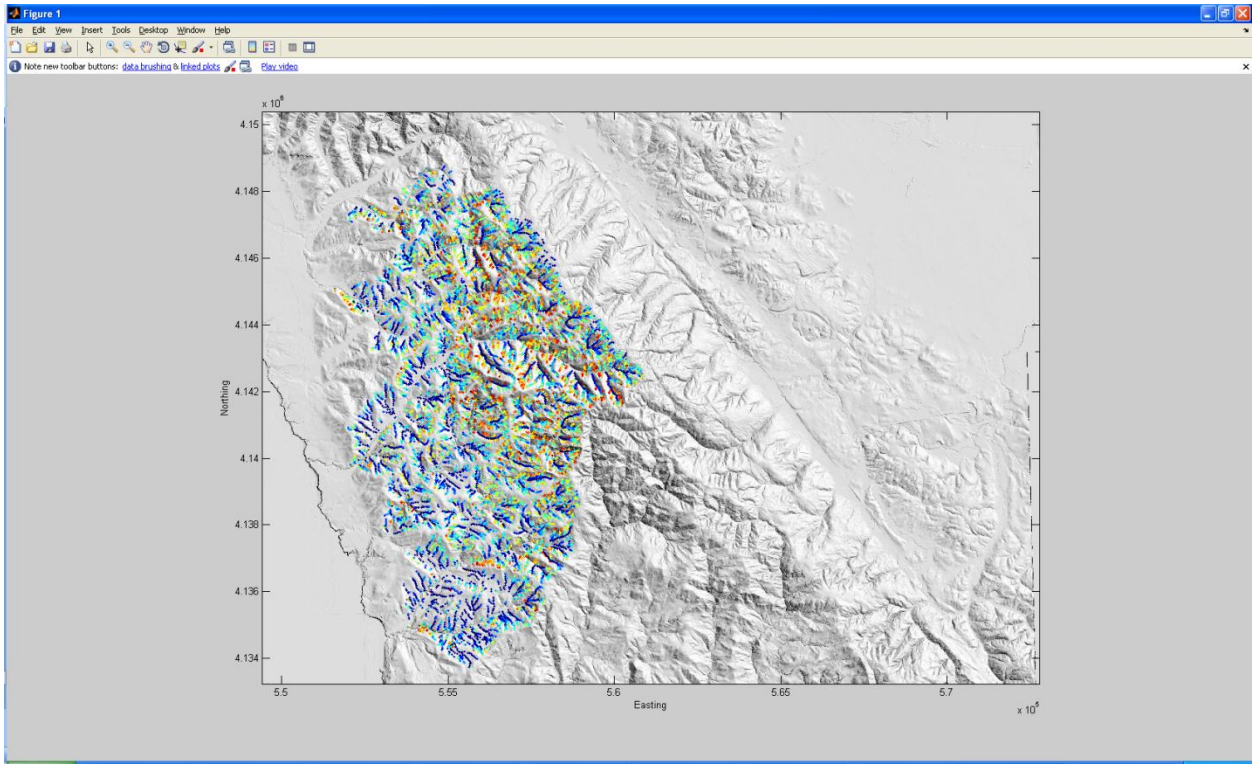


Here, you can see that most of the variation in log ks for the small basins occurs between values of -2 and -1. Likewise, most of the variation in log ks for the large basins occurs between values of -6 and -4.

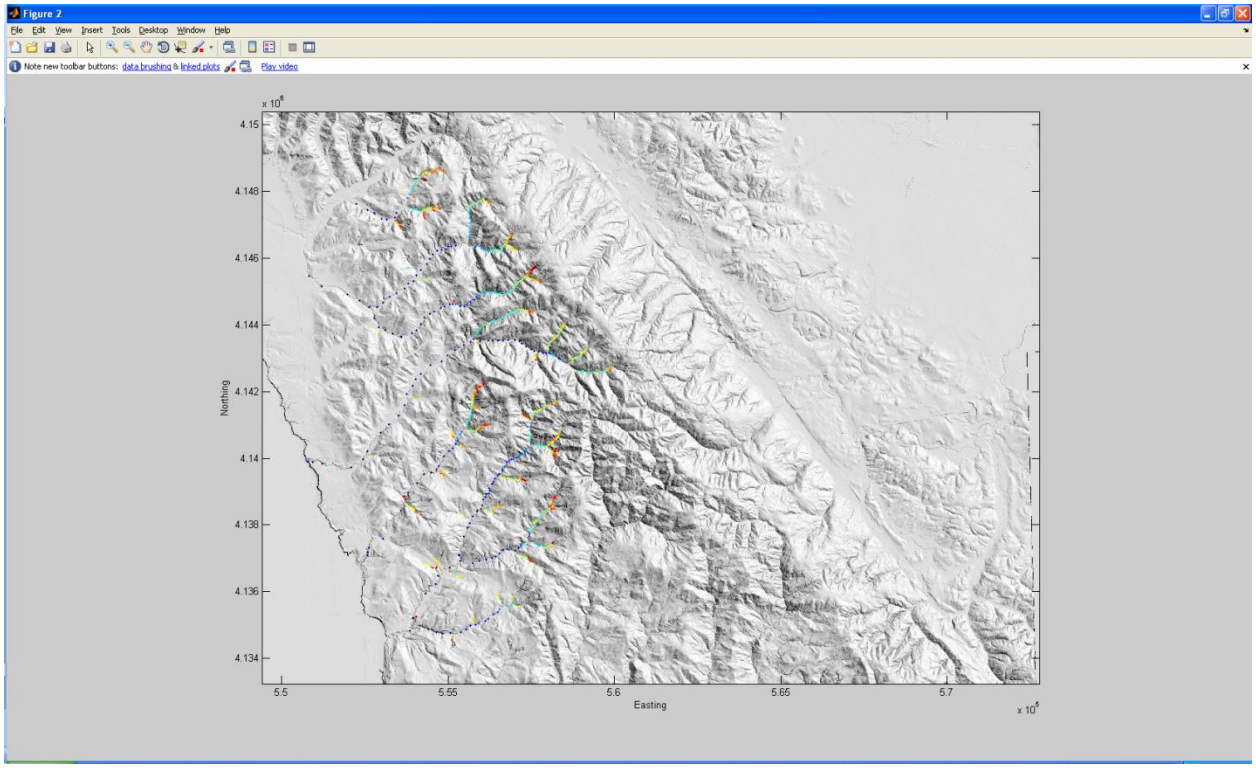
Next, to plot a color-coded plan-view map that shows the spatial distribution of steepness values, type the following command:

```
>> plotks(dem,ks,type,x,y,320,30,[-2 -1],[-6 -4]);
```

The last four arguments to this command (320, 30, [-2 -1],[-6 -4]) allow you to set the way in which the data are displayed on the map. The first of these arguments (320) is the azimuthal angle used to produce the shaded relief map onto which steepness values are plotted. Likewise, the second (30) is the elevation angle between the horizon and sun location. The next argument specifies the range of the linear ramp used when plotting the steepness values for the small basins. As seen above from the histograms, most of the values are contained within the range of -2 to -1, and so we use this argument to clip the data to lie between this range. Points whose steepness values exceed or are less than this range are set to the color that corresponds to the maximum and minimum value of the range, respectively. Finally, the last argument specifies the range of the linear ramp used when plotting the steepness values of the large basins in an identical manner to the small basins. This command will produce two maps: one shows the spatial distribution of channel steepness for the small basin points:



While the other shows the spatial distribution of the large basin points:



Finally, you can export the values of steepness for each of the two defined basin types as a comma-delimited file that can be read into ArcGIS as follows:

>> export_ksdata('tile2lowres',x,y,ks,type);

Where 'tile2lowres' is the prefix of the filename you would like to use to store your results. To this prefix, a suffix of "_smallbasins.txt" and "_largebasins.txt" will be appended. Your results are recorded in two separate files to segregate those points identified with low concavity and small basin area from those points thought to be part of the fluvial system. NOTE: values of ks exported in this process will be recorded as log-transformed values.

Step 4: Calculate mean channel steepness values in each basin and use this to create a basin map of mean channel steepness.

Next, we will create a map that shows the mean log-transformed channel steepness value in each of the basins that you selected. Because channel steepness has been calculated for both large and small basin areas, we will create two different maps that record the mean log-transformed value of ks for each of these basin classes. To do this, use the following LAME command at the Matlab prompt:

>> [meanlogks_smallbasins, meanlogks_largebasins] = map_meanlogks(fd,xo,yo,ks,b,type);

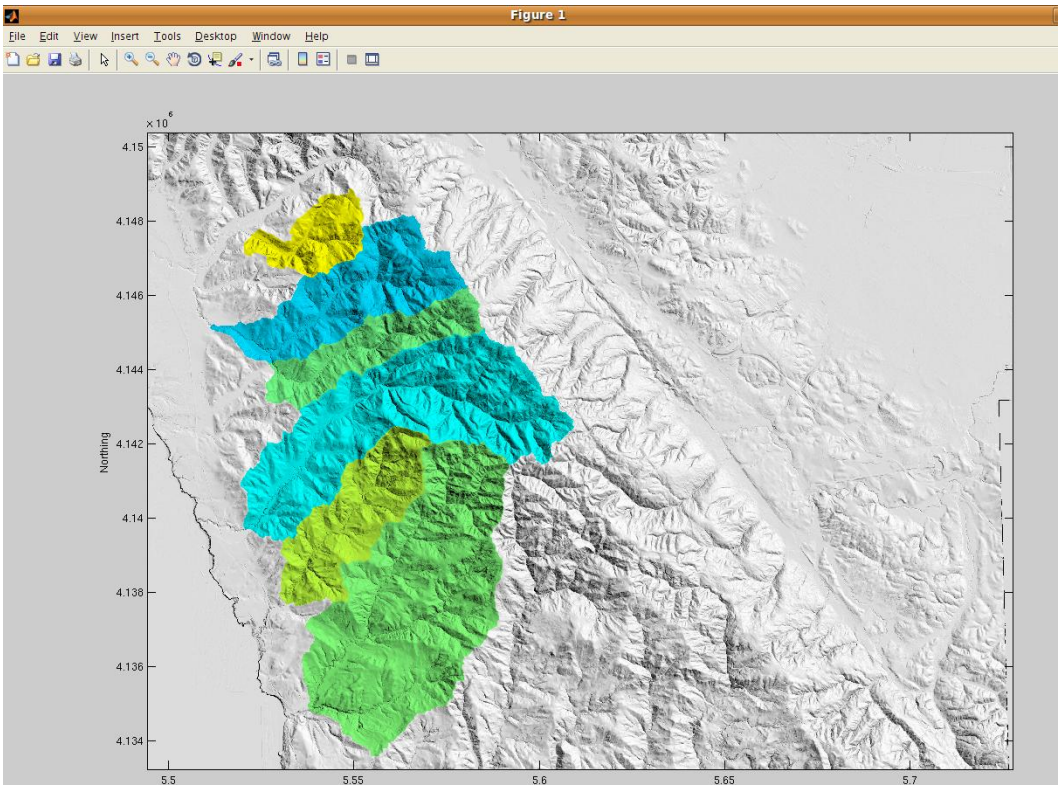
This will create two new grids, meanlogks_smallbasins and meanlogks_largebasins that will contain the mean log-transformed values of ks when considering only points from small basin areas and large basin areas, respectively. In these maps, the entire basin will assume the value of the mean log-transformed ks value.

Once created, these maps can be displayed using the following command:

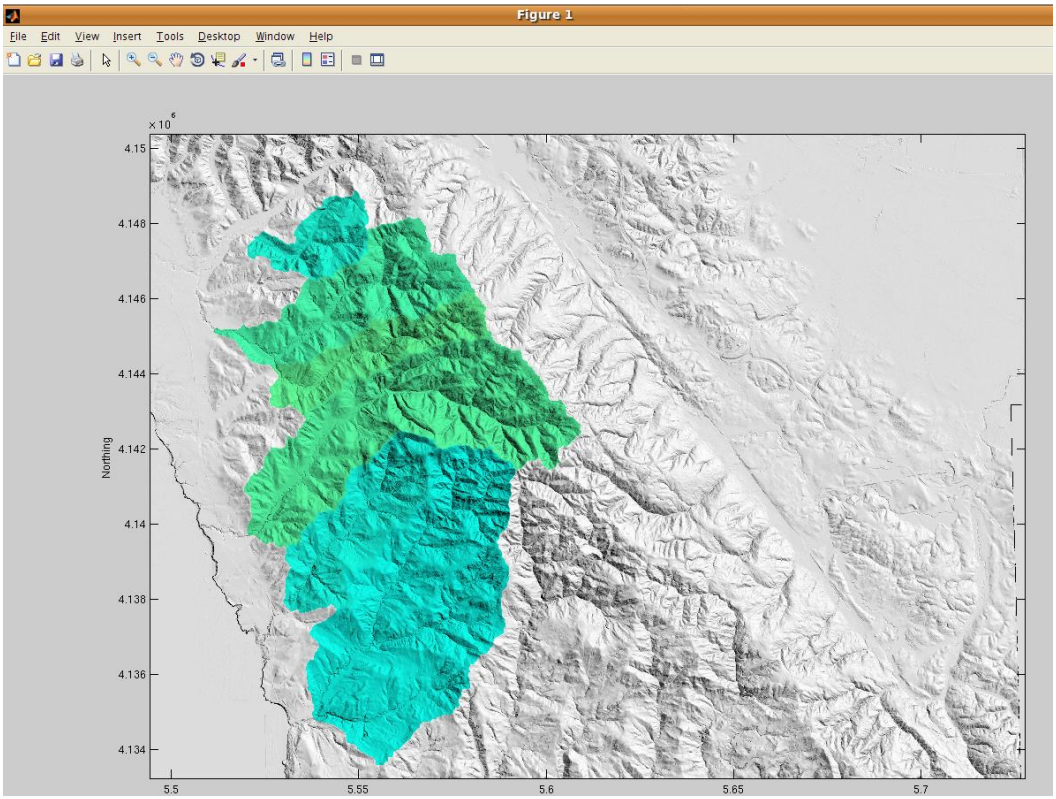
>> plotgridandhs_valuegrid(meanlogks_smallbasins,dem,320,30,-2,-1);

This command will create a shaded-relief map of the ALSM topography over which is draped the mean log-transformed value for ks for each basin:

For large basins:



And small basins:



Finally, you can export these two grids into a format that ArcGIS can read using the following command:

```
>> topoarcmatlab_writegrid(meanlogks_smallbasins,'meanlogks_smallbasins');

>> topoarcmatlab_writegrid(meanlogks_largebasins,'meanlogks_largebasins');
```

The first argument of this command passes the actual grid that you wish to write to a file to the function, while the second (in apostrophes) denotes the filename to which you wish to save the data. NOTE: This is a general command that can be used to export any grid that is derived from the analysis described above.

Step 5: Calculate mean slope angle and convexity for basins whose areas are < 1 km^2, but whose maximum basin area in total is > 0.5 km^2 that are nested within selected basins. Slope angles and convexity values will be computed for planar, concave, and convex portions of the landscape separately, in addition to mean values within these basins regardless of convergence/divergence along hillslopes.

The preceding exercise provided information about the relative steepness of different portions of the convergent areas of topography. In this section, we will use LAME to isolate points that lie within small basins to determine how various metrics of hillslope topography change across the study area. To make these calculations, we must first isolate all outlets of sub-basins whose basin areas are at least 0.5 km^2 and are no larger than 1 km^2. This range of basin areas serves as a compromise between basins whose areas are so small that they do not yield reliable estimates for values such as slope angle and concavity, and those that are large enough to be significantly impacted by the effects of channel processes. Then, for each of these sub-basins, points will be classified into convex, concave, and planform geometries—the mean value for each of these sub-basins will be computed according to these morphologic classifications. Finally, we will map the spatial distribution of convexity and slope angle across the landscape to discern spatial patterns and associations with factors such as the rate of erosion of the basin and underlying bedrock substrate.

Luckily for you, I have provided a matlab script and a MEX interface to Standard Option (SO)-LAME's analysis package to calculate grids of the mean hillslope angle within these different types of basins, as well as the mean concavity of these small basins. First, to calculate a grid whose values denote the mean slope value of each sub-basin for each morphology, use this command:

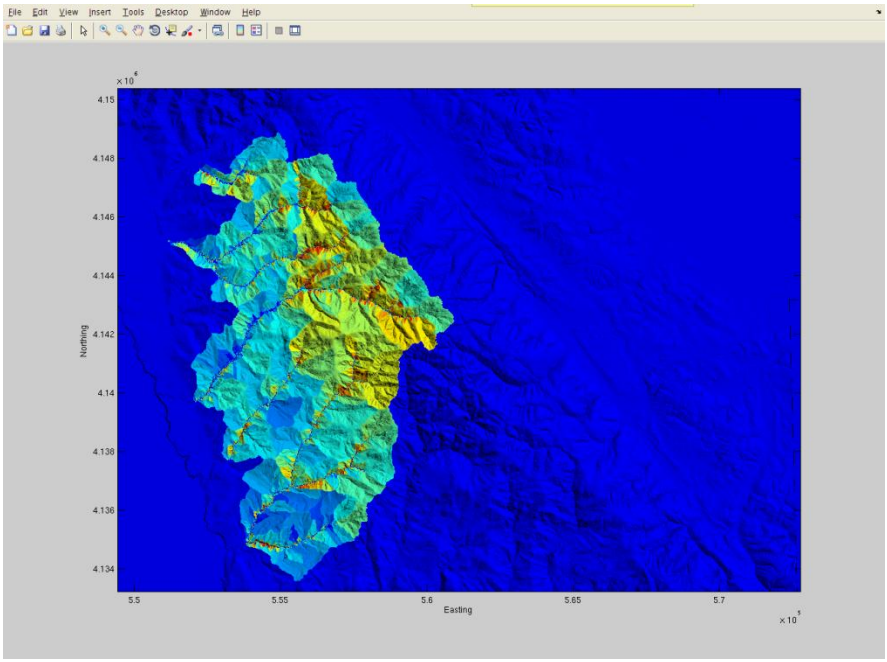
```
>> [meanslp_conc,meanslp_plan,meanslp_conv] = calculate_meanslope_hs(fd,area,dem,xo,yo,-
0.001,0.001,0.5,1);
```

Where fd, area, and dem are the flow, area, and DEM grids, respectively, xo and yo are the outlet locations you defined in Step 1 of Part 2, the second to last argument is the smallest basin size permitted in the analysis, and the last argument is the maximum basin size permitted when calculating mean slopes for the different morphologies.

The fourth- and third-to last arguments specify which portions of the landscape should be regarded as concave, planar, or convex. As noted by Dietrich et al. (1993), there is a somewhat arbitrary threshold that needs to be assigned to divide the landscape into these different morphologies, since even those morphologies that are almost perfectly planar will have some finite concavity due to either the nature of the surface or the way in which it was sampled by the ALSM instrument. The fourth-to-last parameter (-0.001 in the above example) defines the concavity below which hillslopes will be considered convex-up. Similarly, the third-to-last parameter defines the concavity above which hillslopes will be considered concave-up. Those points possessing concavity values between these two values will be defined as planar. NOTE: The concavities that are calculated are actually the Laplacian of the surface, rather than the true curvature tensor.

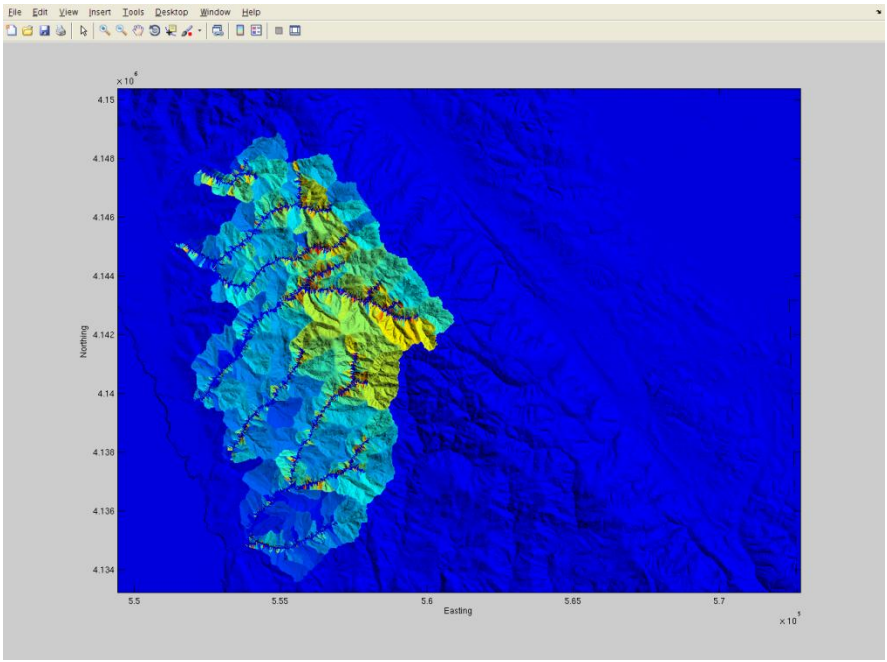
This operation can take some time to complete because of the deep recursions required to sample the many sub-basins that have small catchment areas. Once completed, the function will return three arguments, each of which is a grid whose values record the mean slope for each of the small sub-basins when points are segregated according to their morphology (convex, concave, planar). Thus, meanslp_conc will contain a grid that represents the spatial distribution of mean slopes within concave portions of these small catchments, meanslp_plan contains the distribution of mean slopes within planar portions of the sub-basins, and meanslp_conv contains the distribution of mean slopes within convex portions of the sub-basins. You can visualize these distributions using the plotgridandhs_valuegrid command as follows:

```
>> plotgridandhs_valuegrid(meanslp_conc,dem,320,30,0,1);
```



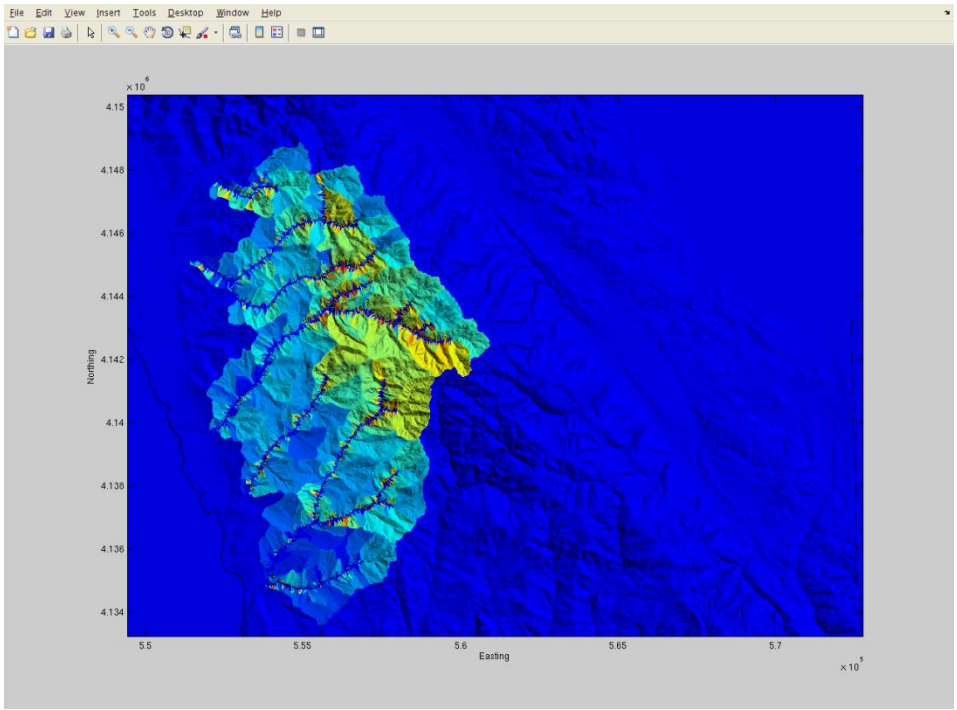
Likewise:

```
>> plotgridandhs_valuegrid(meanslp_plan,dem,320,30,0,1);
```



And:

>> plotgridandhs_valuegrid(meanslp_conv,dem,320,30,0,1);



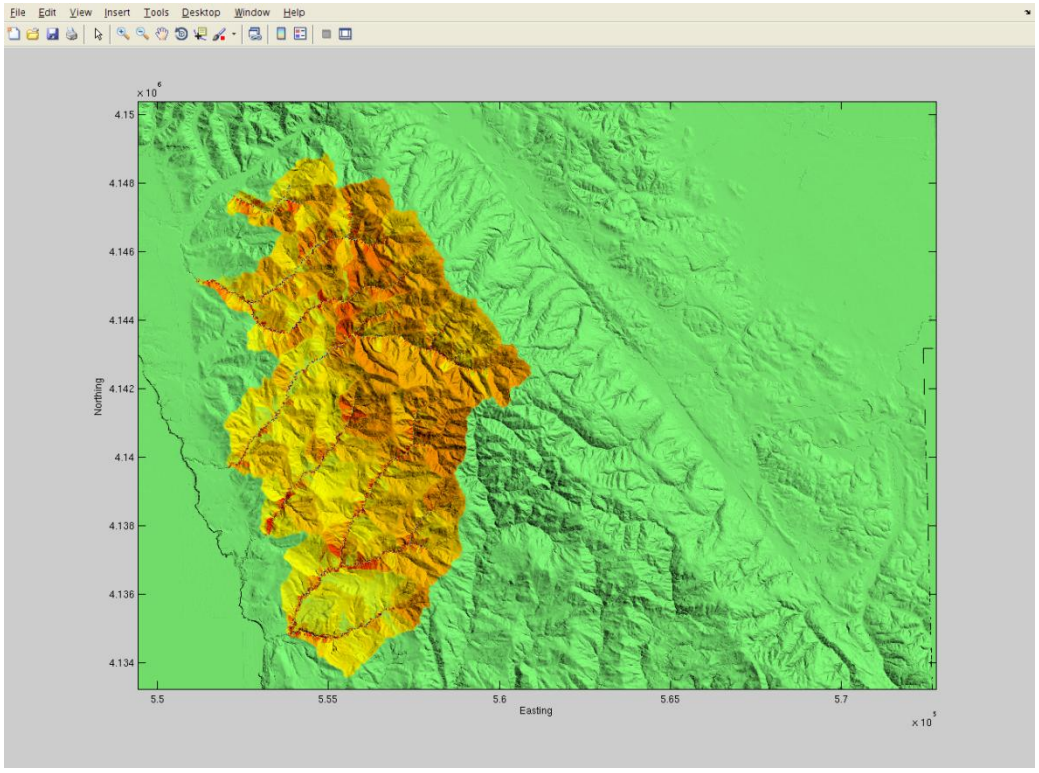
These grids can be written in Arc/INFO ASCII format using the topoarcmatlab_writegrid command described above. Go ahead and do this now so that we can later compare the spatial distribution of these values with geologic and tectonic factors in ArcGIS.

Finally, you will compute the mean curvature for different portions of these sub-basins according to morphologic type. Especially in areas subjected to diffusive hillslope processes such as bioturbative creep where slopes are not too steep, we might expect the curvature to decrease (become more strongly negative) with increasing base-level lowering rate. We can calculate the mean curvature for each of these sub-basins using a similar command as above:

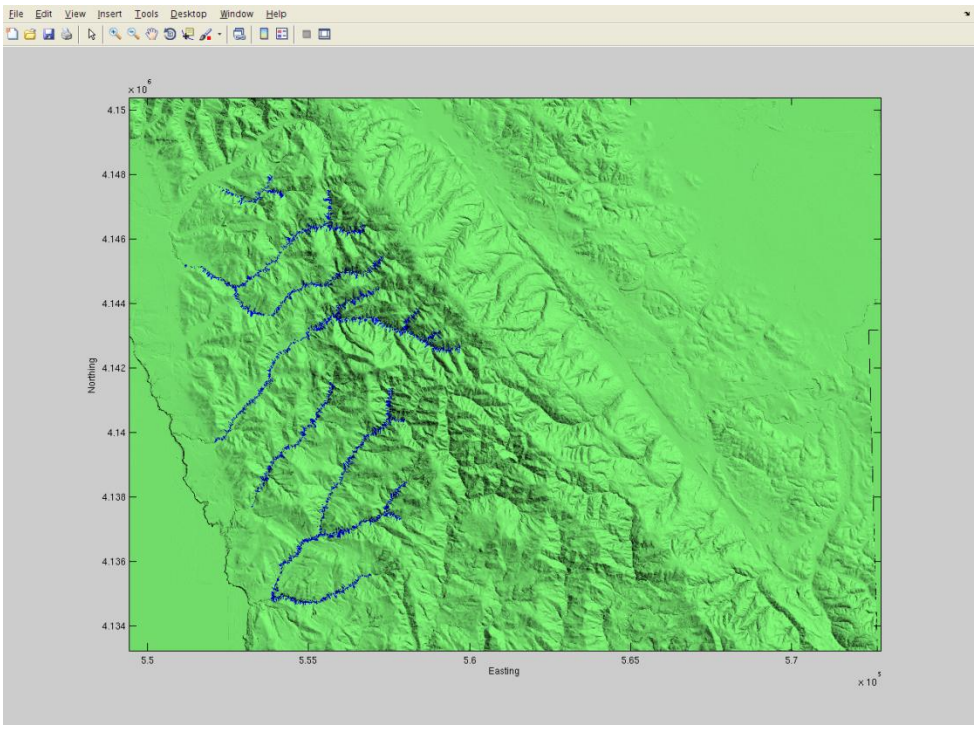
>> [meanconv_conc,meanconv_plan,meanconv_conv] = calculate_meanconv_hs(fd,area,dem,xo,yo,-0.001,0.001,0.5,1);

This returns the mean concavity values for concave, planar, and convex hillslopes in the way described above. You can plot the maps of these parameters using the plotgridandhs_valuegrid command:

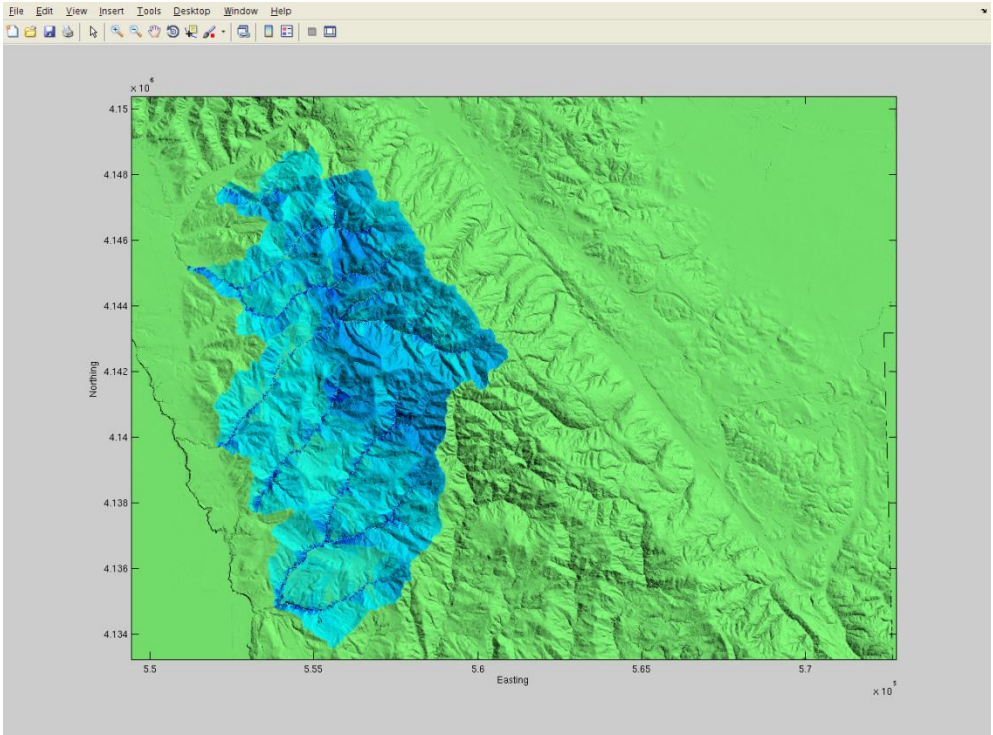
>> plotgridandhs_valuegrid(meanconv_conc,dem,320,30,0,1);



>> plotgridandhs_valuegrid(meanconv_plan,dem,320,30,0,1);



>> plotgridandhs_valuegrid(meanconv_conv,dem,320,30,0,1);



Finally, go ahead and save each of these grids in Arc/INFO ASCII format using the topoarcmatlab_writegrid command described above.

Congratulations, you have now finished the analysis portion of my LAME exercise. We will now plot these maps in Arc/INFO, along with other information that we have regarding the denudation rate, geologic substrate, and coastal rock uplift rates to interpret the form of this landscape in terms of these extrinsic factors.