ME3700 – Fall 2001
Running Gambit & Fluent from CADE

Running Gambit

Before performing a calculation using Fluent, a computational domain must be generated to define the geometry of problem as well as grid resolution.

Log on and run Gambit:

1. Log on to a CADE UNIX machine.
2. Create a Unix shell.
3. Create a new directory called Fluent, so that your Fluent project resides on this directory. To do so, Enter "mkdir Fluent" and then go to this new directory by entering "cd Fluent".
4. Also create a new directory called Gambit, as a subdirectory of Fluent. Go to the Gambit subdirectory by typing "cd Gambit".
5. Gambit may now be started. Enter "gambit -id flatplate &". After a while (sometimes a couple minutes!), the main Gambit window should appear on your screen. Note the "&" at the end of the command, which tells UNIX to run this in background mode. This will free up your UNIX window for other commands while Gambit continues to run.
6. In the upper left of the screen, Solver-Fluent 5. This selects Fluent as the solver for which gambit will eventually write the mesh file.

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

1. First, create a vertex at the origin, which is the leading edge of the flat plate. In *Geometry*, Vertex Command Button-R-Create Vertex-From Coordinates (where -R- indicates a right mouse click). In *Create Real Vertex-Global*, enter 0, 0, and 0 for x, y, and z coordinates respectively. Type an appropriate label for this vertex, i.e. "start of plate", and Apply. A vertex is created, as indicated by a small plus sign at this location.
2. Create another vertex with coordinates (-0.1,0,0). You can call it "lower left" or something like that.
3. In similar fashion, enter the coordinates of the other points defining the limits of the computational domain, i.e. (0.5,0,0), (0.5,H,0), and (-0.1,H,0), where H is the height of the computational domain. (This will ensure that the farfield boundary is "far enough away" from the boundary layer.) You can name them appropriately ("lower right", "upper right", and "upper left" are suggested), or you can let Gambit label these vertices for you by default.
4. Finally, define a point at the top, but at x = 0 so that the grid can be nice and rectangular. Enter vertex coordinates (0,H,0) and call it "upper middle" or something descriptive of its location.

5. At this point, it is wise to zoom in somewhat to see the vertices more clearly. There are two ways to do this. You can zoom in and move with the right mouse

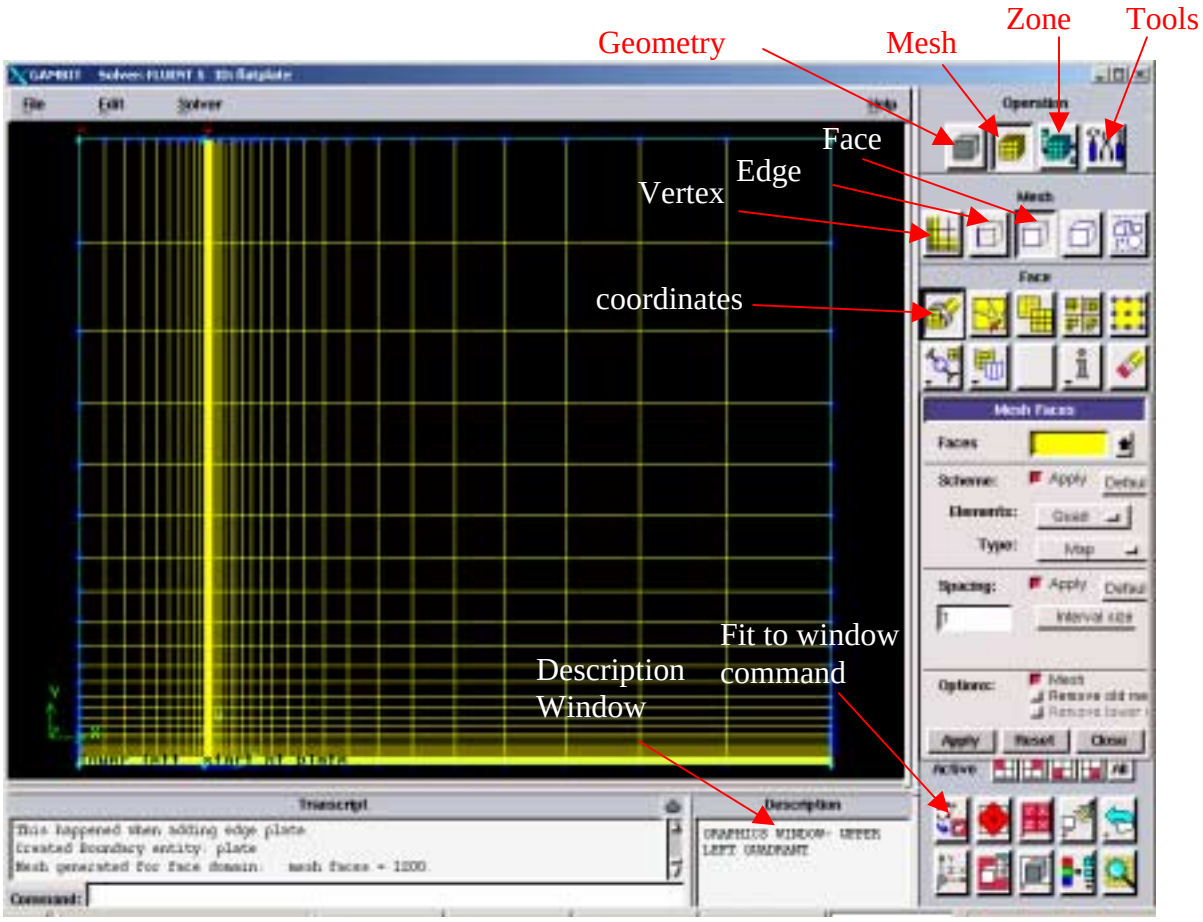

Figure 1 – Gambit screen with various command icons labeled along with the meshed grid to be imported into Fluent.

6. and middle mouse, respectively. Or, to fit all the geometry into the available screen space (a very handy tool!), in *Graphics/Windows Control* (near the bottom right), Fit to Window.
7. 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. Now some edges will be created to define the flat plate and the rectangular boundary surrounding the plate. Under *Geometry*, Edge Command Button. (Note: If the button is already open, clicking it again will make the options disappear! If this happens, click it again.)
2. Under *Edge*, right mouse click the top left icon, which is called Create Edge, and then Create Straight Edge. (In this learning module from now on, this type of command which requires a right click of the mouse will be denoted by R-Create Edge-Create Straight Edge.)
3. In *Create Straight Edge*, make sure the text area called *Vertices* is highlighted - if not, click in that area. Vertices are selected by Shift + left mouse click. Select the two rightmost vertices, first the one we called lower right, and then the one we called upper right.
4. Type in an appropriate label for the edge about to be created. I suggest "outlet" or something equally descriptive. Apply. Gambit will create a straight edge between these two vertices. A yellow line should be drawn on the screen connecting the vertices.
5. In similar fashion, create two separate edges along the top, one along the top right of the computational domain, and one along the top left of the computational domain. If desired, label these "top right" and "top left" respectively.
6. Generate an edge along the left side of the domain, and call it "inlet".
7. For the bottom of the domain, create two separate edges - one from the lower left vertex to the origin (leading edge of the flat plate), and one from the origin to the lower right vertex. Recommended labels are "leading" and "flat plate" or "bottom left" and "bottom right".
8. Now Close the *Create Straight Edge* window.
9. 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.

Generate a face above the flat plate from the available edges:

1. Under *Geometry*, Face Command Button-R-Create Face-Wireframe.
2. It is important to select the edges *in order* when creating a face from existing edges. (I like to select them in mathematically positive counterclockwise order). Select the rightmost edge first (using shift+left mouse button), followed by the top right edge, the top left edge, the leftmost edge, the edge upstream of the flat plate, and then finally the edge representing the plate itself. 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.

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 the leftmost (inlet) edge(using shift+left mouse button), and in *Mesh Edges*, change the *Spacing* option from Interval Size to Interval Count. Enter the number 30 as the *Interval Count*. It is desirable to cluster or bunch nodes close to the plate. This is accomplished by changing *Ratio* (in the *Grading* section of the *Mesh Edges* window). A Ratio of 1.2 is recommended so that there are lots of nodes near the bottom. Note: If the bunching is backwards from the way you desire, click on the Reverse button. Apply. Blue circles should appear at each created node point along that edge.
3. Put the same node distribution on the rightmost edge (the outlet). You may have to Reverse the distribution. Don't forget to Apply; otherwise the node distribution is not saved.
4. Select the left portion of the bottom edge (the part upstream of the plate). Change *Interval Count* to 10. This time, we want to cluster the nodes near the leading edge of the plate. Keep *Ratio* at 1.2, and Reverse if necessary so that the clustering is correct. Apply.
5. Similarly, define 30 node points along the plate itself, clustered near the leading edge of the plate.
6. Define the same distribution on the two top edges as on the two bottom edges so that the mesh lines will be vertical. This is not necessary, but makes the mesh look nicer, and rectangular cells lead to better convergence. Don't forget to Apply each time, or the nodes will not actually be saved.
7. When all edges have been assigned nodes, Close the Mesh Edges window.

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, inlet, outlet, etc. Actual numerical values, such as inlet velocity, will be specified as boundary conditions from within Fluent itself. In *Operation*, Zones Command Button-Specify Boundary Types Command Button.
2. In the *Specify Boundary Types* window, change *Entity* to Edges (the default is usually Faces). In this problem, which is 2-D, boundary conditions will be applied to edges rather than to faces.
3. Select the left-most edge of the computational domain, which will become an inlet. Change *Type* to Velocity_Inlet, and type in the name "inlet". Apply. Some words indicating this boundary condition will appear on the screen.
4. Select the right-most edge. In similar fashion, make this a Pressure Outlet named "outlet". Be sure to Apply, or the boundary condition will not actually be changed.

5. Select *both* of the top-most edges. These two edges will be combined into one boundary for Fluent, which will also be a pressure outlet. The label "top" is suggested. Apply.
6. Select the horizontal edge running from the lower left to the origin (the one called "leading"). Select its Type as Symmetry, and label it "leading". Apply.
7. Finally, Select the edge defining the flat plate itself. Name this boundary type "flat plate", and change *Type* to Wall. Apply.
8. Now Close the *Specify Boundary Types* window.

Generate the mesh on the face:

1. Under *Operation*, Mesh Command Button-Face Command Button. The default window that pops up 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 will be ignored since nodes have already been defined on all 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 bottom. This will help Fluent to resolve the boundary layer along the flat plate.
4. Zoom back out so that the entire mesh can be clearly seen. This is most easily accomplished by clicking on Fit to Window in the *Graphics/Windows Control* (near the bottom right of the screen).
5. 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-Accept. (The default file name can be changed here if desired, but the default name is acceptable in this case.)
2. When the Transcript (at lower left) informs you that the mesh is done, File-Exit-Yes.
3. When Gambit exits, the UNIX window will re-appear. Enter "ls -la" to see the files created by Gambit. The flatplate.msh file is the one to be used by Fluent.
4. It is a good idea to move this file into the Fluent directory, so that it is ready to be read in. Enter "cd ..", which will move your working directory up one level to the Fluent directory. To verify, Enter "pwd" (print working directory).
5. Now enter "mv Gambit/flatplate.msh .", which will move the file from the Gambit subdirectory into the working directory. Don't forget the period at the end of this command! Enter "ls -la" to verify that the file is now in the Fluent directory.

Running Fluent

Log on and launch Fluent:

1. If already logged in, enter "pwd". This Unix command will print your working directory. If your working directory is not Fluent, enter "cd ~/Fluent" to make Fluent your working directory. If logged in and at the correct Fluent working directory, skip to step 5.
2. Log on to a computer which has access to the Fluent software.
3. Enter "cd Fluent" to change the working directory to Fluent. (Note: It is assumed that the grid was generated from the Fluent/Gambit subdirectory, and then moved to the Fluent directory, as instructed in the Gambit learning module.)
4. Enter "fluent 2d &". After a few seconds, the main *Fluent* window should appear on your screen. (The & symbol lets Fluent run in background mode so that the Unix shell is still usable.)

Read the grid points and geometry of the flow domain:

1. Select File-Read-Case. There should be a file shown on the right side called flatplate.msh. Highlight this file (i.e. click on it), and OK . Fluent will read in the grid geometry and mesh that was previously created by Gambit. Some information is displayed on the main screen. If all is read well, it should give no errors, and the word "Done" should appear.
   Note - If you have trouble reading your grid, you probably made a mistake in Gambit somewhere. Go back and try to either modify your grid or remake it following the module above.
2. Check the validity of the grid: Grid-Check. If the grid is valid, no errors should appear. If there are errors, you may have done something wrong in the grid generation, and will have to go back and regenerate the grid.
3. Look at the grid to make sure it is correct. Display-Grid-Display . A new graphical display window opens up showing the grid. If this window is too big, re-scale it by dragging the edges of the window. It is best if the graphical display window is small enough that both it and the *Fluent* window are visible simultaneously. The *Fluent* window and/or the graphical display window may need to be moved to accomplish this.
4. The graphical display can be zoomed-in or zoomed-out with the *middle* mouse button. If you start on the *lower left* and draw a rectangle with the middle mouse button towards the upper right, the display will zoom in on what is included in the rectangle. Meanwhile, the *left* mouse button can be used to drag the image to a new location. If you draw a rectangle *backwards* with the middle mouse button, i.e. from right to the left, it will zoom out. Zoom in if necessary until the grid is shown nicely in the window. Close the *Grid Display* window; the display itself will remain. Note that each boundary condition type should have it’s own color (red for pressure, violet inlet, yellow symmetry and white for the plate).

Generate lines at x = 0.10 m and at x = 0.30 m:

1. The boundary layer profile will be examined in detail at three x locations along the flat plate, namely at x = 0.10 m, 0.30 m, and 0.50 m. The last of these is the outlet of the domain, but lines need to be defined within Fluent for the first two.
2. In the main *Fluent* window, Surface-Line/Rake. Type in the desired starting and ending x and y locations of the vertical line, i.e. a vertical line going from (0.10,0) to (0.10,H).
3. The *New Surface Name* should be assigned at this point. It is suggested that this line be called "profile0.10" or something descriptive of its intended purpose.
4. Click on Create to create the line.
5. Similarly, create a line at x = 0.30; a suggested label is "profile0.30".
6. To view these newly created lines, return to the main *Fluent* window, and Display-Grid-Display. Unselect (by left mouse click) the default interior, and select the newly created lines instead. Display. The lines should be visible at the appropriate locations. If not, create them again more carefully.
7. Now Close both the *Line/Rake Surface* window and the *Grid Display* window.

Define the fluid as liquid water:

1. The default fluid is air, which must be changed to water. In the main *Fluent* window, Define-Materials-Database. Select water-liquid from the *Fluid Materials* drop down list. Copy.
2. Write down the density and viscosity of liquid water, which are needed later to calculate Reynolds numbers, etc. Close.
3. Now Close the *Materials* window. Caution: This has added liquid water into the list of fluids available as boundary conditions, but has not actually changed the fluid from air to water. This will be done when specifying the boundary conditions.

Define the boundary conditions:

1. Now the boundary conditions need to be specified. In Gambit, the boundary conditions were declared, i.e. wall, velocity inlet, etc., but actual values for inlet velocity, etc. were never defined. This must be done in Fluent. In the main *Fluent* window, click on Define-Boundary Conditions, and a new *Boundary Conditions* window will pop up.
2. Select fluid and Set. Choose water-liquid as the *Material Name* from the drop-down list of material names. OK.
3. The default boundary condition for the flat plate (wall) is okay (stationary wall), so nothing needs to be done to it.
4. Likewise, the default boundary conditions for the symmetry plane (symmetry) and the two pressure outlets (outlet and top) are okay, so nothing needs done to them.
5. Select inlet, which is the left side of the computational domain. Set. The *Velocity Specification Method* should be Magnitude and Direction by default. If not,

change it. Change the Velocity Magnitude to 2.0 m/s. The default flow directions are for zero angle of attack, which is what is desired here, so OK.

6. Boundary conditions are complete, so Close the *Boundary Conditions* window.

Set up some parameters and initialize:

1. In the main *Fluent* window, Define-Models-Viscous . Laminar flow is the default, so we really don't need to do anything here. Later on, however, we will try turbulent flow calculations; this is where the turbulence models are specified in Fluent. OK.
2. Now the convergence criteria need to be set. As the code iterates, "residuals" are calculated for each flow equation. These residuals represent a kind of average error in the solution - the smaller the residual, the more converged the solution. In the main *Fluent* window, Solve-Monitors-Residual. In the *Residual Monitors* window that pops up, turn on Plot in the *Options* portion of the window. The Print option should already be on by default. Here, Print refers to text printed in the main *Fluent* window, and Plot causes the code to plot the residuals on the screen while the code is iterating.
3. Since there are three differential equations to be solved in a two-D incompressible laminar flow problem, there are three residuals to be monitored for convergence: continuity, x-velocity, and y-velocity. The default convergence criteria are 0.001 for all three of these. Experience has shown that this value is generally not low enough for proper convergence. Change the *Convergence Criterion* for continuity from 0.001 to 1.E-06 (click on the number, and then on the window that pops up, type in the new value, and OK to set it). There is no need to change the other convergence criteria.
4. To apply the changes, OK, which will also close the *Residual Monitors* window.
5. Experience has shown that for most problems, the solution converges better if the under-relaxation factors are reduced from their default values. Without going into a lot of detail, reduced under-relaxation factors sort of "damp out" changes in the solution as it iterates. Sometimes these changes are too aggressive, and adversely affect convergence. In the main *Fluent* window, Solve-Controls-Solution. Reduce the *Under-Relaxation Factors* for pressure and momentum by a factor of two from their default values. OK.
6. In the main *Fluent* window, Solve-Initialize-Initialize. The default initial values of velocity and gage pressure are all zero. These are good enough for this problem. Init and Close.
7. At this point, and every so often, it is wise to save your work. In the main *Fluent* window, File-Write-Case & Data. In the *Select File* window which pops up, the default file name is flatplate.cas, which should be changed to something more descriptive ("laminar_bl.cas" is suggested). Note that Case & Data refers to both the case (the grid plus all boundary conditions and other specified parameters) and the data (the velocity and pressure fields calculated by the code. The code will actually write out two files, laminar_bl.cas and laminar_bl.dat.
8. If not on by default, turn on the option to Write Binary Files (to save disk space). To save even more disk space, the files can be compressed by adding a "gz" at the

end of the file name. The complete file name should be "laminar_bl.cas.gz". This is done by typing “gunzip laminar_bl.cas”.

OK to write the file onto your directory.

Iterate towards a solution:

1. In the main *Fluent* window, Solve-Iterate to open up the *Iterate* window. Change Number of Iterations to 200, and Iterate . The main screen will list the residuals after every iteration, while the graphical display window will plot the residuals as a function of iteration number. The residuals may rise at first, but should slowly start to fall. It is normal for the residuals to fluctuate up and down. Do not be concerned if there are reverse flow warnings; these will disappear in time.
2. At the end of 200 iterations, check to see how the solution is progressing. In the main *Fluent* window, Display-Velocity Vectors-Display . The graphical display window will show the velocity vectors. Zoom in with the middle mouse, as described above, to view the velocity field in more detail if desired. In particular, view the boundary layer near the end of the plate. Is it starting to look like a boundary layer profile?
3. Iterate some more - (To restart the iteration, either find the *Iterate* window, which is probably hidden under some other windows at this point, or click again on Solve-Iterate to re-open the *Iterate* window.) In the *Iterate* window, set Number of Iterations to 100, Apply, and Iterate.
4. Check the velocity vectors, as described above after another 100 iterations. It is wise to move the *Iterate* window someplace out of the way of the other windows so you can easily restart the iteration. The residuals may go up sometimes. This is normal, as the code attempts to zero in on a solution - after a while the residuals should again decay or level off. If the residuals all go below the convergence criteria, the calculations will stop. In some cases, however, the residuals reach a lower limit, and further iterations don't improve the solution.
5. 200 to 300 iterations should be sufficient for the first attempt. Before iterating further, the grid must be refined.

Refine the grid:

1. Our grid is not tight enough near the wall to accurately resolve the boundary layer, especially near the front of the flat plate where the boundary layer is very thin. Fortunately, Fluent has an "Adapt" feature that automatically adds grid points where needed for better resolution. There are several options for grid adaptation - we shall adapt by velocity gradient.
2. In the main *Fluent* window, Adapt-Gradient. In the new *Gradient Adaption* window, select Gradients of Velocity.
3. Select the Compute option. Minimum and maximum velocity gradients will appear in the window.

4. As a good rule of thumb, set the Refine Threshold to about 1/10 of the maximum gradient, and the Coarsen Threshold to about 1/1000 of the Refine Threshold. Enter these values in the appropriate text boxes.
5. Now select Mark. The main *Fluent* window will display how many cells have been selected for refining and coarsening. The coarsening cells can be ignored since Fluent is unable to coarsen the original grid - it can only refine the original grid.
6. Optional: If you want to see where the grid will be adapted, click Manage-Display. Areas destined for grid refinement will be highlighted.
7. Back in the *Gradient Adaption* window, Adapt. You will be prompted about hanging-node mode. Answer Yes to continue. The main *Fluent* window will display some information about the grid adaptation.
8. Optional: To see what the refined grid looks like, you can select Display-Grid-Display from the main *Fluent* window, and zoom in close to the wall. You should see some new cells near the wall (especially near the leading edge of the flat plate) where velocity gradients are highest.
9. The *Gradient Adaption* window can be closed at this point. Or, better yet - move it somewhere on the screen where it can be accessed again, since we will need it again later.

Iterate some more:

1. If the *Iterate* window is still available, go to it. If not, Solve-Iterate from the main *Fluent* window to re-open the *Iterate* window. Change Number of Iterations to about 200, Apply, and Iterate.
2. At the end of these iterations, check to see how the solution is progressing. In the main *Fluent* window, Display-Velocity Vectors-Display. The graphical display window will show the velocity vectors.

Examine the velocity profiles in detail:

1. At this point, the velocity profile at three desired downstream locations (x = 0.10, 0.30, and 0.50 m) will be plotted and examined in detail.
2. In the main *Fluent* window, select Plot-XY Plot. A window called *Solution XY Plot* will open up. In this window, at the lower right corner, select (highlight) the surfaces we created above, i.e. the ones called profile0.10 and profile0.30. Also select outlet, which is at x = 0.50 m.
3. In the upper left corner of the window, turn ***off*** Position on X Axis, and turn ***on*** Position on Y Axis. This will make the y-axis the y position on the plot, as desired.
4. In the upper middle part of that window, set Plot Direction to X = 0 and Y = 1. This will make the y-coordinate position appear on the y-axis, as desired for a standard velocity profile plot.
5. The upper right part of the window selects the variable to be plotted. The *Y Axis Function* will be set automatically to Position, and should be left alone. For the *X Axis Function*, select Velocity and X-Velocity. Plot.

6. The boundary layer profiles should be there, but may be hard to see since the y-axis extends all the way to the upper boundary of the computational domain. The axes limits can be changed as follows: Axes. Choose X if necessary (X should already be the default). ***Unselect*** Auto Range, and select Major Rules. Set *Range* from 0 to 2.2 m/s. Apply. (Nothing will happen to the plot yet, so don't panic.)
7. Now choose the Y axis. Unselect Auto Range and select Major Rules for this axis as well. Set the range from 0 to around 0.005 m or so, such that the entire boundary layer should be visible on the plot.
8. To make the scale more readable, change the Number Format to Type = Float and Precision = 3. Apply.
9. Back in the *Solution XY Plot* window, Plot. Adjust the axes and/or number format as desired to obtain a nice plot of the boundary layer profiles. If done correctly, all three profiles should be visible on the plot, and the growth of the boundary layer with downstream distance should be apparent.

Iterate towards a final solution:

1. At this point, the boundary layer should be reasonable, but more grid refinement and iteration may be required. If there are less than about 10 data points within the thickness of the boundary layer at any of the three profile locations, the grid should be refined again.
2. Refine the grid and re-iterate as necessary to obtain a final solution. Each time you adapt the grid, you must re-calculate the gradients (Compute), re-adjust the refine threshold (again, it should be set to about 1/10 of the maximum gradient), Mark, and Adapt-Yes.
3. Another useful way to refine the grid is the boundary option. Adapt-Boundary. Turn off (unselect) all the *Boundary Zones* except the plate itself, which is the only boundary on which we wish to refine the grid. Change *Number of Cells* to 2, and Adapt. Yes, hanging nodes are okay.
4. Iterate at least 200 to 300 iterations after each grid adaption. The residuals will rise dramatically after an adaption, but will decay as the solution adjusts itself to the newly refined grid.
5. Caution: Don't refine too much, or the computations will take too much CPU time. Note that every time you refine the grid, the computer must calculate the flow field at more grid points, requiring longer for each successive iteration.
6. After refinement and iteration, look at the velocity profiles (Plot-XY Plot-Plot from the main *Fluent* window, or simply Plot if the *Solution XY Plot* window is still available).

Save your velocity profiles and your calculations:

1. In the main *Fluent* window, File-Write-Case & Data. In the *Select File* window which pops up, the default file name should be "laminar_bl.cas.gz", as previously entered. OK to write the file onto your directory. OK again since it is okay to overwrite these files.
2. From the *Solution X-Y Plot* window, Plot. When the plot is to your liking, the plot of the boundary layer profiles will be saved.
3. Before saving the plot, your name and a short description should be added to the title. Click with the left mouse button just below the existing title in the bottom left of the plot. A cursor should appear. Type your name, identify thesolution as laminar, and type the Reynolds number on your plot.
4. In the main *Fluent* window, File-Hardcopy-Save, name the file (something like "laminar_profiles.ps" is appropriate), OK, and Close. The postscript file generated can be printed out from the Unix shell later.
5. The data points along the x = 0.1,0.3, and 0.50 m line will now be saved to an ASCII file for further analysis and comparison to predictions. From the *Solution XY Plot* window, select all three locations for profiles. Write to File, and Write. Name the file something ("laminar_profile.txt" is suggested) and OK.
6. Finally, save a plot of the residuals. In the main *Fluent* window, Plot-Residuals-Plot. Then, File-Hardcopy. It is suggested that *Coloring* be changed to Monochrome since the printer prints in black and white only. Save, name the file (something like "laminar_residuals.ps" is appropriate), OK, and Close.

Calculate the drag force and drag coefficient on the body:

1. In order to calculate drag coefficient correctly, the proper reference values for area, velocity, etc. need to be defined. In the main *Fluent* window, Report-Reference Values. A window called *Reference Values* will open. The proper reference area to use is the planform area of the plate. Planform area is defined as the area seen from above, in this case L times b, where L is the plate length and b is the depth or span of the plate. Assume unit depth (b = 1.0 m) for convenience so that drag force is per unit depth. *Depth* should already be set to 1.0 m by default.
2. Change *Area* to L times b, change *Velocity* to the freestream velocity of the flow, and change *Length* to L, the length of the flat plate.
3. Unfortunately, the reference values for density and viscosity are those of air, the default fluid, even though we defined our working fluid as liquid water. (This is apparently a bug in Fluent.) Change *Density* and *Viscosity* to their proper values. (These values for liquid water were written down previously, but can also be found from Define-Materials, and selecting water-liquid under *Fluid Materials*.) All other reference values should be okay, so OK.
4. In the main *Fluent* window, Report-Forces. In the window which pops up, the only available *Wall Zone* is flat-plate , which is the surface of the flat plate, so the default is okay. You will need to enter the components of the force vector, however. These are simply the x and y components of a unit vector pointing in the

direction of the desired force. For example, setting the x-component to 1 and the y-component to 0 would cause Fluent to calculate the force acting on the wall in the x-direction. In our problem, we want drag force, which is defined as parallel to the freestream direction. Enter the appropriate x and y components so that the result is the drag force. Note that the calculated force will actually be in units of N/m since this is a 2-D problem (force per unit span).

5. Click on Print to perform the calculation. The results will be printed to the main *Fluent* window. You may need to widen that window to see the entire length of the line. Notice that the force is broken into a pressure component and a viscous component. The pressure component is zero. Why?
6. Write down the total drag force and drag coefficient. (Remember that this is the drag on one side (the top) of the plate, all the way to the end of the computational domain, i.e. from x = 0 m to x = 0.50 m.) Does the calculated drag coefficient at this Reynolds number agree with published data?
7. To exit the *Force Reports* window, Close.

These Fluent Modules are modifications of the great work done by Professor John M. Cimbala at Penn State (http://spike.me.psu.edu/~cimbala/Learning/learning.htm).