Instructions for Downloading, Compiling, and Running MeshGenC++ by M. Elsey and J.A. Rossmanith

DISCLAIMER:


```
// Copyright by the authors
//
// This software is made available for research and instructional use only.
// You may copy and use this software without charge for these non-commercial
// purposes, provided that the copyright notice and associated text is
// reproduced on all copies.  For all other uses (including distribution of
// modified versions), please contact the author at the address given below.
//
// *** This software is made available "as is" without any assurance that it
// *** will work for your purposes.  The software may in fact have defects, so
// *** use the software at your own risk.
```


Step 1. Download MeshGenC++ source code:

https://sourceforge.net/apps/trac/meshgencpp/

Step 2. Download QHULL source code:

http://www.qhull.org/download/

Step 3. Configure and compile QHULL:

- Move into main QHULL directory: `cd qhull-2010.1`
- Configure by typing: `./configure`
- Compile by first typing: `make`
- And then: `make install`
- Create a path to QHULL. For example in bash you would modify your `.bashrc` file by adding a line of the following form:

  export QHULL = /XXXX/YYYYY/qhull − 2010.1

  where XXXX and YYYY need to be replaced by the correct sub-directories.

Step 4. Compile and run one of the examples:

- Change directories to one of the examples. For example type:

  `cd /XXXX/YYYY/MeshGenC++/2d/examples/square_circ/`

  where XXXX and YYYY need to be replaced by the correct sub-directories.
- Compile by typing: `make`
- Run by typing: `./mesh2d.exe` (or `./mesh3d.exe` for 3d example)
- After completion the code will create three output files:

  `nodes.dat` – list of grid points in the form (x, y, C) in 2d and (x, y, z, C) in 3d, where x, y, and z are the coordinates of the grid points and C is the area/volume associated with each dual cell centered at that grid point.

`elements.dat` – a list of pointers for each grid element in the form (i, j, k, A) in 2d and (i, j, k, ℓ, A) in 3d, where i, j, k, and ℓ are pointers to grid points and A is the area/volume associated with each grid element.

`boundary.dat` – a list of pointers to each node that sits on the domain boundary.

Step 5. Plot the results.

For each example a MATLAB script entitled `plot_grid.m` is provided. If the software package MATLAB (`http://www.mathworks.com`) is available to you, open it up and move into the correct `MeshGenC++` example directory. Execute the plotting script by typing: `plot_grid`

Step 6. Modify input files. There are two input files that need to be modified for your specific geometry:

`input.data` – Set the following parameters

- `Max Iterations` – maximum number of iterations allowed by algorithm
- `H0` – grid spacing for the initial distribution of points
- `Bounding box (xmin ymin xmax ymax)` – set a rectangular region that encloses your specific geometry (the initial point distribution will be laid down in this rectangular region)
- `Number of fixed points` – set the number of points that will remain in the grid, no matter how the other points are laid down
- `Fixed points (x y)` – give the (x, y) coordinates of the points which are to remain fixed in the grid, no matter how the other points are laid down

`input2D.cpp` (or `input3D.cpp`) – Set the following functions

- `double d(point pt)` – a signed distance function that is zero on the boundary, negative in the physical domain, and positive outside the physical domain.
- `double h(point pt)` – a function that sets the desired grid point spacing as a function of space. If this function is simply set to a constant, the code will try to produce a grid that is nearly uniform.

NOTE: if the file `input.cpp` is modified, you will need to re-compile by typing: `make`

For more information we refer you to the `DistMesh -- A Simple Mesh Generator in` MATLAB webpage of Per-Olof Persson:

`http://www-math.mit.edu/~persson/mesh/`

and the SIAM Review paper by Per-Olof Persson and Gilbert Strang:

P.-O. Persson, G. Strang, A Simple Mesh Generator in MATLAB, *SIAM Review*, Volume **46** (2), pp. 329-345, June 2004.