Chapter 3

fort.14 File - Domain Mesh

The foundation of an ADCIRC simulation is the model domain mesh. This chapter gives specific information about obtaining, formatting, and refining the data required for this mesh.

3.1 Obtaining Data

3.1.1 Coastline

In preparing the mesh for the domain, one of the key ingredients is a description of the coastline. For the present application, the Coastline Extractor program, provided by the National Geophysical Data Center, was used. This database may be accessed at http://rimmer.ngdc.noaa.gov/coast/. There, a user may identify a region of interest, using an interactive web-based program, and download, in a variety of formats, available coastline data for that region.

In the present project, data were downloaded in Matlab format. In this case, the data were given in two columns representing latitude / longitude pairs describing the coastline. Distinct segments of coastline, for example different islands, are identified in the Matlab format by beginning each segment with the line 'NaN NaN.'

During the extraction and downloading process, the user is given the option of creating a graphical plot of the accessed data. An example of this is provided in Fig. 3.1.

Note that other coastline data sources exist and may be of use. For exam-

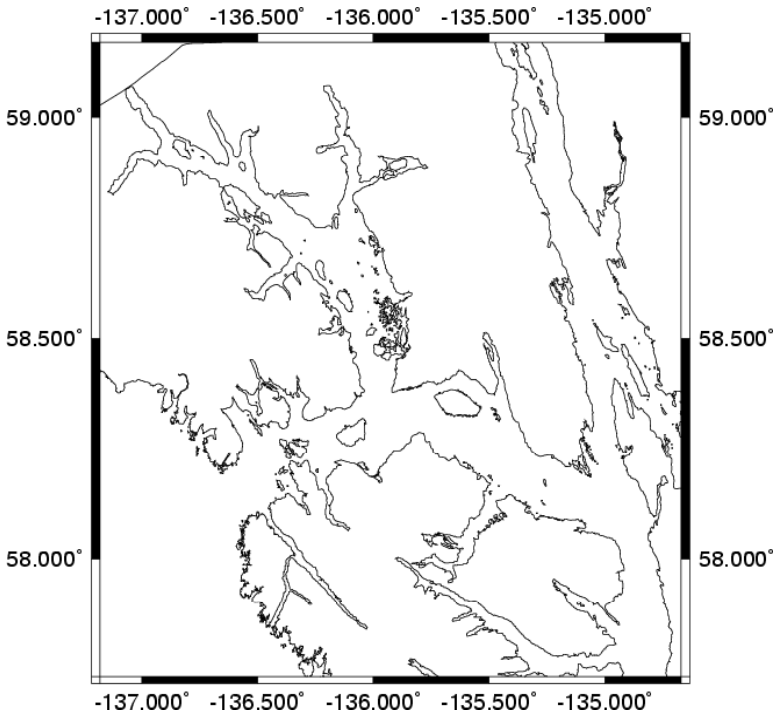


Figure 3.1: Graphical output fromt he Coastline Extractor program indicating the coastline data in the vicinity of Glacier Bay.

ple, inspection of Fig. 3.1 reveals that several of the inlets in the upper east arm of Glacier Bay appear to be truncated. This *may* be due to significant ice cover in those inlets during the time of those surveys; the exact reason is not known. Other data sources include the NOAA Shoreline Data Explorer Program, found at http://www.ngs.noaa.gov/newsys_ims/shoreline/. These additional data can be used to verify and / or fill in missing holes in the Coastline Extractor data.

3.1.2 Bathymetry Data

The other key ingredient in the construction of a finite-element mesh is the bathymetry, or depth. As with the coastline data, several excellent online sources of data exist. For example, the National Geophysical Data Center maintains a Geophysical Data System (GEODAS) that allows a user to interactively search for a wide variety of oceanographic data, including bathym-

etry. This database is found at http://www.ngdc.noaa.gov/mgg/gdas/gd_sys.html. As with the Coastline Extractor, a region of interest is identified and then the data may be downloaded in a variety of formats. Additional data can be obtained from National Ocean Service (NOS) maps and surveys at http://www.ngdc.noaa.gov/mgg/bathymetry/hydro.html.

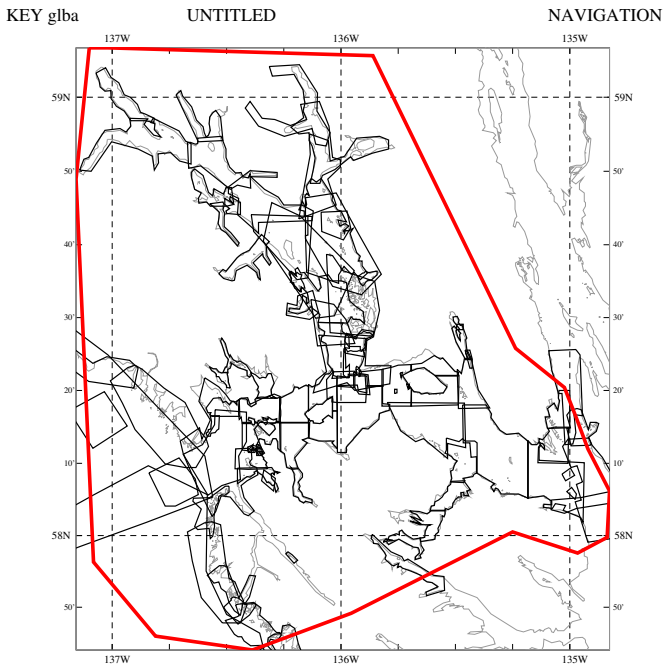


Figure 3.2: Graphical output from GEODAS indicating the bathymetry data obtained for the Glacier Bay region. The solid red line indicates the bounding polygon identified by the user. The light gray lines indicate coastlines and the heavier black lines indicate individual data sets.

Figure 3.2 shows an overview of the bathymetry data downloaded, using GEODAS, for the present application. The individual black polygons represent individual data sets from different cruises. For the present case, the data were downloaded in simple ASCII text format. In this case, each sounding is represented as a (lon, lat, depth) triplet of numbers. The datasets shown in Fig. 3.2 totalled some 800,000 soundings. A downsampled (by a factor of 20) scatter plot of the aggregated bathymetry is shown in Fig. 3.3. Here the red denotes shallow water and the blue deep water. We note in particular

the deep regions in the Alaskan Gulf, the Lynn Canal, and the upper west arm of Glacier Bay.

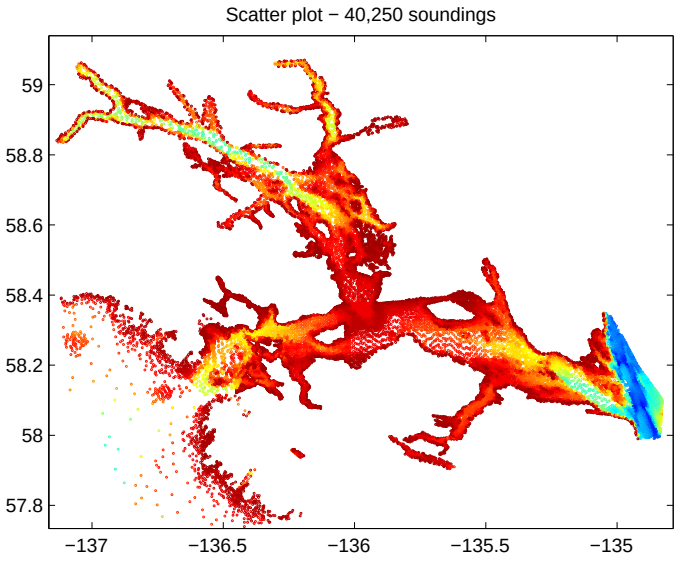


Figure 3.3: Scatter plot of domain bathymetry.

One cautionary note is illustrated in Fig. 3.4. Initial ADCIRC runs revealed unusual current patterns in the vicinity of Gustavus. A close inspection of the bathymetry data in that region revealed one survey (H08816) for which incorrect conversions appeared to have been applied. This was confirmed with NOAA and the problem was corrected both in their database and in the author's fort.14 file.

Finally, note that an inspection of the metadata from the surveys reveals that the depths are given relative to MLLW in most cases. A proper ADCIRC simulation requires depths relative to the geoid, which is well approximated by a datum such as NAVD88. The chief difficulty here is the lack of stations in Alaska where tidal datums are specified relative to a vertical datum like NAVD88. Lacking this, the adopted strategy has been to recast bathymetric depths relative to mean sea level (MSL). This was done by performing a long-term simulation with the raw bathymetric data, computing MSL and MLLW from the results, and adjusting the bathymetry. This process was iterated until satisfactory convergence was obtained.

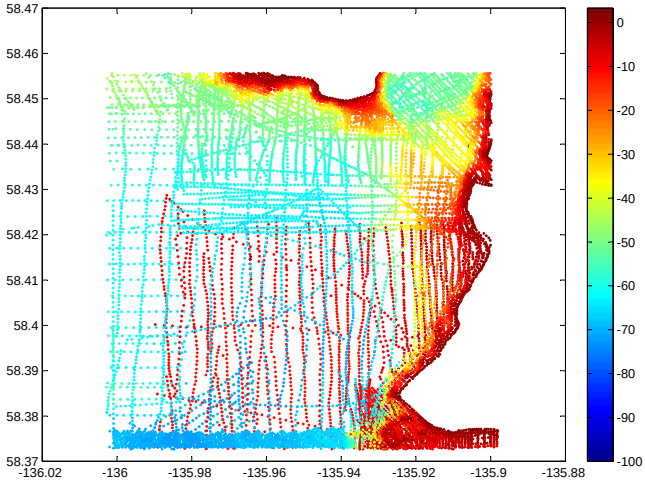


Figure 3.4: Incorrect data (red dots in the main channel) in the vicinity of Gustavus, AK.

3.2 Preparing Data

Once the necessary physical data has been assembled, the next step is conditioning this data and formatting it as required by the ADCIRC model.

3.2.1 Conditioning Coastline Data

The coastline data as obtained from the Shoreline Extractor program is not completely suitable. For example, it contains more of the coast along the Gulf of Alaska and the Lynn Canal boundaries than we require. In addition, the resolution is too fine in some spots. This, combined with the very large number of islands, many extremely small, proves prohibitive in terms of required mesh resolution. Therefore, a number of steps were taken to smooth, trim, and simplify the coastline data. These steps are enumerated below.

1. First, it was decided to run ADCIRC simulations using Cartesian coordinates, instead of latitude / longitude coordinates. This was done by using a UTM Zone 8 projected coordinate system. A free Matlab package, entitled m_map (http://www.eos.ubc.ca/ rich/map.html) makes this conversion quite straightforward.

2. Next, the raw coastline data file, as downloaded from the Shoreline Extractor, consisted of some 700 segments, some of which were mainland segments, and the bulk of which were islands. For the islands, a minimum area criterion of 2.5×10^5 m^2 was adopted, bringing the number of shoreline segments down to under 200.

3. Third, insufficient bathymetry data were found to exist in the areas of Lisianski Inlet and western Neka Bay. Therefore, the shoreline was manually edited in order to eliminate these regions from the domain.

4. Next, it was envisioned that the domain would be forced by two open boundaries, on at the west end of Cross Sound, and the other at the east end of Icy Strait. It is common, though not essential, to make open boundaries roughly semi-circular. To this end, the mainland shoreline segments along the Gulf of Alaska and Lynn Canal were 'trimmed' and then connected with semi-circles of evenly-distributed points. This is illustrated in Fig. 3.5.

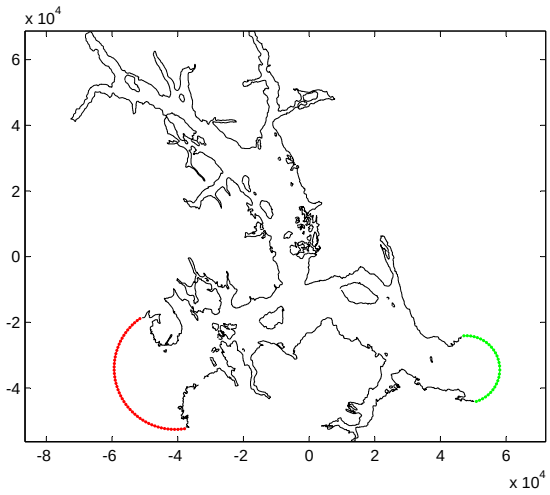


Figure 3.5: Reduced domain shoreline, with open boundaries placed at the Gulf of Alaska and the Lynn Canal.

5. Fifth, and as discussed previously, the shoreline data were inadequate in the Muir and Wachusett Inlets, artificially truncating those channels. Therefore, the shoreline data were manually edited so as to fully enclose the bathymetry data available in those locations.

6. Finally, the shoreline data, as conditioned to this point, are still irregularly spaced. Preliminary attempts at mesh generation revealed that this was very unsatisfactory, yielding very small and poorly formed elements in certain regions. As a result, the shoreline data were first evenly spaced along the shoreline, with a 400 m resolution, and then a moving window filter (in both the x and y directions) was applied to smooth the data. Figure 3.6 illustrates the before and after for a sample segment of shoreline. Clearly, this smoothing approximation results in some loss of accuracy, but this simplification is necessary if unrealistic computational demands are to be avoided. Note that many small islands do not survive this smoothing operation due to their small perimeter. The final conditioned domain shoreline therefore contains two open boundaries, two continuous mainland boundaries, and 55 islands.

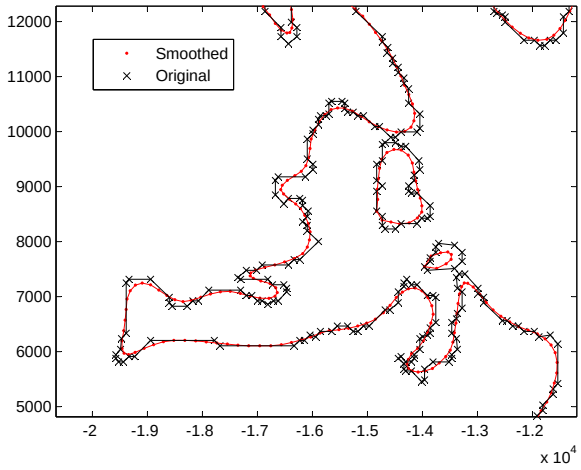


Figure 3.6: Illustration of original vs. smoothed shoreline data. Area shown is Berg Bay.

3.2.2 Conditioning Bathymetry Data

As mentioned previously, the aggregate bathymetry data from the GEODAS database consisted of some 800,000 soundings, which were downsampled to approximately 40,000. The only other required preparation to the bathym-

etry data was to convert the longitude / latitude coordinates to equivalent UTM Zone 8 projected coordinates.

3.3 Mesh Generation

Before writing the finite-element mesh information to the fort.14 file, the mesh itself must be generated. For the present project, this was accomplished using BATTRI. This is a freely-distributed (http://www-nml.dartmouth.edu/Software/battri/) graphical Matlab-based interface to Triangle. Triangle is a (C language) two-dimensional grid generator developed by Jonathan Shewchuk, presently at the University of California Berkeley. Triangle, and a complete user's manual, can be downloaded directly at http://www.cs.cmu.edu/afs/cs/project/quake/public/www/triangle.html. As a complete BATTRI manual is distributed with the software, only a brief summary of the key steps will be provided here.

3.3.1 Preliminary Steps

The use of BATTRI for mesh-generation has essentially two requirements. First, one must have bathymetry data in the form of (x, y, z) triplets, where z is the bathymetric depth. Second, one must have a .poly file, which describes the domain. More specifically, this ASCII text file describes the domain boundaries, in terms of listing the coordinates of the points on the boundaries and listing the 'edges,' in terms of describing how the boundary points link up to form continuous segments. In addition to the BATTRI manual mentioned above, a simple and useful tutorial is given by Edwards & Werner (2002).

In brief, the .poly file structure is given in Fig. 3.7. An extremely simple sample domain, containing one 'island,' and the corresponding .poly file are given in Fig. 3.8.

For a realistic application, like the present one, the boundary may be made up of thousands of points and there may be dozens to hundreds of islands. Manually assembling a .poly file in this case is not an option. Therefore, a Matlab script was written to automate this task. This script, entitled write_poly.m, and other scripts are distributed with this report and are fairly well documented with inline comments

When BATTRI is run, a number of initial choices will have to be made regarding the general display of data. Generally speaking, it is sufficient

The structure of a *poly* file is:

Line 1: nn nd natt nbm	nn:	total number of nodes (x,y pairs)
	nd:	# of dimensions (must be 2)
	natt:	# of attributes (0 in these examples)
	nbm:	# of boundary markers (0 in these examples)
Next *nn* lines: n x y	n:	node number
	x:	x-coordinate
	y:	y-coordinate
Next line: nedges nbm	nedges:	total # of land and open boundary element (node pairs)
	nbm:	# of boundary markers (0 in these examples)
Next *nedges* lines: edge na nb	edge:	number of the edge (or boundary line segment)
	na:	first node of edge
	nb:	second node of edge
Next line: nisl	nh:	number of islands (holes) in mesh
Next nh lines: isl xisl yisl	isl:	island (hole) number
	xisl:	x-coord of a point that lies within the island (hole)
	yisl:	y-coord of a point that lies within the island (hole)

Figure 3.7: Structure of the .poly file required by BATTRI.

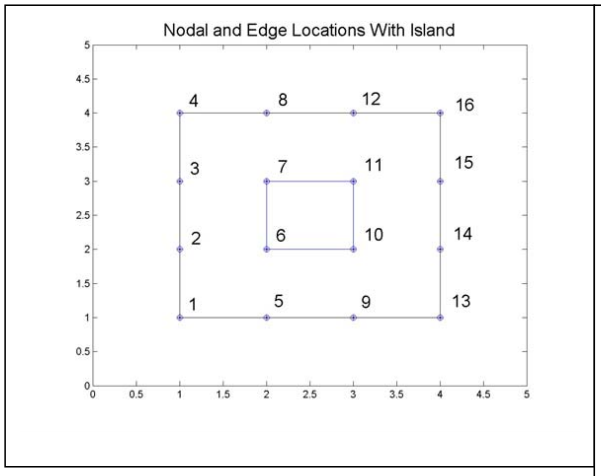
at this point to accept the default suggestions for the parameter choices. Additionally, the .poly file will be loaded and the boundaries will be plotted on the screen.

At this point, the user is allowed to do some 'editing' of the .poly file before the mesh is created. This editing includes features such as adding, deleting, or moving points, and adding, deleting, or dividing edges into smaller segments. These features are useful in continuing to smooth and improve the boundary. Additionally, the ability to add interior points is helpful in terms of avoiding finite elements that 'span' very narrow inlets. A general example of this ability to manually add points is given in Fig. 3.9.

For the present model application, the entire domain was reviewed very carefully in this step of the BATTRI execution and points were added and deleted as needed.

3.3.2 First Cut Mesh Generation

When the manual editing of the domain boundaries is complete, a 'first cut' mesh will be generated. It is at this point that BATTRI calls the Triangle




```
16 2 0 0
1   1.0 1.0
2   1.0 2.0
3   1.0 3.0
4   1.0 4.0
5   2.0 1.0
6   2.0 2.0
7   2.0 3.0
8   2.0 4.0
9   3.0 1.0
10  3.0 2.0
11  3.0 3.0
12  3.0 4.0
13  4.0 1.0
14  4.0 2.0
15  4.0 3.0
16  4.0 4.0
16  0
1    1   2
2    2   3
3    3   4
4    4   8
5    8  12
6   12  16
7   16  15
8   15  14
9   14  13
10  13   9
11   9   5
12   5   1
13   7   6
14   6  10
15  10  11
16  11   7
1
1  2.469 2.539
```

Figure 3.8: Sample domain and corresponding .poly file.

program (which runs in the background) and attempts to create a finite element mesh of the domain. The user is allowed to specify parameters such as the maximum number of nodes to add, the maximum (triangular) finite element area, the minimum interior angle allowed in each triangle, and so on. Additionally, the user can specify whether or not the boundary, as given in the .poly file can be altered / refined (by subdividing segments, for example) during the mesh generation process.

The author's experience was that significant trial and error can be expected at this step of the process. There is a fine balance to be struck since it is desired to have a mesh that is of high quality but not too large. If the former requirement is not met, the results of the hydrodynamic simulations

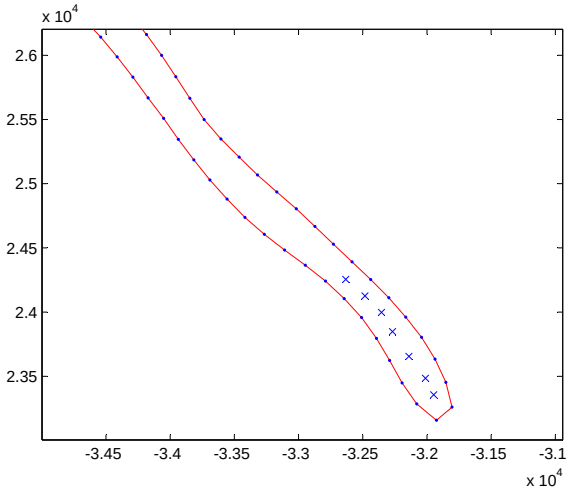


Figure 3.9: Intermediate plot output from BATTRI. The area shown is Charpentier Inlet; the blue dots and red lines indicate the boundary as specified by the .poly file, the blue x marks indicate user added interior points.

may be suspect. If the latter requirement is not met, the simulations may be prohibitively expensive in terms of computer time.

Figure 3.10 shows a representative example of what a typical 'first cut' mesh might look like. The area shown is Tarr Inlet in the West Arm. At this point, if the user is satisfied with the first cut, the bathymetry is interpolated onto the mesh. Note that this can be a very long step, depending upon the user's computer's processor speed and the size of the mesh.

3.3.3 Diagnostic Plotting

After the first cut, the user is allowed to make a number of 'diagnostic' plots. For example, the user may choose to plot the bathymetry as interpolated onto the mesh. Many of the other options have to do with subsequent mesh refinement options. For example, the user can plot the mesh element 'quality,' which is a measure of the shape of the triangles. The quality parameter ranges from a value of 0 for an extremely flat (approaching a line) triangle to a value of 1 for an equilateral triangle. Based upon experience, it is generally accepted that triangles with a quality measure less than 0.6 will cause problems with the numerical calculations.

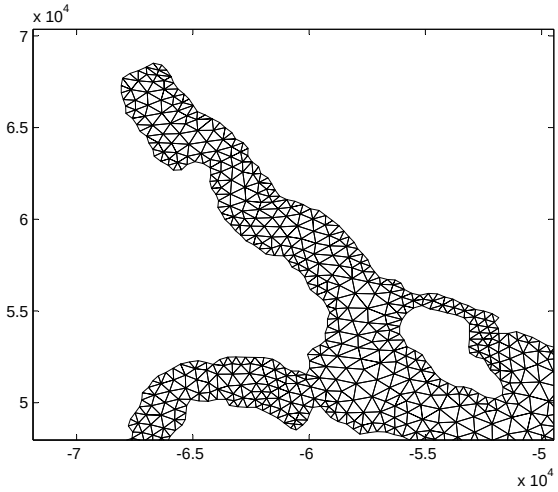


Figure 3.10: Sample mesh, as generated by BATTRI, for Tarr Inlet.

As another example, the user can plot the ratio $(\Delta h)/h$, where Δh is equal to the difference between the maximum depth of an element and the minimum depth and h is the average depth. Elements with a high $(\Delta h)/h$ value indicate regions of sharp bathymetric gradients that will need to be more finely resolved.

3.3.4 Mesh Refinement

After a first cut mesh has been generated, the diagnostic plots discussed above are helpful in determining how the mesh should be refined. The first cut is purely a two-dimensional exercise based only upon the domain boundaries; no consideration to the bathymetry is given. However, there are two very important ways in which the mesh must be refined.

1. First, there is the wavelength to grid size ratio, given by

$$\frac{\lambda}{\Delta x} = \frac{\sqrt{gh}}{\Delta x} T,$$

where λ is the tidal wavelength, Δx a measure of the linear dimension of a given element, h the mean depth of the element, and T the tidal wave period. To properly resolve the shape of the wave, it is generally sought to keep this ratio greater than 100. As tidal wavelength decreases in

shallower water, Δx must decrease as well. Therefore, this criterion has the effect of using smaller elements in shallow water and larger elements in deeper water.

2. Second, there is a criterion known as the topographic length scale criterion. This is given by the ratio

$$\frac{\Delta h}{h} = \alpha.$$

Here, Δh is maximum depth of an element minus the minimum depth and h is the mean depth. Thus, this criterion addresses the bathymetric slope, and not just the local depth. It is desired to keep α less than or equal to one.

When refining the mesh with both of these criteria, one must balance the desire to fully meet the criteria with the desire to keep element sizes from becoming too small. Thus, both criteria are generally imposed along with a minimum area criterion.

As a final step in the mesh refinement process, a 'springs relaxation' operation can be performed. Essentially, this process smooths the grid and greatly improves the element quality values.

3.3.5 Refinement in xmGredit

For the most part, the mesh generation and refinement tools available in BATTRI suffice. What BATTRI lacks, however, is the ability to go in and edit individual nodes and / or elements after the mesh has been generated. Occasionally, there will be a few problematic elements of low quality that need this type of manual editing.

As a result, the author found the program xmGredit to be extremely helpful in the preparation of the final Glacier Bay mesh. This program, available at http://www.ccalmr.ogi.edu/CORIE/software/, is a Linux / Unix based program that allows the user to create and edit two-dimensional meshes. Note that only the source code is available at the above site; interested users will have to compile this source into an executable on their own machines. A user's manual for xmGredit is available and is quite helpful.

It is fairly straightforward to take BATTRI output and route it into xmGredit for refinement. Using xmGredit has the auxiliary benefit that it

allows for the easy construction of fort.14 files, which is the goal of this section. To do this, an ASCII 'grid' file must be prepared. As described in the manual, this file simply describes the nodes and the elements of the mesh. For example, a very simple grid file might look like

```
This is a file identifer
2 4 # number of elements and number of nodes
1 0.0 0.0 1.0 # node number, x, y, depth
2 1.0 0.0 9.0 # node number, x, y, depth
3 1.0 1.0 3.0 # node number, x, y, depth
4 0.0 1.0 2.0 # node number, x, y, depth
1 3 1 2 4 # element number, number of nodes, node list
2 3 2 3 4 # element number, number of nodes, node list
```

Note that the # symbol is a comment symbol and that it and the trailing comments are optional. This sample mesh has four nodes and two elements.

To proceed, note that the final step of BATTRI creates three files of interest:

1. filename.nod - a file that describes the x and y locations of the nodes,
2. filename.bat - a file that gives a depth value for each node, and
3. filename.ele - a file that gives the node numbers (3) making up each element.

The Matlab script convert_gredit.m takes these three files and combines the information into a single .grd file, named for example grid.grd, for use with xmGredit.

Once in xmGredit, the open and mainland boundary segments must be identified and exported, say to a file named boundary.txt. If any changes are made to the mesh, it must exported, say to a file named newgrid.grd.

3.4 Final Mesh Characteristics

The final mesh for the present study of Glacier Bay is shown in Fig. 3.11, although the density of element makes it difficult to view. The final mesh has 48,144 nodes and 88,404 elements. These numbers are high given the relatively small domain and there are two reasons for this. First, the bathymetry

and the shoreline are both highly variable. Second, it was decided that a somewhat slow but accurate model was preferable to one that was fast but less accurate. Thus, the author retained as much bathymetric and coastline detail as was practical.

Figures 3.12-3.13 derive from a post-grid-generation Matlab script that was developed in order to assess the quality of the grid. In the first figure, histograms of grid size characteristics, element qualities, and time-step constraints are provided. The lattermost of these will be discussed in detail in the next chapter. Regarding the quality, note that only one element has a quality measure less than the desired 0.6 and that the vast majority of elements have qualities close to 1.0.

In the second figure, we find that the wavelength to gridsize ratio is satisfied everywhere. The TLS criterion proves to be more challenging, with many elements having $\alpha > 1$. These elements are generally found in very shallow waters and were prevented from being further refined by the minimum area criterion described above.

3.5 Writing fort.14 File

The final step is to prepare the actual fort.14 file itself. A complete description of this input file structure is given in the ADCIRC manual, which may be accessed at http://www.adcirc.org. In general, ADCIRC is highly sophisticated, allowing for a wide variety of boundary types. For example, there are open boundaries, where the tidal elevation is specified and which drive the rest of the domain. There are also no-flow boundaries, i.e. land segments. Additionally, boundary segments where the flow is specified (i.e. a river discharge) are allowed. Finally, internal and external barrier boundaries, where the normal flow is zero unless the elevation exceeds a critical value (the height of the barrier), are allowed.

3.5.1 xmGredit

For the present application, if no river inflows are considered, we have only mainland and open boundaries to contend with. In this case, xmGredit can be used to immediately create the fort.14 file. As described in the previous section, the boundary information is exported to boundary.txt and the grid information is exported to newgrid.grd. The command

```
>>cat newgrid.grd boundary.txt > fort.14
```

issued at the Linux prompt concatenates the node / element information with the descriptions of the boundaries.

If prescribed (non-zero) normal flow boundary segments (i.e. river discharges) are to be included in the ADCIRC simulation, this represents a fairly simple extension to implement. Note, first of all, that the fort.14 file structure contains, after the nodal, element, and open boundary information, the following:

```
NBOU NVEL for k=1 to NBOU
  NVELL(k), IBTYPE(k)
  for j=1,NVELL(k)
    NBVV(k,j) include if IBTYPE(k) = 0,1,2,10,11,12,20,21,22,30
    NBVV(k,j), BARLANHT(k,j), BARLANCFSP(k,j)  include if
        IBTYPE(k) = 3,13,23
    NBVV(k,j), IBCONN(k,j), BARINHT(k,j), BARINCFSB(k,j),
        BARINCFSP(k,j)  include if IBTYPE(k) = 4,24
  end j loop
end k loop
```

This section of the fort.14 file describes the boundary of the domain excluding the open boundaries. IBTYPE is a parameter that describes the boundary segment 'type.' For example, 0 refers to a no-flow external boundary, such as a piece of coastline. An IBTYPE of 1 refers to a no-flow internal boundary, or island. An IBTYPE of 2 refers to an external boundary with non-zero normal flow. Although xmGredit does not explicitly handle river discharge boundaries, they can therefore be implemented as follows:

1. In xmGredit, subdivide the external boundaries into coastline segments and river discharge segments. Then, export the boundary information and create the fort.14 file as described above.

2. Using a text editor, find the river discharge segments in the file and change the IBTYPE from 0 to 2.

3. Create, as per the ADCIRC manual and as described in Chapter 5, a fort.20 file which describes the river discharge into the domain.

3.5.2 Matlab

Another option is to prepare the fort.14 file using a Matlab script written by the present author, called write_fort14.m. There are some important points to note. First of all, the code requires the final .node, .ele, and .poly files generated from BATTRI *prior* to the optimization. As the user progresses through the mesh refinement in BATTRI, files such as filename_1.node (.poly, .ele), filename_2.node (.poly, .ele), and so on, are created. The group of three files with the highest number contain the exact same information as the (.nod, .ele, and .bat) files generated by the final optimization; the only difference is in how the nodes and elements are numbered.

Also, the text files containing the open boundary information are required. When the program is executed, the user will be prompted to provide information about the various land and open boundaries. Note that ADCIRC requires the nodes of land segments to be ordered with the 'land on the right.' This means that mainland segments are specified in a counterclockwise direction while island segments are specified in a clockwise direction. The script asks the user for help in assembling these segments in the proper order.

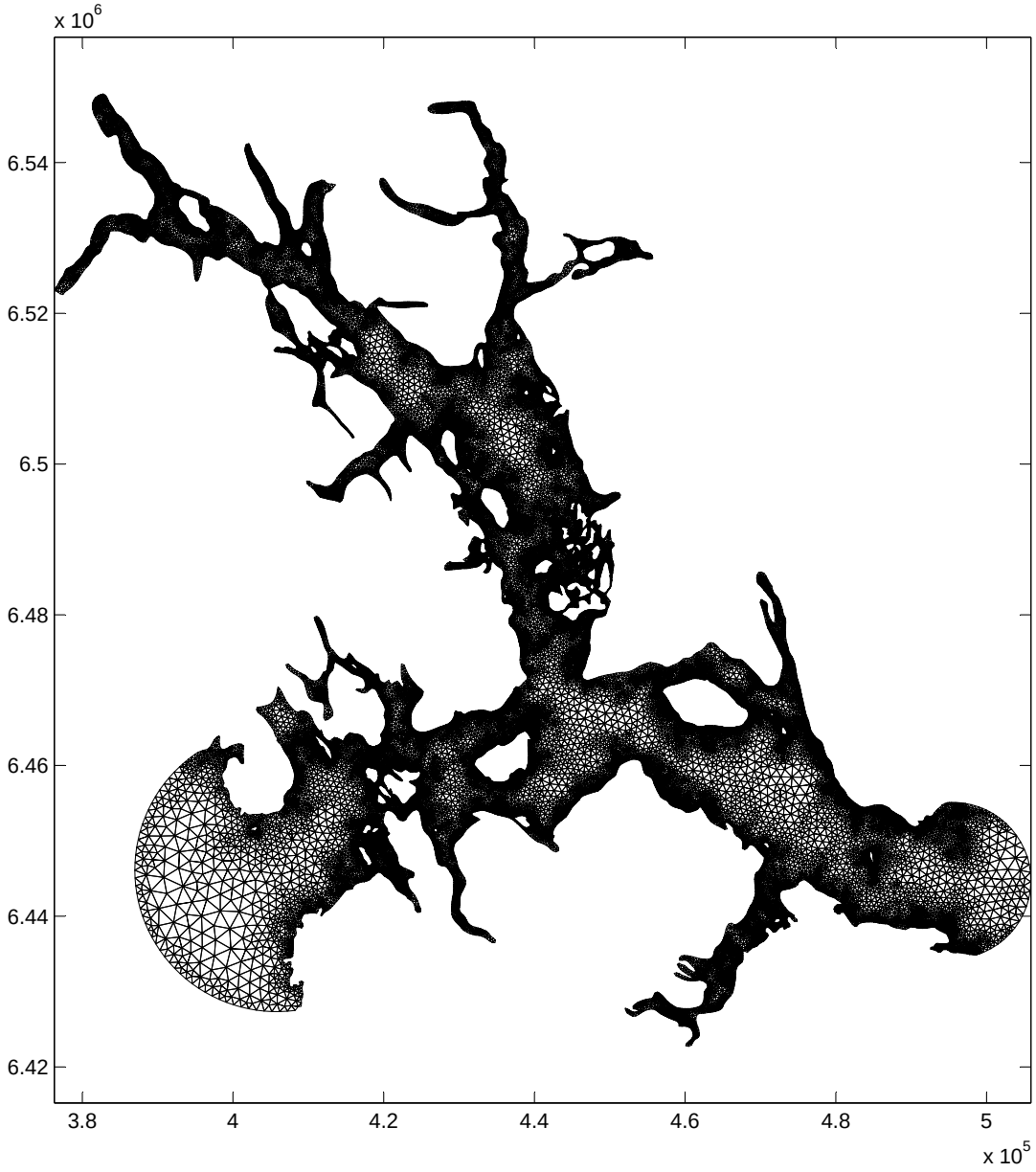


Figure 3.11: Refined (final) mesh for Glacier Bay and Icy Strait / Cross Sound.

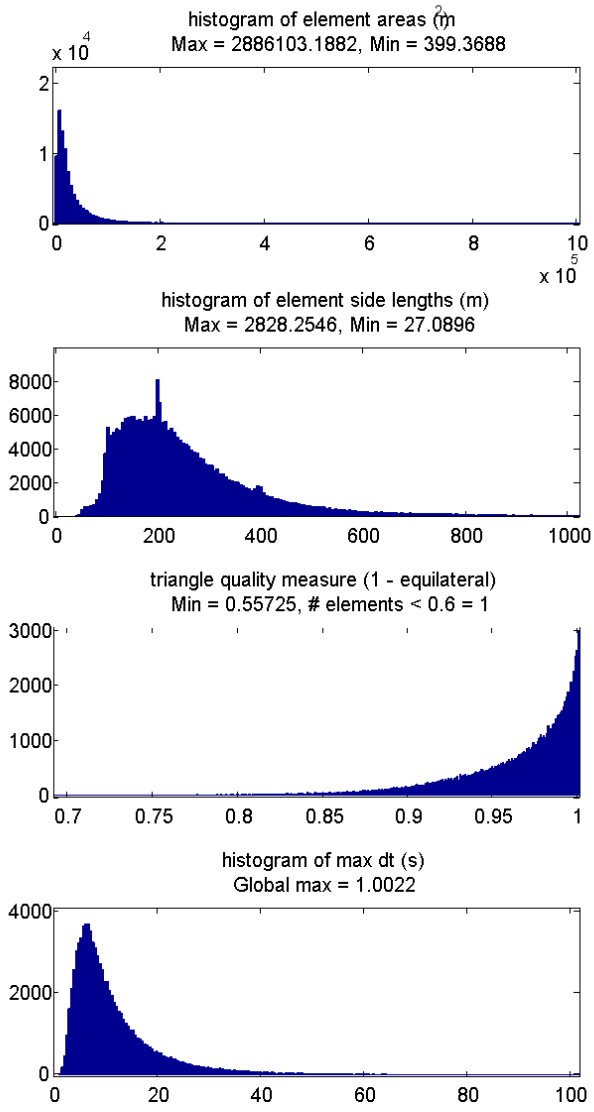


Figure 3.12: Histograms of grid element parameters.

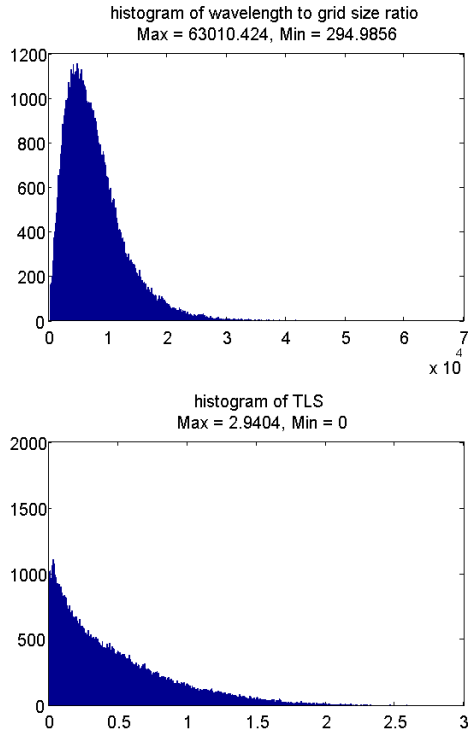


Figure 3.13: Histograms of grid element parameters.