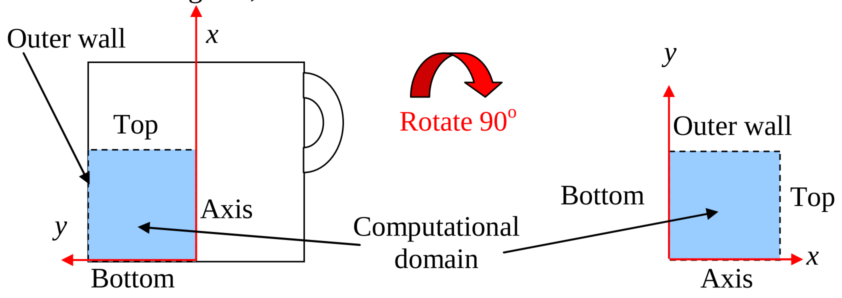
Gambit – Generation of grid for the teacup problem

Author: John M. Cimbala, Penn State University
Latest revision: 29 January 2008

Introduction and Instructions:
In this document is a procedure that enables you to generate a two-dimensional, structured rectangular grid that will be used with the CFD program Fluent to solve for rotationally symmetric swirling flow inside a teacup.

Overall description of the problem and grid:

1. Because of rotational symmetry, only one slice of the teacup geometry needs to be generated. The grid is a simple rectangle, representing one slice from the centerline of the cup to its rim. Fluent's axisymmetric option limits the geometry such that the *x*-axis must be the axis of rotation. Thus, the computational domain must be "tilted" or rotated 90 degrees, as sketched below.



2. In the sketch, *x* is in the direction of the axis of symmetry and *y* is actually the radius *r* from the *x*-axis.
3. In order to adequately resolve the boundary layer along the edges, grid points are clustered exponentially near the corners of the computational domain.

Log on and start Gambit

1. Log onto any Windows or Linux computer that has a valid license for the Ansys-Fluent family of programs – Gambit (grid generator) and Fluent (CFD solver).
2. Start Gambit. *Note*: If Gambit does not open properly, you may not have the proper path set up in your user files. If this is the case, check with the system administrator.
3. In Gambit, when the mouse cursor is placed over a button, a short description of the function of that button appears in the lower right window under *Description*. This is useful for understanding the commands.

Create some data points (vertices) for the desired geometry:

1. First, create a vertex at the origin, which is the center of the teacup. In *Geometry*, Vertex Command Button. Under *Vertex*, right mouse click the top left icon called Create Vertex and then From Coordinates. (In this learning module from now on, this type of command which requires a right click of the mouse is preceded by "R-". Here, for example, R-Create Vertex-From Coordinates).
2. In *Create Real Vertex-Global*, enter 0, 0, and 0 for *x*, *y*, and *z* coordinates respectively. You may label this vertex "origin" if desired, and Apply. A vertex is created, as indicated by a small plus sign at this location.
3. The radius of the teacup is 0.05 m (5 cm). Vertices need to be created at the other three corners of the domain. Create vertices with coordinates (0,0.05,0), (0.05,0.05,0), and (0.05,0,0). You may name them if you wish. These will be displayed on the screen, but they will hardly be distinguishable from the vertex at the origin since the dimensions are so small.
4. Zoom in to see the vertices more clearly. The easiest way to do this is Fit to Window in the *Global Control* area on the lower right of the Gambit screen.
5. In the main Gambit window near the upper left, File-Save. This will save your work so far. It is a good idea to do this every so often, especially after a major task is completed.

Create edges from these vertices to define the computational domain:

1. Some edges need to be created to define the teacup boundary. Under *Geometry*, Edge Command Button. (*Note*: If the option is already open, clicking it again will make the options disappear! If this happens, click it again.)
2. Under *Edge*, R-Create Edge-Straight. The straight edge option should be the default.
3. Select (shift + left mouse click) the two right-most vertices, type in a label ("top" is suggested, as on the figure), and Apply. A yellow line should appear indicating successful creation of this edge.
4. Similarly, following the figure, create straight edges called "outer wall", "bottom", and "axis".
5. Close the *Create Straight Edge* window.

Generate a face defining the computational domain:

1. Under *Geometry*, Face Command Button-R-Form Face-Wireframe.
2. Select the edges *in order* when creating a face from existing edges. (I like to select them in mathematically positive counterclockwise order). Select the four edges just created. These edges outline a closed face.
3. In *Create Face From Wireframe*, type in a label for this face if desired ("domain" is suggested), and Apply. If all went well, a pretty blue outline of the face should appear on the screen; this is a face, which is now ready to be meshed.

Specify the boundary types on all edges:

1. In order for the mesh to be properly transferred to Fluent, the edges must be assigned boundary types, such as wall, axis, etc. In *Operation*, Zones Command Button-Specify Boundary Types Command Button.
2. In the *Specify Boundary Types* window, change *Entity* to Edges (which may or may not be the default). In this problem, which is 2-D, boundary conditions are applied to edges rather than to faces.
3. Select the left-most edge of the computational domain, labeled "bottom". Change *Type* to Wall, if necessary (wall is usually the default) and type in the name "bottom". Apply. Some words indicating this boundary condition should appear on the screen.
4. Similarly, make the top-most edge a wall called "outer wall" and the right-most edge a wall called "top".
5. Finally, select the bottom-most edge called "axis". Label it "axis" and make its *Type* Axis. Apply.
6. Close the *Specify Boundary Types* window, and save your work.

Define node points along the edges:

1. In *Operation*, Mesh Command Button-Edge Command Button. *Mesh Edges* should be the default window that opens; if not, Mesh Edges.
2. Select all four edges, and in *Mesh Edges*, change the *Spacing* option from Interval Size to Interval Count. Enter the number 100 as the *Interval Count*.
3. It is desirable to cluster or bunch nodes close to the edges. This is accomplished by changing *Ratio* (in the *Grading* section of the *Mesh Edges* window). First, select the Double sided option. Then, change both Ratio 1 and Ratio 2 to 1.1. Apply. Blue circles should appear at each created node point along each edge.
4. Now all the necessary edges have been assigned nodes; Close the *Mesh Edges* window. Save your work.

Generate the mesh on the face:

1. Under *Mesh*, Face Command Button. The default window should be *Mesh Faces*. If not, Mesh Faces.
2. Select the face by shift clicking on one of its edges. *Elements* should be Quad by default; if not, change it. Also change *Type* to Map if necessary. The *Spacing* options is ignored since nodes have already been defined on the appropriate edges of this face.
3. Generate the mesh by Apply. If all goes well, a structured mesh should appear. Zoom in to see how the cells are nicely clustered near the corners. This will help Fluent to resolve the boundary layers along the edges of the computational domain.
4. You can now Close the *Mesh Faces* window.

Write out the mesh in the format used by Fluent, and then exit Gambit:

1. In the main Gambit window, File-Export-Mesh. (The default file name can be changed at this point if desired.) Check the option to export a 2-D mesh, and Accept.
2. When the Transcript (at lower left) informs you that the mesh is done, File-Exit-Yes.
3. The mesh file should now be ready for use by Fluent.