Tools for Using Rasters For SMS

At the time of writing, performance of SMS rendering contours is terrible for datasets bigger than about 100Mb (assuming good RAM). There are also several SMS bugs that prevent SMS from georeferencing rasters correctly. SMS does not render DEMs when they are initially loaded and it assigns elevations from rasters that are inconsistent with its contours.

Visualizing Raster Contours

There are several options for obtaining contours quickly. Both require GDAL utilities and GDAL for python.

GDAL Contour utility

The first is the use of gdal tools out of the box. The gdal_contour utility will convert rasters into a correctly georeferenced contour shapefile which can be loaded as a GIS dataset. This is useful because the GIS feature can be automatically converted into a (fairly accurate) set of SMS arcs.

The second method is the script clip_dems.py, which uses gdal behind the scenes to clip a larger DEM to a smaller one. Type:

```
$ clip_dems.py --help
```

One argument to the script is a text file listing DEMs in order from highest priority to lowest. Numerous input formats are excepted, but tiffs < 200Mb are recommended over ascii for efficiency.

You must also identify the clipping coordinates. You can enter bounding coordinates explicitly, but the easiest way is by example. Zoom in SMS to the extent you want and "save as" a jpeg. Then use the switch "--image bay_delta.jpg". All the DEMs that "hit" the image will be clipped to the image bounds and saved in separate files named according to priority. The first file is called clipped_0.asc, clipped_1.asc etc, but you can change the prefix with the switch "--prefix suisun" to produce suisun_0. suisun_1.

Finally, the argument "--hshift" is needed to shift the DEM so that its contours will be correctly located. This argument is essential when working with contours in SMS.

Warning #1: If the script is laborious, you should consider abandoning the script and a smaller area. It won't render in SMS anyway. The renderable area in 2m is pretty small.

Warning #2: SMS cannot render a raster during the session in which it is loaded. Once the new DEMs are created, you have to load them, exit SMS and restart for them to render. This is a known/reported bug in SMS. Be aware that SMS will also "drop" any DEMs that are pure missing data. This is annoying if, like me, you just want to keep "clipped_0.asc" in your session and reload it with new contents

Warning #3: It will not be possible to visualize contours and rasters correctly in SMS until 11.1. Even then, there are no plans to fix the "inconsistent interpolation".

Consistently Populating a Mesh in SMS

Often, modelers digitize their SMS Map/mesh to conform to functionally important contours. Examples would be SMS "Map" elements that follow shores, inner channels and canyons as well as regions that are above/below the TVD depth threshold in SELFE. Unfortunately, SMS does not assign elevations from the DEM that conforms to its own contours -- the features can be displaced by roughly one DEM cell. SMS uses bilinear interpolation to locate contours and assign elevations from point datasets, but it uses piecewise flat values from DEMs.

In order to get around this inconsistency, the script stacked_dem_fill.py can be used to assign elevations using a bilinear interpolation that is consistent the SMS contours. The script maps 2dm to 2dm, so that the results can be seen in the SMS environment. This makes it very easy to visualize and correct small errors (overly high shore nodes, for instance) in one environment. See

```
$ stacked_dem_fill.py --help
```

for usage. The two main input parameters are the 2dm file and a text file (say, dem.txt) that lists DEMs in order from high priority to low. You should use the same list as you did for rendering the contours.