Project 5: 2D Mesh Generation

AE 623, Computational Fluid Dynamics II, Winter 2020

Due: March 20, 11:55pm, electronically via Canvas

1 Problem Description

In this project you will generate unstructured, triangular meshes for a two-dimensional miniature "city" geometry, consisting of four buildings in a rectangular, wall-enclosed domain. The setup and a sample mesh are shown in Figure 1. Buildings B1, B2, B3 are rectangles, whereas B4 is an octagon (not equal-sided) obtained by clipping corners from a $.15 \times .15$ square. All length units are meters.

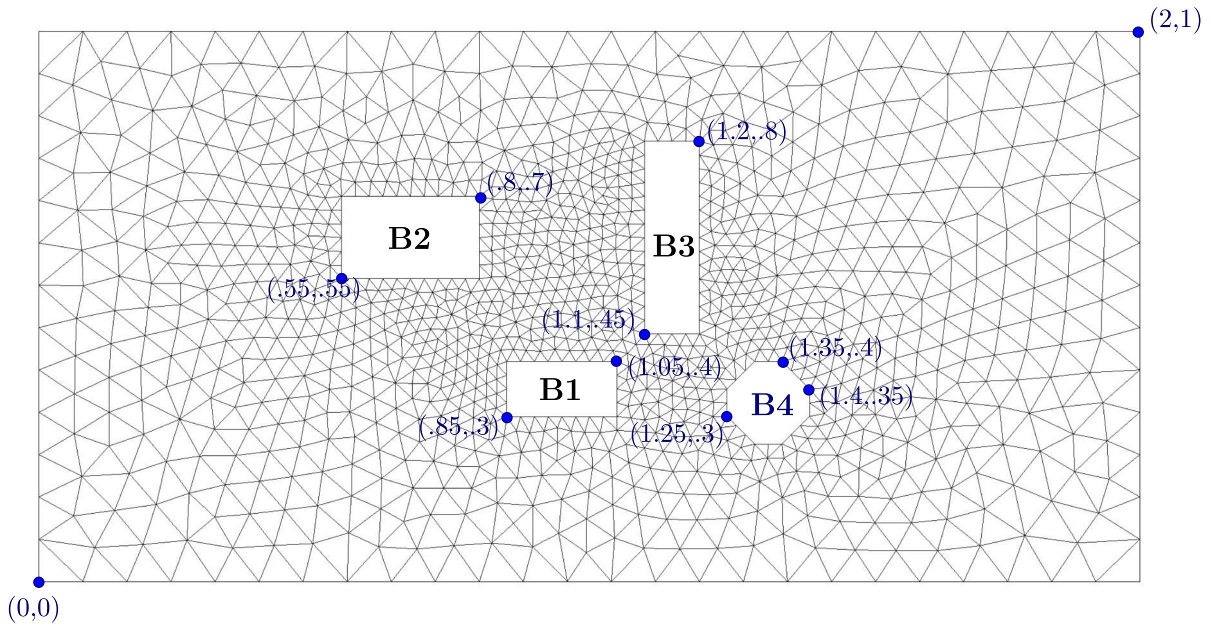


Figure 1: Problem setup and a sample coarse mesh of about 2000 triangles.

2 The `.gri` Mesh File Format

You will make multiple meshes of increasing refinement, and you will use the `.gri` format to store your meshes. A sample file, `test.gri`, is included on Canvas with this assignment. The `.gri` file contains the node coordinates, a list of boundary groups and faces, and a list of elements. The general format and definitions for a `.gri` file are given below:

```
nNode nElemTot Dim
for i = 1:nNode
  x(i) y(i) [z(i)]
nBGroup
for i = 1:nBGroup
  nBFace(i) nf(i) [Title(i)]
  for j = 1:nBFace(i)
    NB(i,j,1) .. NB(i,j,nf(i))
for i = 1:nElemGroup
  nElem(i) Order(i) Basis(i)
  for j = 1:nElem(i)
    NE(i,j,1) .. NE(i,j,nn(i))
```

`nNode`	Number of nodes; linear and high order
`nElemTot`	Total number of elements in all groups
`Dim`	Dimension: 2 or 3
`nBGroup`	Number of boundary groups (bgroups)
`nBFace(i)`	Number of boundary faces (bfaces) in group i
`nf(i)`	Number of linear nodes per bface on group i
`Title(i)`	String title of bgroup i (no spaces)
`NB(i,j,*)`	List of nf(i) linear nodes for bface j on bgroup i
`nElemGroup`	Number of element groups (egroups)
`nElem(i)`	Number of elements in egroup i
`Order(i)`	Geometry order of elements in egroup i (q)
`Basis(i)`	String defining the geometry interpolation basis
`nn(i)`	Number of nodes per element in egroup i
`NE(i,j,*)`	List of nn(i) nodes for element j in egroup i

Note, the number of element groups is not given explicitly in the file. The element reader stops when the total number of elements read in (sum of `nElem(i)`) reaches `nElemTot`. In this project, you only need to work with one element group. No convention for the numbering of boundary face nodes (`NB`) is specified, as the outward normal can be obtained from the geometry of the element containing these nodes. For each linear triangular element, `Order(i)` should be set to 1, and `Basis(i)` should be set to "TriLagrange". The numbering for the element nodes, `NE`, should be counterclockwise. Note, all indices in a `.gri` file are 1-based (they start at 1, not 0).

For this project, you should make the buildings separate boundary groups so that in the subsequent projects you can calculate forces on each individual building.

3 Test mesh and visualization

A test mesh, `test.gri` is provided with this assignment. This consists of a square domain divided into two triangles, as shown in Figure 2. For visualization of this mesh and your meshes, you can use one of the provided functions: `plotgri.py` or `plotgri.m`. Note that being able to plot a mesh is one test of the validity of the `.gri` file.

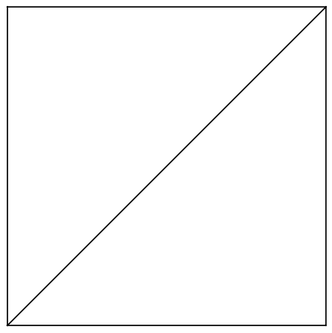


Figure 2: Test mesh.

4 Tasks

1. **[30%]** Generate a coarse triangular mesh of approximately 2000 elements for the provided domain, using a mesh generation program or a method of your choice. You should cluster nodes in the vicinity of the buildings, similarly to the mesh shown in Figure 1. Describe your approach, include an image of your mesh in your writeup, and include the `.gri` file, called `city0.gri` in your `.zip` archive.

2. **[30%]** Write a code that processes the mesh generated in the previous task and generates the following matrices (all 1-based):

   - `I2E`: a mapping from interior faces to elements. The number of rows is the total number of interior faces in the mesh. Each row contains four integers, `elemL, faceL, elemR, faceR`, where `elemL, elemR` are the indices of the two adjacent elements. To avoid ambiguity, define the left element (L) as the one with the smaller element index. `faceL, faceR` are numbers between 1 and 3 inclusive that encode which *local* face in each element corresponds to the interior face under consideration. In each element, local face number i is the face opposite local node i.

   - `B2E`: a mapping from boundary faces to elements and boundary groups. The number of rows is the total number of boundary faces. Each row contains three integers: `elem, face, bgroup`, where `elem` is the element adjacent to that boundary face, `face` is the boundary face's local face number within that element, and `bgroup` is the boundary index of that face. The boundary index should correspond to the order of boundary groups in the `.gri` file.

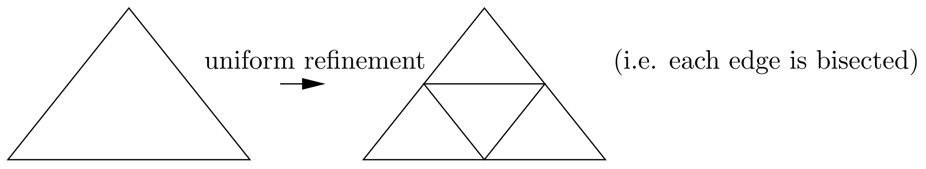
   - `In`: normal vectors for interior faces. The number of rows is the number of interior faces. Each row contains two floating-point numbers: the x and y components of the normal that points from the L to the R element, where L and R designations are defined in the `I2E` description.

   - `Bn`: normal vectors for boundary faces. The number of rows is the total number of boundary faces. Each row contains two floating-point numbers: the x and y components of the normal that points outward from the adjacent element.

   - `Area`: the area of each element. The number of rows is the total number of elements, and each row contains one number: the area of the element.

   Include in your writeup a description of the methods used, and a printout of these matrices for the test mesh included with this assignment.

3. **[30%]** Include support for uniform mesh refinement, in which each element is split into four subelements through edge bisection, as shown below.

   Generate two refinements of your original mesh, `city1.gri`, `city2.gri`. Include pictures of these meshes in your writeup, and the mesh files in your `.zip` archive.

5 Deliverables

You should turn in, electronically via Canvas,

1. A technical report as a `.pdf` file that describes your methods and results, and that answers the questions. The report should be professional, complete, and concise. **10%** of the project grade will be determined by professionalism: neatness, labeling of axes, spelling, etc.
2. Documented source files of all codes written for this project, as one `.zip` archive. You should provide all files necessary to run your program(s).

This is an individual assignment. You can discuss the project at a high level with each other, but you must turn in your own work.